The Prague Bulletin of Mathematical Linguistics
NUMBER 108 JUNE 2017

EDITORIAL BOARD

Published twice a year by Charles University (Prague, Czech Republic)

Editorial office and subscription inquiries:
ÚFAL MFF UK, Malostranské náměstí 25, 118 00, Prague 1, Czech Republic
E-mail: pbml@ufal.mff.cuni.cz

ISSN 0032-6585

List of reviewers

Marianna Apidianaki, LIMSI-CNRS
Mihael Arcan, Insight @ NUI Galway
Eleftherios Avramidis, German Research Center for Artificial Intelligence (DFKI)
Wilker Aziz, University of Amsterdam
Bogdan Babych, Centre for Translation Studies, University of Leeds
Parnia Bahar, RWTH Aachen University
Loïc Barrault, LIUM - University of Le Mans
Laurent Besacier, Laboratoire d'Informatique de Grenoble
Frederic Blain, University of Sheffield
Hervé Blanchon, Laboratoire d'Informatique de Grenoble - Equipe GETALP
Michael Bloodgood, The College of New Jersey
Ondřej Bojar, Charles University
Fabienne Braune, University of Stuttgart (Germany)
Iacer Calixto, CNGL Dublin City University & ICT Chinese Academy of Sciences
Michael Carl, Copenhagen Business School
Francisco Casacuberta, Universitat Politècnica de València
Helena Caseli, Federal University of São Carlos (UFSCar)
Daniel Cer, Stanford University
Boxing Chen, NRC-CNRC
Colin Cherry, National Research Council Canada
David Chiang, University of Notre Dame
Daniel Dahlmeier, SAP Innovation Center Network
Fahim Imaduddin Dalvi, Qatar Computing Research Institute
Jinhua Du, CNGL, School of Computing, Dublin City University
Christian Dugast, tech2biz
Kevin Duh, Johns Hopkins University
Nadir Durrani, University of Edinburgh
Andreas Eisele, European Commission, DGT
Cristina España-Bonet, UdS and DFKI
Miquel Esplà, Universitat d'Alacant
Mireia Farrús, Universitat Pompeu Fabra
Christian Federmann, Microsoft Research
Orhan Firat, Google Research
Mark Fishel, University of Tartu
Mikel Forcada, Universitat d'Alacant
George Foster, NRC
Alexander Fraser, LMU Munich
Markus Freitag, IBM Research
Federico Gaspari, Dublin City University
Teresa Herrmann, Fujitsu

PBML

The Prague Bulletin of Mathematical Linguistics
NUMBER 108 JUNE 2017

CONTENTS

Articles

PBML

The Prague Bulletin of Mathematical Linguistics
NUMBER 108 JUNE 2017

EDITORIAL

Foreword from the president of the European Association for Machine Translation

As president of the European Association for Machine Translation (EAMT), it is a great pleasure for me to write the foreword to this special issue of the Prague Bulletin of Mathematical Linguistics, which also serves as the proceedings of the research track of the 20th annual conference of the EAMT in Prague, the Czech Republic.

The EAMT started organizing annual workshops in 1996; later, these workshops became annual conferences, and were hosted all around Europe. Years ago, the venue was steadily moving from west to east: from Barcelona (2009) to Saint-Raphaël (2010) to Leuven (2011) to Trento (2012) to Dubrovnik (2014) —after skipping one year to host the successful world-wide MT Summit 2013 in Nice— , but recently turned around to go west again at Antalya (2015), to go to Riga (2016) and now Prague (2017). Again, you have guessed: EAMT 2018, our 21th annual conference, will surely be west from Prague. It will be announced at EAMT 2017 shortly after I am writing these lines. Those who miss our conference, will find out by visiting our Association's website, EAMT.org.

By the way, if you have not done so yet, please consider joining the EAMT. Our membership rates are low, particularly for students, and have not increased since the EAMT's inception. You will benefit from discounts when attending not only our conferences, but also the conferences held by our partner associations the Asia-Pacific Association for Machine Translation (AAMT) and the Association for Machine Translation in the Americas (AMTA). You will also have an exclusive chance to benefit from funding for your activities related to machine translation. And perhaps you can get even more involved and participate in serving the European machine translation community by becoming a member of the Executive Committee of the EAMT.

But let me go back to EAMT 2017. As in previous conferences, it is great to see the strong programme put together by our programme chairs: Alexander Fraser, research track chair, and Kim Harris, user track chair. As in previous editions, there will also be a projects and products session which showcases the advance of machine translation in Europe. And, last but not least, I also feel very fortunate to have João Graça from Unbabel as our invited speaker.

EAMT 2017 would have never been possible without the generous offer to host and the hard work subsequently done by the local organizing committee at the well-known machine translation group of Charles University, headed by Jan Hajič and Ondřej Bojar. I warmly thank them all, also because they have made it possible for the research papers of our conference to become a special issue in an open-access journal which is well-known to the machine translation community; I'm sure this will multiply the impact of the research presented in our conference.

It is also with great pleasure that I thank our sponsors: Memsource (gold sponsor), Star Group (silver sponsor), text&form (bronze sponsor), and Prompsit and Apertium (supporting sponsors).

Finally, I would like to thank EAMT 2017 attendees for coming to Prague. I hope the conference leads to new friendships and fruitful collaboration.

Mikel L. Forcada
EAMT President

mlf@ua.es

Preface from the Program Chair (Research Track)

It is my pleasure to welcome you to the 20th annual conference of the European Association for Machine Translation (EAMT) to be held in Prague, Czech Republic. I have really enjoyed serving as a program chair for this edition of the conference. The EAMT conference has become the most important event in Europe in the area of machine translation for researchers, users, professional translators, etc. As in previous editions, the conference is organised around three different tracks: research, user and projects/products. The research track concerns novel and significant research results in any aspect of machine translation and related areas while the user track reports users' experiences with machine translation, in industry, government, NGOs, etc. Finally, the project and product track offers projects and products the opportunity to be presented to the wide audience of the conference. This year we have received 49 submissions to the research track, 15 submissions to the user track and 25 descriptions of projects and products. Overall, submissions come from 35 different countries. Each submission to the research and user tracks was peer-reviewed by at least three independent members of the Programme Committee. In the research track 29 papers out of 49 (59%) were accepted for publication.

Aside from regular papers from the three tracks, the program includes an invited talk by João Graça, CTO and co-founder of Unbabel on the hot topic of using AI techniques for end-to-end translation. We will also have a presentation by the winner of the EAMT Best Thesis Award.

We would like to thank the Program Committee members and additional reviewers, whose names are listed above, for their high quality reviews and recommendations. These have been very useful for the Program Chairs to make decisions. We would also like to thank all the authors for trying their best to incorporate the reviewers' suggestions when preparing the camera ready papers. For those papers that were not accepted, we hope that the reviewers' comments will be useful to improve them. Special thanks to Kim Harris, user track, Mikel L. Forcada, projects and products track, and also to Lucia Specia and Ondřej Bojar for helping out in many ways. And finally, thanks to Dušan Variš, for his hard work in putting together this special issue of PBML.

Alexander Fraser
CIS, LMU Munich
EAMT 2017 Program Chair (Research Track)

fraser@cis.uni-muenchen.de

PBML

The Prague Bulletin of Mathematical Linguistics
NUMBER 108 JUNE 2017 13–25

Empirical Investigation of Optimization Algorithms in Neural Machine Translation

Parnia Bahar, Tamer Alkhouli, Jan-Thorsten Peter,
Christopher Jan-Steffen Brix, Hermann Ney

Human Language Technology and Pattern Recognition Group,
RWTH Aachen University, Ahornstraße 55, 52074 Aachen, Germany

Abstract

Training neural networks is a non-convex and a high-dimensional optimization problem. In this paper, we provide a comparative study of the most popular stochastic optimization techniques used to train neural networks. We evaluate the methods in terms of convergence speed, translation quality, and training stability. In addition, we investigate combinations that seek to improve optimization in terms of these aspects. We train state-of-the-art attention-based models and apply them to perform neural machine translation. We demonstrate our results on two tasks: WMT 2016 En→Ro and WMT 2015 De→En.

1. Introduction

Training a neural network involves the estimation of a huge number of parameters. Ideally, optimization seeks to find the global optima, but in such a non-convex problem, global optimality is given up and local minima in the parameter space are considered sufficient to obtain the models that generalize beyond the training data (Goodfellow et al., 2016, Chapter 8). Besides obtaining better performance, choosing an appropriate optimization strategy could accelerate the training phase of neural networks and brings higher training stability.

Modeling and training problems are two major issues involved in Neural Machine Translation (NMT) systems. (Junczys-Dowmunt et al., 2016) state that averaging the parameters of a few best models from a single training run, considered as a single model, leads to improvement in terms of both translation metrics and perplexity. This

 Corresponding author: bahar@i6.informatik.rwth-aachen.de
Cite as: Parnia Bahar, Tamer Alkhouli, Jan-Thorsten Peter, Christopher Jan-Steffen Brix, Hermann Ney. Empirical Investigation of Optimization Algorithms in Neural Machine Translation. The Prague Bulletin of Mathematical Linguistics No. 108, 2017, pp. 13–25. doi: 10.1515/pralin-2017-0005.

indicates that we might have a training problem since the model and the number of parameters are exactly the same in this scenario. We call this averaging averaged-best. On the other hand, building ensembles requires training several models which is time consuming, however, it is common to do that in NMT (Jean et al., 2015). Thus, an investigation is needed to discover whether either the model or the estimation of its parameters is weak.

In this work, we empirically investigate the most prominent first-order stochastic optimization methods to train an NMT system and exclusively investigate their behavior in NMT. We address three main concerns. a) translation performance, b) training stability and c) convergence speed. On one hand, how well, fast and stable different optimization algorithms are able to find appropriate local minima and on the other hand, how a combination of them can solve these aspects of training problems. The results show that applying these combinations leads to faster convergence, translation performance boost and more regularized behavior compared to running an optimizer alone. In this work, we follow the same standalone attention-based NMT proposed by (Bahdanau et al., 2015) but with different optimization schemes.

1.1. Related Work

There are many works in which researchers interpret the characteristics of different optimization techniques theoretically (Goodfellow et al., 2016; Ruder, 2016). Moreover, some other works try to show the performance of optimizers in the investigation of loss surface for image classification task such as (Im et al., 2016). (Zeyer et al., 2017) investigate various optimization methods for acoustic modeling empirically. (Dozat, 2015) compares different optimizers in language modeling. Furthermore, (Britz et al., 2017) study a massive analysis of NMT hyperparameters aiming for better optimization being robust to the hyperparameter variations.

To the best of our knowledge, there is no work comparing different optimization algorithms for NMT. Most of the works in this area focus on the modeling problem and rely on Adadelta used in vanilla NMT (Cho et al., 2014; Bahdanau et al., 2015).

Recently, (Wu et al., 2016) utilized the combination of Adam and a simple Stochastic Gradient Descend (SGD) learning algorithm. They run Adam for a fixed number of iterations after which they switch to SGD to slow down the training phase. Furthermore, (Farajian et al., 2016) optimize the networks with both Adagrad and Adadelta and show that using Adagrad leads to faster convergence and better performance.

2. Neural Machine Translation

Given a source $\mathbf{f} = f_1^I$ and a target $\boldsymbol{e} = e_1^I$ sequence, NMT (Sutskever et al., 2014; Bahdanau et al., 2015) models the conditional probability of target words given the source sequence. The NMT training objective function is to minimize the cross-entropy over the S training samples $\left\{\left\langle \mathbf{f}^{(s)}, \boldsymbol{e}^{(s)} \right\rangle\right\}_{s=1}^{S}$ which is defined as below:

$$J(\theta) = \sum_{s=1}^{S} \sum_{i=1}^{I^{(s)}} \log p(e_i{}^{(s)} | e_{<i}{}^{(s)}, f^{(s)}; \theta) \quad (1)$$

Since computing the objective function for the whole training data is expensive, we randomly select a small number of samples and take the average over them. This is so-called mini-batch training, resulting in mini-batch gradient calculations. We leave out the corresponding notations for mini-batches for simplicity.

3. Optimization

Fast convergence and robustness against stochasticity are important aspects desired in an optimizer so that it finds the global optimum. In practice, local optima can be sufficient and gradient-based techniques are able to find it (Goodfellow et al., 2016).

3.1. Stochastic Gradient Descent

The commonly used gradient-based algorithm in neural network is Stochastic Gradient Descent (SGD) (Robbins and Monro, 1951) which updates a set of parameters, θ, as shown in Algorithm 1. η is called the learning rate, determining how large the update is and g_t represents the gradient of cost function J. Then, the parameters in the direction of the gradients are updated. For simplicity, we refer to g_{θ_t} as g_t. Through this paper, we use the term SGD to state the simple SGD defined here.

Algorithm 1 : Stochastic Gradient Descent (SGD)

1: $g_t \leftarrow \nabla_{\theta_t} J(\theta_t)$
2: $\theta_{t+1} \leftarrow \theta_t - \eta g_t$

SGD usually uses scheduling-based step size selection and the learning rate is one of the important hyperparameters of training that should be carefully tuned. Unlike simple SGD, a number of methods have been introduced to adapt the separate learning rate for each parameter, called adaptive optimizer. It is still necessary to choose proper hyperparameters for these methods, but less sensitive. The most prominent first-order gradient-based optimizers are Adagrad, RmsProp, Adadelta and Adam that are briefly discussed in the following.

3.2. Adagrad

Adagrad (Duchi et al., 2011) is a gradient-based method in which the shared global learning rate η is divided by the l_2-norm of all previous gradients, n_t, as seen in Algorithm 2, line 3. Hence, it introduces different learning rates for every parameter

at each time step, so that larger gradients have smaller learning rates and vice versa. This property helps to perform larger updates for the dimensions with infrequent changes and smaller updates for those that have already large changes. On the other hand, n_t in the denominator is a positive growing value which might aggressively shrink the learning rate. ϵ is the stabilizing numerical constant.

Algorithm 2 : Adagrad

$g_t \leftarrow \nabla_{\theta_t} J(\theta_t)$
$n_t \leftarrow n_{t-1} + g_t^2$
$\theta_{t+1} \leftarrow \theta_t - \frac{\eta}{\sqrt{n_t}+\epsilon} g_t$

Algorithm 3 : RmsProp

$g_t \leftarrow \nabla_{\theta_t} J(\theta_t)$
$n_t \leftarrow \nu n_{t-1} + (1-\nu) g_t^2$
$\theta_{t+1} \leftarrow \theta_t - \frac{\eta}{\sqrt{n_t}+\epsilon} g_t$

3.3. RmsProp

Instead of storing all the past squared gradients from the beginning of the training, one can restrict a window over the recent gradients to acquire local information. An efficient way to do it is RmsProp in which instead of using the sum of squared gradients, a decaying weight of squared gradients is applied (Algorithm 3) (Hinton et al., 2012).

Algorithm 4 : Adadelta

$g_t \leftarrow \nabla_{\theta_t} J(\theta_t)$
$n_t \leftarrow \nu n_{t-1} + (1-\nu) g_t^2$
$r(n_t) \leftarrow \sqrt{n_t + \epsilon}$
$\Delta\theta_t \leftarrow \frac{-\eta}{r(n_t)} g_t$
$s_t \leftarrow \nu s_{t-1} + (1-\nu)\Delta\theta_t^2$
$r(s_{t-1}) \leftarrow \sqrt{s_{t-1} + \epsilon}$
$\theta_{t+1} \leftarrow \theta_t - \frac{r(s_{t-1})}{r(n_t)} g_t$

Algorithm 5 : Adam

$g_t \leftarrow \nabla_{\theta_t} J(\theta_t)$
$n_t \leftarrow \nu n_{t-1} + (1-\nu) g_t^2$
$\hat{n}_t \leftarrow \frac{n_t}{1-\nu^t}$
$m_t \leftarrow \mu m_{t-1} + (1-\mu) g_t$
$\hat{m}_t \leftarrow \frac{m_t}{1-\mu^t}$
$\theta_{t+1} \leftarrow \theta_t - \frac{\eta}{\sqrt{\hat{n}_t}+\epsilon} \hat{m}_t$

3.4. Adadelta

Similar to RmsProp, Adadelta takes the decaying mean of the past squared gradients. As shown in Algorithm 4, n_t accumulates this quantity, and its square root becomes the r_t of past squared gradients up to the time t. The obtained parameter update is stored in $\Delta\theta_t$. Then the squared parameter updates, s_t, is accumulated in a decaying manner to compute the final update (Zeiler, 2012). Since $\Delta\theta_t$ is unknown for the current time step, its value is estimated by the r_t of parameter updates up to the last time step. Eventually, the update rule requires no default learning rate to set.

3.5. Adam

Adaptive Moment Estimation (Adam) is another gradient-based approach that has been proposed recently (Kingma and Ba, 2014). It not only accumulates the decaying average of the past squared gradients $\mathbf{n}_t$, like RmsProp and Adadelta, but also stores a decaying mean of past gradients $\mathbf{m}_t$. There are two terms which can be considered as the first and second moments. In Algorithm 5, $\hat{\mathbf{m}}_t$ and $\hat{\mathbf{n}}_t$ are the bias corrected terms for instability against zero initialization.

3.6. Combination of Optimizers

Because the learning trajectory significantly affects training process, it is required to control the learning rate. Many research attempts show that simple SGD is able to find a minimum, but it might take long and it relies on the initial learning rate (Goodfellow et al., 2016, Chapter 8). Therefore, at the beginning, a fast convergence to the zone in which local minima located is desired. Then, by reducing the decay rate, the model has better opportunity to find the best critical point within that area. To do so, one can combine different optimization algorithms to take advantage of methods which accelerate the training and afterwards switch to the techniques with more control on the learning rate (Wu et al., 2016). Here, we combine adaptive optimizers with the simple SGD not only to regulate the learning phase, but also to accelerate the whole process. We start the training with any of the five considered optimizers, then run the variants of reducing the learning rate. These variations are:

1. Fixed-SGD: means the training is carried on by the simple SGD algorithm with a constant learning rate. Thus, it is easy to apply and there is no need to have any schedules or thresholds in advance. Here, we use a learning rate of 0.01.
2. Annealing: refers to the scheduling scheme in which the learning rate of the associated optimizer is decreased based on a pre-defined schedule between epochs. We use a schedule in that the learning rate is halved after every sub-epoch.

4. Experiments

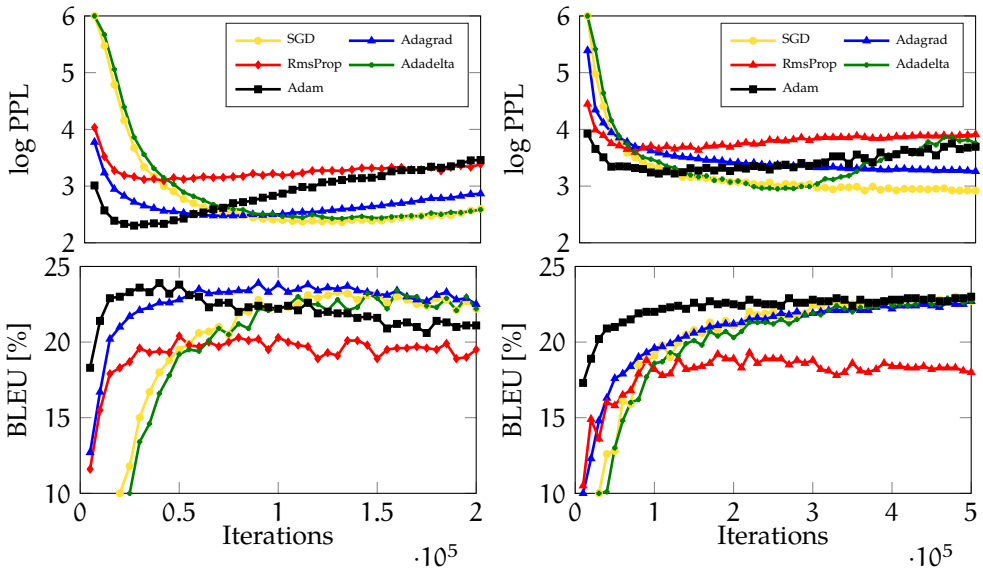
We have carried out the experiments on two translation tasks. The WMT 2016 En→Ro and WMT 2015 De→En. All experiments use the bilingual data, without any monolingual data. All systems follow the architecture by (Bahdanau et al., 2015). We use an implementation based on Blocks (Merriënboer et al., 2015)[1] which is a framework on top of Theano (Bastien et al., 2012). To deal with OOVs, we use the joint-BPE approach (Sennrich et al., 2016) to have a sequence of subwords in both the source and the target sides. In both tasks, the number of joint-BPE operations is 20K. All words are projected into a 620-dimensional embedding space. Both encoder and decoder are equipped with LSTMs with peephole connections with 1000 cells. We shuffle the training samples once before training and use mini-batches of 50 sentence pairs

[1] https://github.com/mila-udem/blocks-examples

and remove sentences longer than 75 subwords. Decoding is performed using beam search with a beam size of 12. The models are trained with different optimization schemes (see § 3) using the same architecture, the same number of parameters and all are identically initialized by the same random seed. The total number of parameters are 73M and 76M for En→Ro and De→En respectively. The systems are evaluated using case-sensitive BLEU and case-sensitive TER (Papineni et al., 2002; Snover et al., 2006) computed by MultEval (Clark et al., 2011).

For WMT 2016 En→Ro, the training data consists of 604K pairs of bilingual sentences with 16.8M English and 17.7M Romanian subwords. Validation is performed on 1000 sentences of the `newsdev2016` corpus. We stop training after 200K iterations and evaluate them every 5K. One iteration is one mini-batch. The `newstest16` corpus consisting of 1999 sentences is used as our test set.

For WMT 2015 De→En translation task, the bilingual training data includes 4.2M sentence pairs. The data set is composed of 133M German and 125M English words. The concatenation of `newstest2011` and `newstest2012` is used as our validation set named (`newsdev11+12`) resulting to 5984 sentences. We evaluate and save the models every 10K and stop training after 500K iterations. `newstest2014` and `newstest2015` are used as the test set including 3003 and 2169 samples respectively. For adaptive-based algorithms, which adapt the learning rate during training, we use the default hyperparameters proposed by the original publications (see Algorithms 2-5).



(a) En→Ro *(b) De→En*

Figure 1: log PPL and BLEU score of all optimizers on validation sets.

5. Analysis

Figure 1 shows the behavior of five optimizers for both En→Ro and De→En tasks in terms of BLEU and perplexity (PPL). As it is shown, Adam and Adagrad are faster at the beginning and converge to a relatively good area in the parameter space in terms of the cost function. At the beginning of the training, Adam outperforms the other methods in terms of both BLEU and perplexity. As it is seen in the Figure 1a, Adam reaches 23.9% in BLEU after only 40K iterations for En→Ro and 22.8% BLEU after 220K iterations on De→En compared to the others (see Fig. 1b). Its aggressive movement diverts from the local minima afterward.

In Adagrad, the denominator accumulates the sum of square of past gradients over training iterations leading to a significantly small learning step which slows down the training phase. Although RmsProp moves fast initially, it converges to a worse point compared to the other optimizers. We leave out RmsProp in the rest of our experiments since it has not shown promising results. Adadelta and simple SGD (with a constant learning rate) have a similar smooth pattern. Both have a moderate behavior for the first iterations and move slowly towards saturation. The same patterns for all of the optimizers in terms of log PPL can be seen in Figure 1. Again Adam converges to a proper point faster than the others.

In our experiments, we continue the training of the best model using different combinations described in Section 3.6. We monitor the perplexity and BLEU score on the validation set during training. We pick the best model among all based on BLEU to continue training the network by one of the explained combinations. In this case, our intuition is that we have already reached an appropriate region in the parameter space and it is a good time to slow down the training. By means of finer search, the optimizer has better chance not to skip a good local minima.

Figures 2 and 3 show the BLEU on the validation sets for En→Ro and De→En translation tasks respectively using these combinations. For example, the network is firstly trained by Adam and followed by Fixed-SGD, Annealing-SGD and Annealing-Adam (Fig. 2d and 3d). As the name suggests in Fixed-SGD, we continue training with the simple SGD and a fixed learning rate is used. While in the annealing-based strategies, the network continues having an annealing schedule. The difference between the two last variants is that in the former (Annealing-SGD), the learning rate of the simple SGD is reduced, whereas in the latter (Annealing-Adam) the learning rate of the adaptive-based optimizer, Adam, is decreased.

One can find the detailed results of the individual configuration on the validation set for each task in Table 1. In this Table, on one hand, the performance of each strategy has been compared with its base optimizer (e.g. line 12 compared to line 15) and on the other hand, the overall analogy among different groups has been shown. For each group, the improvement over the base optimizer has been written in the parenthesis and the best performance is marked by (†). As depicted, for all of the optimizers, applying these combinations improves both BLEU and TER. We also observed the same performance boost in terms of PPL. The smallest boost is associated with the

Adagrad optimizer on De→En. We speculate that the learning rate of Adagrad is already too small and annealing it makes the entire term much smaller leading to the slow training. Therefore, it is not possible to find better optima in a proper time.

Overall analogy states that Adam followed by Annealing-Adam gives the best results on `newsdev11+12` for De→En up to 25.4% in BLEU and 56.7% in TER. Moreover, the boost of this approach is the same as the Adam plus the other configurations on `newsdev16` for En→Ro and obtains 26.2% in BLEU and 55.9% in TER. We believe that the small fluctuations in BLEU and TER scores might be the noise.

Since the performance of the third strategy (results listed in the line 3, 7, 11 and 15) is as good as or better than the rest, we choose it to narrow down the results and verify the results on the test sets. In this case, for each parameter, we have an individual learning rate.

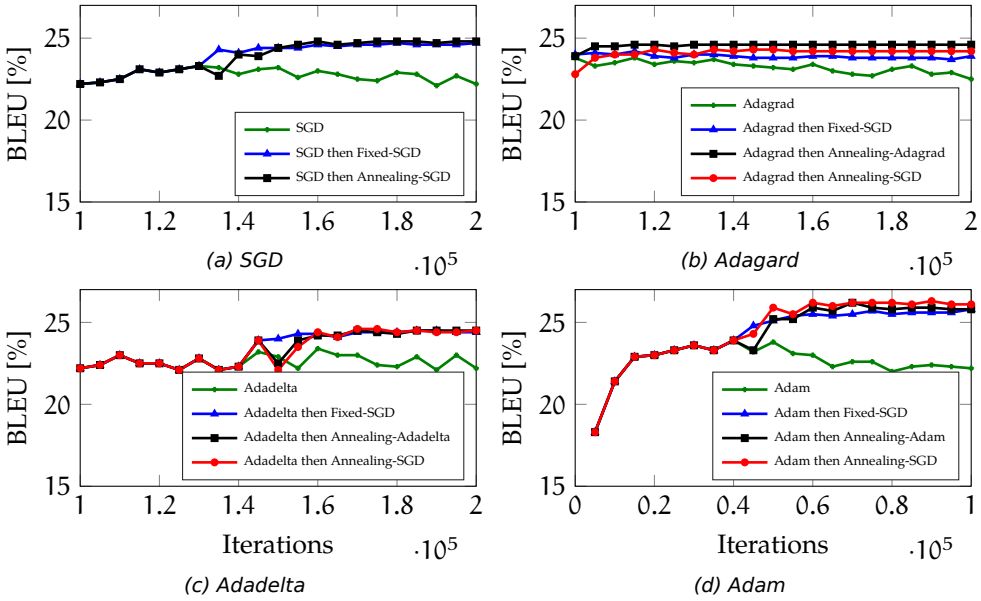
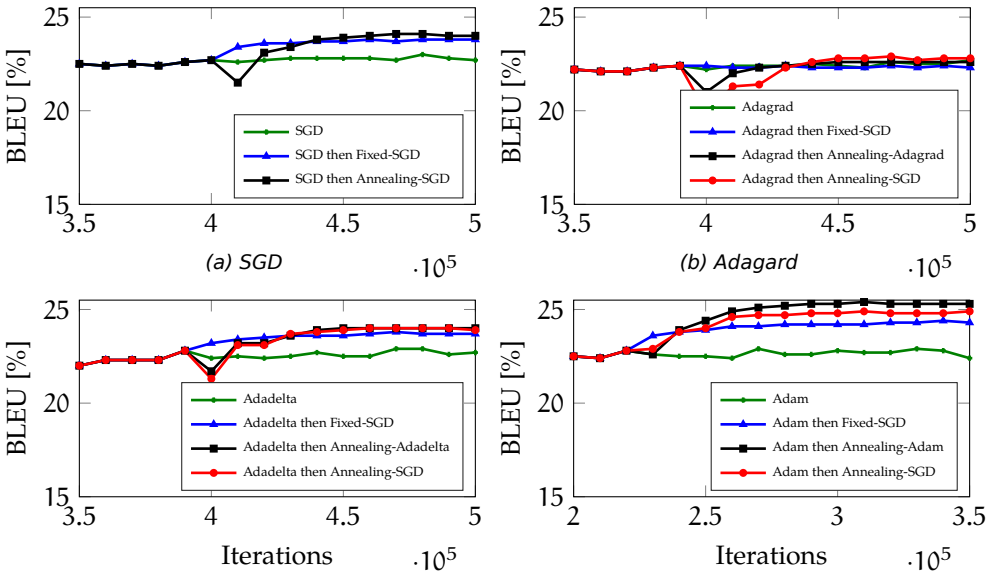


Figure 2: BLEU of optimizers followed by the combinations on the val. set for En→Ro.



(a) SGD *(b) Adagard* *(c) Adadelta* *(d) Adam*

Figure 3: BLEU of optimizers followed by the combinations on the val. set for De→En. The representation of x-axis of Adam is different as it is faster.

		En→Ro		De→En	
	Optimizer	newsdev16		newsdev11+12	
		BLEU	TER	BLEU	TER
1	SGD	23.3	59.5	22.8	59.5
2	+ Fixed-SGD	24.7 (+1.4)	57.0 (-2.5)	23.8 (+1.0)	58.4 (-1.1)
3	+ Annealing-SGD	24.8 (+1.5)	57.0 (-2.5)	24.1 (+1.3)	58.1 (-1.4)
4	Adagrad	23.9	58.1	22.6	60.0
5	+ Fixed-SGD	24.2 (+0.3)	57.7 (-0.4)	22.4 (-0.2)	60.3 (+0.3)
6	+ Annealing-SGD	24.3 (+0.4)	57.4 (-0.7)	22.9 (+0.3)	59.7 (-0.3)
7	+ Annealing-Adagrad	24.6 (+0.7)	57.0 (-1.1)	22.6 (0.0)	59.9 (-0.1)
8	Adadelta	23.2	59.1	22.9	59.8
9	+ Fixed-SGD	24.5 (+1.3)	57.3 (-1.8)	23.8 (+0.9)	58.5 (-1.3)
10	+ Annealing-SGD	24.6 (+1.4)	57.6 (-1.5)	24.0 (+1.1)	58.2 (-1.6)
11	+ Annealing-Adadelta	24.6 (+1.4)	57.5 (-1.6)	24.0 (+1.1)	58.4 (-1.4)
12	Adam	23.9	58.2	23.0	59.4
13	+ Fixed-SGD	26.2 (+2.3)	55.6 (-2.6)	24.5 (+1.5)	57.7 (-1.7)
14	+ Annealing-SGD	26.3 (+2.4)†	55.8 (-2.4)†	24.9 (+1.9)	57.4 (-2.0)
15	+ Annealing-Adam	26.2 (+2.3)	55.9 (-2.3)	25.4 (+2.4)†	56.7 (-2.7)†

Table 1: Results in BLEU[%] and TER[%] on val. sets. † shows the best results.

5.1. Translation Quality

Table 2 lists the results of the test sets for En→Ro and De→En tasks. We also report the results of the Annealing method for each optimizers. In addition to the performance of the best model, the results of the averaged-best model which is the average of the four best training points which is also a single model is shown. Clearly, the Adam followed by Annealing-Adam outperforms the rest of combinations. An interesting point to be highlighted is that the averaged-best leads to improvements as good as the annealing strategy except for one case in the table. On `newstest16` En→Ro[2], the averaged-best of Adam outperforms the Annealing-Adam. If one does not follow any schedule scheme, he can easily run the pure Adam, save the intermediate points and average the weights. We observed the same pattern of improvements for TER on both tasks. As a summary, we reach the conclusion that the model has the opportunity to find the better critical point within that located area, if the learning step is reduced. Applying these variations differing in the handling of the learning rate can be helpful to focus more on an area containing the local minima. We conclude that shrinking the learning steps might lead to a finer search and prevent stumbling over a local minimum. By comparing the performance of different optimization approaches, we showed that Adam followed by Annealing-Adam gains the best performance.

		En→Ro		De→En			
		Best	Averaged-best	Best		Averaged-best	
	Optimizer	`newstest16`		`newstest14`	`newstest15`	`newstest14`	`newstest15`
1	SGD	20.3	22.5	25.2	26.1	26.7	27.4
2	+ Annealing-SGD	22.1	21.9	26.9	27.4	26.9	27.2
3	Adagrad	21.6	21.7	24.8	26.2	24.9	26.0
4	+ Annealing-Adagrad	21.9	21.9	24.6	25.5	24.6	25.5
5	Adadelta	20.5	22.2	25.2	25.6	26.6	27.4
6	+ Annealing-Adadelta	22.0	22.0	26.4	27.6	26.5	27.4
7	Adam	21.4	24.6	25.0	25.7	28.0	28.9
8	+ Annealing-Adam	23.0†	23.1	28.1†	29.0†	28.2	29.0

Table 2: Results measured in BLEU[%] for best and averaged-best models on test sets. † shows the best performance for the Best models.

5.2. Training Stability

As shown in Table 2, using one of these configurations to slow down the learning phase narrows the gap between averaged-best and the best model. For example for En→Ro, averaged-best model gains 1.7% in BLEU using Adadelta (line 5) while applying Annealing-Adadelta has already covered this offset and it does not get more boost from averaging (line 6). Moreover, this gap has been compensated by decreasing the learning steps. This property holds for all of the cases in Table 2. We conclude that pure Adam training is less regularized, therefore the model is allowed to navigate

[2]Note that, the best performing En→Ro NMT system in the WMT 2016 shared task has been used the synthetic data which is not the case in our work.

varying areas in the parameter space. It stumbles on good cases. Thus it is beneficial to average the best cases. Whereas in the Adam+Annealing-Adam the parameter space navigated is more regularized, leading to less varieties.

5.3. Convergence Speed

The momentum in Adam optimizer regulating the directions causes a fast descent towards the minimum. By adding the previous updates into the current update, momentum enforces the updates in a particular direction. The convergence of Adam followed by Annealing-Adam obtained after 70K iterations for En→Ro shown in figure 2d as well as 310K iterations for De→En pictured in figure 3d. This results to 50% faster convergence in the training on average on both tasks. As the number of training samples for En→Ro is seven times smaller than those for De→En, the convergence of this task is faster.

6. Conclusion

We practically analyzed the performance of five common first-order gradient-based optimization methods in NMT which are either run alone or followed by the variations differing in the handling of the learning rate. We benefited from the methods accelerating the training at the beginning and then switched to the techniques with more control on the learning rate to find the better local minimum in parameter space. The quality of the models in terms of BLEU and TER scores as well as the convergence speed and robustness against stochasticity have been investigated on two WMT translation tasks. We concluded that in order to speed up the training and enhance the performance in terms of both BLEU and TER, one could apply Adam followed by Annealing-Adam. Experiments done on WMT 2016 En→Ro and WMT 2015 De→En show that the mentioned technique leads to 1.6% BLEU improvements on `newstest16` for En→Ro, and 3.1% BLEU on `newstest15` for De→En. Moreover, it results to faster convergence of 50% as well as the training stability. We showed that using Annealing-Adam compensates the offset between the best model and the averaged-best. We recommend that, if someone does not utilize the annealing scheme to reduce the learning rate, he should average the best training points to increase the translation performance. Similar to NMT, we hope that the proposed techniques would help other neural network training including non-sequential models.

Acknowledgements

The work reported in this paper results from two projects, SEQCLAS and QT21. SEQCLAS has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation program under grant agreement n° 694537. QT21 has received funding from the European Union's

Horizon 2020 research and innovation program under grant agreement nº 645452. The work reflects only the authors' views and neither the European Commission nor the European Research Council Executive Agency is responsible for any use that may be made of the information it contains. Tamer Alkhouli was partly funded by the 2016 Google PhD Fellowship for North America, Europe and the Middle East.

Bibliography

Bahdanau, Dzmitry, Kyunghyun Cho, and Yoshua Bengio. Neural Machine Translation by Jointly Learning to Align and Translate. *CoRR*, abs/1409.0473, 2015.

Bastien, Frédéric, Pascal Lamblin, Razvan Pascanu, James Bergstra, Ian J. Goodfellow, Arnaud Bergeron, Nicolas Bouchard, and Yoshua Bengio. Theano: new features and speed improvements. Deep Learning and Unsupervised Feature Learning NIPS 2012 Workshop, 2012.

Britz, Denny, Anna Goldie, Thang Luong, and Quoc Le. Massive Exploration of Neural Machine Translation Architectures. *arXiv preprint arXiv:1703.03906*, 2017.

Cho, Kyunghyun, Bart van Merrienboer, Dzmitry Bahdanau, and Yoshua Bengio. On the Properties of Neural Machine Translation: Encoder-Decoder Approaches. In *Proceedings of SSST@EMNLP 2014, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation, Qatar*, pages 103–111, 2014.

Clark, Jonathan H., Chris Dyer, Alon Lavie, and Noah A. Smith. Better Hypothesis Testing for Statistical Machine Translation: Controlling for Optimizer Instability. In *49th Annual Meeting of the Association for Computational Linguistics*, pages 176—181, USA, 2011.

Dozat, Timothy. Incorporating Nesterov momentum into Adam. Technical report, 2015.

Duchi, John C., Elad Hazan, and Yoram Singer. Adaptive Subgradient Methods for Online Learning and Stochastic Optimization. *Journal of Machine Learning Research*, 12:2121–2159, 2011.

Farajian, M Amin, Rajen Chatterjee, Costanza Conforti, Shahab Jalalvand, Vevake Balaraman, Mattia A Di Gangi, Duygu Ataman, Marco Turchi, Matteo Negri, and Marcello Federico. FBK's Neural Machine Translation Systems for IWSLT 2016. In *Proceedings of the ninth International Workshop on Spoken Language Translation, USA*, 2016.

Goodfellow, Ian, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016.

Hinton, Geoffrey, N Srivastava, and Kevin Swersky. Lecture 6a overview of mini–batch gradient descent. *Coursera Lecture slides https://class.coursera.org/neuralnets-2012-001/*, 2012.

Im, Daniel Jiwoong, Michael Tao, and Kristin Branson. An Empirical Analysis of Deep Network Loss Surfaces. *CoRR*, abs/1612.04010, 2016.

Jean, Sébastien, Orhan Firat, Kyunghyun Cho, Roland Memisevic, and Yoshua Bengio. Montreal Neural Machine Translation Systems for WMT'15. In *Proceedings of the Tenth Workshop on Statistical Machine Translation, WMT 2015, Portugal*, pages 134–140, 2015.

Junczys-Dowmunt, Marcin, Tomasz Dwojak, and Rico Sennrich. The AMU-UEDIN Submission to the WMT16 News Translation Task: Attention-based NMT Models as Feature Functions in Phrase-based SMT. In *Proceedings of the First Conference on Machine Translation, WMT 2016, Germany*, pages 319–325, 2016.

Kingma, Diederik P. and Jimmy Ba. Adam: A Method for Stochastic Optimization. *CoRR*, abs/1412.6980, 2014.

Merriënboer, Bart, Dzmitry Bahdanau, Vincent Dumoulin, Dmitriy Serdyuk, David Warde-Farley, Jan Chorowski, and Yoshua Bengio. Blocks and Fuel: Frameworks for deep learning. 2015.

Papineni, Kishore, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a Method for Automatic Evaluation of Machine Translation. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 311–318, USA, 2002.

Robbins, Herbert and Sutton Monro. A stochastic approximation method. *The annals of mathematical statistics*, pages 400–407, 1951.

Ruder, Sebastian. An overview of gradient descent optimization algorithms. *CoRR*, abs/1609.04747, 2016.

Sennrich, Rico, Barry Haddow, and Alexandra Birch. Neural Machine Translation of Rare Words with Subword Units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, Germany*, 2016.

Snover, Matthew, Bonnie Dorr, Richard Schwartz, Linnea Micciulla, and John Makhoul. A Study of Translation Edit Rate with Targeted Human Annotation. In *Proceedings of the 7th Conference of the Association for Machine Translation in the Americas*, pages 223–231, USA, 2006.

Sutskever, Ilya, Oriol Vinyals, and Quoc V. Le. Sequence to Sequence Learning with Neural Networks. In *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, Canada*, pages 3104–3112, 2014.

Wu, Yonghui, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus, et al. Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation. *CoRR*, abs/1609.08144, 2016.

Zeiler, Matthew D. ADADELTA: An Adaptive Learning Rate Method. *CoRR*, abs/1212.5701, 2012.

Zeyer, Albert, Patrick Doetsch, Paul Voigtlaender, Ralf Schlüter, and Hermann Ney. A Comprehensive Study of Deep Bidirectional LSTM RNNs for Acoustic Modeling in Speech Recognition. *CoRR*, abs/1606.06871, 2017.

Address for correspondence:
Parnia Bahar
bahar@i6.informatik.rwth-aachen.de
Human Language Technology and Pattern Recognition Group,
RWTH Aachen University,
Ahornstraße 55, 52074 Aachen, Germany

PBML

The Prague Bulletin of Mathematical Linguistics
NUMBER 108 JUNE 2017 27–36

Generating Alignments Using Target Foresight in Attention-Based Neural Machine Translation

Jan-Thorsten Peter, Arne Nix, Hermann Ney

Human Language Technology and Pattern Recognition Group
RWTH Aachen University, Ahornstr. 55, 52056 Aachen, Germany

Abstract

Neural machine translation (NMT) has shown large improvements in recent years. The currently most successful approach in this area relies on the attention mechanism, which is often interpreted as an alignment, even though it is computed without explicit knowledge of the target word. This limitation is the most likely reason that the quality of attention-based alignments is inferior to the quality of traditional alignment methods. Guided alignment training has shown that alignments are still capable of improving translation quality. In this work, we propose an extension of the attention-based NMT model that introduces target information into the attention mechanism to produce high-quality alignments. In comparison to the conventional attention-based alignments, our model halves the AER with an absolute improvement of 19.1% AER. Compared to GIZA++ it shows an absolute improvement of 2.0% AER.

1. Introduction

The field of machine translation has seen a drastic shift in recent years since it has been demonstrated that end-to-end neural machine translation (NMT) models (Bahdanau et al., 2015) are able to outperform traditional phrase-based systems on numerous tasks. A key component of the approach introduced by Bahdanau et al. is the attention mechanism, which has been subject to a lot of research (Luong et al., 2015; Tu et al., 2016; Mi et al., 2016a; Sankaran et al., 2016; Feng et al., 2016; Cohn et al., 2016). The attention mechanism produces a distribution over the source sentence for every decoding step. This distribution is often interpreted as a soft alignment between the source and target sentence. It has been shown that incorporating alignment in-

 Corresponding author: peter@cs.rwth-aachen.de
Cite as: Jan-Thorsten Peter, Arne Nix, Hermann Ney. Generating Alignments Using Target Foresight in Attention-Based Neural Machine Translation. The Prague Bulletin of Mathematical Linguistics No. 108, 2017, pp. 27–36.
doi: 10.1515/pralin-2017-0006.

formation during training as an additional objective function can improve the overall performance of the system (Chen et al., 2016). This indicates that the alignment problem is still relevant.

The relation between attention and alignments provides the motivation for this work, which aims at using the attention-based NMT approach to generate word alignments. However, the attention mechanism has a disadvantage compared to regular word alignment methods. While the word alignment is computed including the knowledge of the whole source and target sentence, the neural network knows only previously seen words on the target side. To remove this disadvantage, we extend the standard attention computation by introducing knowledge of the target word to which we want to align.

2. Related Work

Based on the NMT approach by Bahdanau et al. (2015) researchers have tried to improve the translation quality by modifying the attention mechanism. Most methods add various features to the attention computation (Tu et al., 2016; Mi et al., 2016a; Sankaran et al., 2016; Feng et al., 2016; Cohn et al., 2016), while others attempt to change the attention mechanism itself (Zhang et al., 2016). External alignments have been utilized to teach the network to mimic them by adding them to the objective function during training (Chen et al., 2016; Mi et al., 2016b).

Even though most of these approaches interpret the attention as a soft alignment, to the best of our knowledge, there have been only four publications that empirically measure the impact of their approach on the alignment quality (Tu et al., 2016; Mi et al., 2016a,b; Sankaran et al., 2016). These investigations use the SAER (Tu et al., 2016), AER (Och and Ney, 2003) and F_1 metrics to measure the alignment quality. All authors noticed an improvement in alignment quality by applying their extensions to the attention mechanism, but as Mi et al. (2016b) report, there is still a significant qualitative difference to state-of-the-art alignments.

A method to create alignments using posterior regularization was presented by Ganchev et al. (2010) and Tamura et al. (2014) which used a special purpose recurrent neural network to create alignments.

3. Neural Machine Translation

The neural machine translation approach, as introduced by Bahdanau et al. (2015), is composed of three main components: The encoder, the attention mechanism, and the decoder (Figure 1). The encoder is a bidirectional recurrent neural network (RNN) which is applied to the input sentence f_1^J to produce the source representation h_1^J, where J is the sentence length. In each decoder step $i = 1, \ldots, I$ the encoder state for each source position $j = 1, \ldots, J$ is used to compute the attention energies $\tilde{\alpha}_{ij}$. For this a single hidden layer with weights W_α, U_α and an additional transformation vector

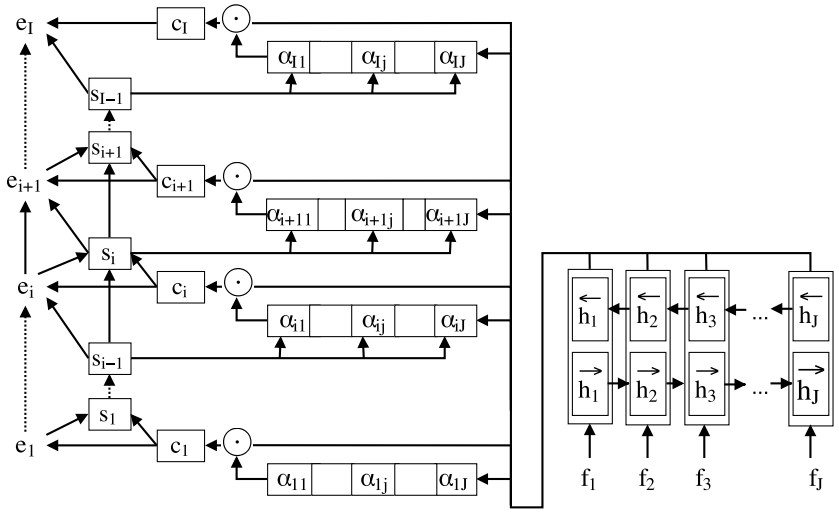


Figure 1. The unmodified attention-based NMT model (Bahdanau et al., 2015)

v_a is applied to the previous decoder state s_{i-1} and the relevant source representation h_j.

$$\tilde{\alpha}_{ij} := v_a^T \tanh(W_a s_{i-1} + U_a h_j) \tag{1}$$

The energies are converted into the attention weights α_{ij} by normalization with a softmax function over all $j = 1, \ldots, J$. These weights are used to compute the context vector c_i as a weighted sum of the encoder representations h_1^J.

This context vector c_i is handed over to the decoder which generates the output word e_i while taking the previously generated output e_{i-1}, the old decoder state s_{i-1} and the context vector c_i as inputs. At the end of each decoding step, the hidden decoder state s_i is updated w.r.t. the previous hidden state s_{i-1}, the context vector c_i and the generated output word e_i.

An extension to the standard training procedure for NMT models is introduced by guided alignment training (Chen et al., 2016; Mi et al., 2016b). This approach is designed to benefit from state-of-the-art alignments by defining an additional cost function that gives feedback explicitly to the components of the attention mechanism. This second loss function is computed for a set of N training samples as the cross-entropy between the soft alignment α_{ij} extracted from the attention mechanism and a given target alignment A_{ij}, provided by e.g.GIZA++ (Och and Ney, 2003):

$$\mathcal{L}_{al}(A, \alpha) := -\frac{1}{N} \sum_n \sum_{i=1}^{I^{(n)}} \sum_{j=1}^{J^{(n)}} A_{ij}^{(n)} \log \alpha_{ij}^{(n)} \tag{2}$$

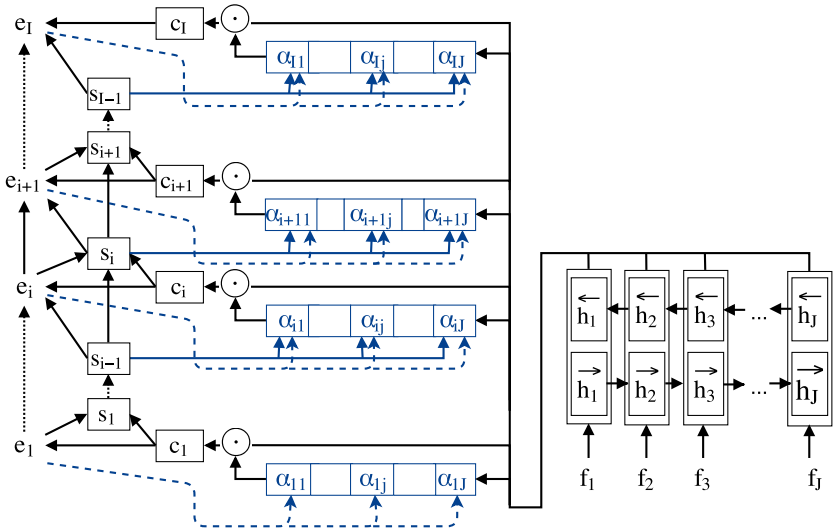


Figure 2. Attention-based NMT with target foresight, the dotted lines show how the current target word is feedback to the alignment computation.

To integrate this additional error measure into the traditional training process a new network loss function is defined as the weighted sum of the standard decoder cost function and the introduced alignment cost function.

4. Target Foresight

Since the introduction of the IBM models (Brown et al., 1993), alignments have always been important for statistical machine translation. And even though the attention mechanism (Bahdanau et al., 2015) does not explicitly generate an alignment, approaches like guided alignment training (Chen et al., 2016) and the analysis by Tu et al. (2016) indicate that the information encoded in the attention weights is related to an alignment from source to target side.

The aim of this work is to explore the alignment capabilities of the attention-based NMT model and to create alignments that are optimized for NMT. The latter is important since the attention mechanism does not assign weights to the source words, but to the encoder representation that is generated from these words. This representation may consequently encode information about neighboring words in the source sentence.

Nevertheless, we interpret the attention weights as a soft alignment for the remaining sections of this work and try to improve the alignment quality compared to the standard attention mechanism. We follow the example of traditional alignment

methods and use the knowledge of the target reference sentence $\hat{e}_1^{\hat{I}}$ to improve the alignment quality of the attention. Therefore, we introduce the target word of the current decoding step $\hat{e}_i$ as additional input for the attention energy computation:

$$\tilde{\alpha}_{ij} = v_a^T \tanh(W_a s_{i-1} + U_a h_j + V_a \hat{e}_i). \quad (3)$$

We refer to this approach as *target foresight (TF)*, since the network is allowed to use the foresight of the target word $\hat{e}_i$ to determine the corresponding source position that should be aligned to $\hat{e}_i$. Figure 2 shows the additional connection added to the NMT model.

To further investigate the target foresight approach, we propose three different methods to be applied during training. First we add random noise to the value of $\tilde{\alpha}_{ij}$, which is supposed to prevent the encoding of target-word information in the attention weights. The second approach is to freeze the values of all weight matrices except for the attention parameters in the update steps of the training. The last approach is to train target foresight using guided alignment training (Chen et al., 2016; Mi et al., 2016b). This approach works by enforcing the network not to diverge too far from a given alignment. It allows however to chose a different alignment point if the improvement in the translation cost is large enough.

5. Experiments

To evaluate the effectiveness of our approach we compare it to GIZA++ (Och and Ney, 2003), the BerkeleyAligner[1], `fast_align` (Dyer et al., 2013), and an unmodified attention-based model.

5.1. Setup

The translation models we use for all experiments in this work are based on the attention-based NMT approach by Bahdanau et al. (2015). We use a word-embedding size of 620 for the projection layer and a 30K shortlist of the most frequent words. The decoder and both directed RNNs of the bi-directional encoder are implemented as gated recurrent units. These RNNs as well as the attention layer have an internal dimension of 1000 nodes. For decoding, we use a beam-size of 12. Our implementation is based on the Blocks framework (Van Merriënboer et al., 2015) and the deep-learning library Theano (Bergstra et al., 2010).

To evaluate the alignment quality of our models, we use a set of 504 bilingual sentence pairs that were extracted from the Europarl (Koehn, 2005) German-to-English task and manually aligned by human annotators. We use this test set to evaluate the alignment quality on AER (Och and Ney, 2003) and SAER (Tu et al., 2016). To evaluate the soft alignment with AER, we convert it into a hard alignment by extracting the

[1] https://code.google.com/archive/p/berkeleyaligner

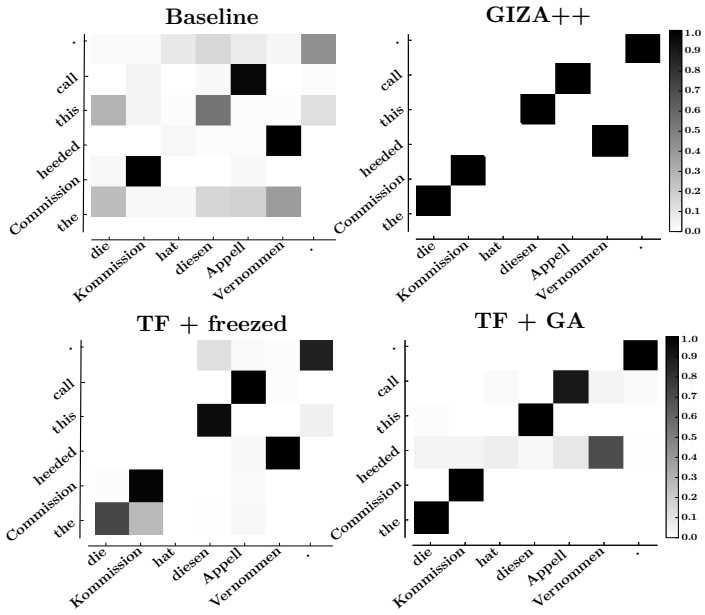


Figure 3. Attention weight matrices visualized in heat map form. Generated by the NMT Baseline, GIZA++, target foresight with freezed encoder and decoder parameters (TF + freezed) and target foresight with guided alignment training (TF + GA)

position with the largest alignment weight in both directions and merged them by applying Och's refined method (Och and Ney, 2003).

The network was trained on the Europarl corpus (Koehn, 2005) excluding the test set using AdaDelta (Zeiler, 2012) for learning rate adaption. Excluding the test data is done to evaluate the performance of the attention-based model on unseen data as it is the case when used for translation. It also shows that target foresight can easily be used to align unseen data without the need to retrain the model, while still outperforming traditional methods that have been trained including the test data. The training data consists of 1.2 million bilingual sentences of 32 and 34 million running words in German and English, respectively. The training is performed for 250K iterations with a batch-size of 40 and evaluated every 10K iterations. The development set of the IWSLT2013 German→English shared translation task[2] is used to select the best performing model which is then evaluated on the IWSLT2013 test as well as on the Europarl alignment test set.

[2] http://www.iwslt2013.org

Model	Alignment Test AER %	Alignment Test SAER %
fast_align	27.9	33.0
GIZA++	21.0	26.8
BerkeleyAligner	20.5	26.4
Attention-Based	38.1	63.6
+ Guided alignment	29.8	38.0
+ Target foresight with fixed en-/decoder	33.9	55.6
+ Target foresight with guided alignment	**19.0**	34.9
+ converted to hard alignment	**19.0**	**24.6**

Table 1. Comparison of target foresight with the pure attention-based approach (with and without guided alignment) and other alignment methods.

5.2. Results

Table 1 shows that GIZA++ creates a far better alignment than `fast_align` and that the BerkeleyAligner creates an even slightly better result. In comparison the attention mechanism produces an AER of 17.6% worse than the BerkeleyAligner.

Interpreting the attention of the attention-based approach as an alignment results in 38.1% AER. If we train the network using guided alignment, we can reduce the AER to 29.8%.

Using the target foresight directly to create an alignment produces no usable results. The network does not learn any meaningful alignment, but uses the attention weights to encode the target word $\hat{e}_i$. It is in nearly all cases able to reproduce the target word on the output layer, even though $\hat{e}_i$ is only given to compute the alignment. Furthermore the computed alignment has no meaningful correlation with the correct alignment. To prevent this behavior, we try to make it harder to encode the target word into the attention weights, by applying noise to the alignment weights and the outputs of the corresponding network components. We also tried to initialize the encoder and decoder using the weights from our trained baseline network. We omitted these numbers since unfortunately none of these techniques gave usable results and used the following methods instead.

Fixing the encoder and decoder weights of our baseline network and training the attention layer for just additional 2000 iterations results in an improvement of 4.2% AER and 8.0% SAER.

Pairing the guided alignment training with the target foresight training yields an AER of 19.0%. This is an improvment of 10.8% compared to only using guided alignment. Compared to the BerkeleyAligner it improved by 1.5% and by 2.0% compared to GIZA++. Note the latter two still perform better considering the SAER score.

An expalaination for this behavior is that the design of SAER makes it easier for systems with hard-alignments to perform well than system using soft-alignments.

To elaborate this point: Even if the soft and the hard-alignment create the correct alignment, the soft-alignment would most likely receive a lower score since is very unlikely that it predicts the correct point with 100% certainty. Most alignment points are predicted correctly by our systems in this task. This allows the hard-alignments to produce a perfect score at most points. The soft-alignments gives these points also the highest probability, but distributes its probability mass more evenly and recives therefore a lower score than the hard-alignment.

To solve this we compute the SAER score also using the hard-alignment that we use to compute the AER score. This gave us a corresponding SAER score that is 10.3% better than its soft equivalent. Using this comparison, the generated alignment outperforms all baseline methods on both evaluation metrics. We obtain an alignment which is superior to the baseline alignments and also to the standard guided alignment approach.

To verify that the obtained alignments can be used to improve the performance of an NMT, system we evaluate the guided alignment training on the IWSLT2013 task. We apply our NMT alignment model to produce a soft alignment for the Europarl training corpus and use it in guided alignment training. The resulting score of 18.8% BLEU was an improvement of 0.4% BLEU compared to a model trained using the GIZA++ alignment and 2.8% compared to the NMT baseline system. We also observe an improvement of 1.3% AER.

6. Conclusion

This work shows that attention-based models are capable of generating alignments that improve the BerkeleyAligner alignments by 1.5% AER. Using target foresight we are able to improve the AER by 19.1% compared to the baseline attention mechanism and outperform the GIZA++ alignments by 2.0% AER absolute and 9.5% relative using training with guided alignment. Additionally, we have shown that the new alignments can be used to improve the training of NMT models. The approach presented in this work shows also that it is possible to train one model and reuse it to align unseen data with a precision that outperforms the classical alignment methods.

Training the network to produce high quality alignments proves to be a hard task. The network seems to encode the knowledge of the target word in the attention weights and produces a non-usable alignment, but guided alignment training seems to counteract this effectively. In future work, we plan to find a way to achieve the same strong alignment without using guided alignment training.

Acknowledgements

The work reported in this paper results from two projects, SEQCLAS and QT21. SEQCLAS has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme under grant agreement nº 694537. QT21 has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement nº 645452. The work reflects only the authors' views and neither the European Commission nor the European Research Council Executive Agency are responsible for any use that may be made of the information it contains.

Bibliography

Bahdanau, Dzmitry, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. In *ICLR*, 2015.

Bergstra, James, Olivier Breuleux, Frédéric Bastien, Pascal Lamblin, Razvan Pascanu, Guillaume Desjardins, Joseph Turian, David Warde-Farley, and Yoshua Bengio. Theano: A CPU and GPU math compiler in Python. In *Proc. 9th Python in Science Conf*, pages 1–7, 2010.

Brown, Peter F., Vincent J. Della Pietra, Stephen A. Della Pietra, and Robert L. Mercer. The Mathematics of Statistical Machine Translation: Parameter Estimation. *Comput. Linguist.*, 19(2):263–311, June 1993. ISSN 0891-2017. URL `http://dl.acm.org/citation.cfm?id=972470.972474`.

Chen, Wenhu, Evgeny Matusov, Shahram Khadivi, and Jan-Thorsten Peter. Guided Alignment Training for Topic-Aware Neural Machine Translation. Austin, Texas, 2016. Association for Machine Translation in the Americas.

Cohn, Trevor, Cong Duy Vu Hoang, Ekaterina Vymolova, Kaisheng Yao, Chris Dyer, and Gholamreza Haffari. Incorporating structural alignment biases into an attentional neural translation model. *arXiv preprint arXiv:1601.01085*, 2016.

Dyer, Chris, Victor Chahuneau, and Noah A. Smith. A simple, fast, and effective reparametrization of IBM model 2. In *Proceedings of the NAACL 7th Workshop on Syntax, Semantics and Structure in Statistical Translation*, Atlanta, Georgia, USA, June 2013.

Feng, Shi, Shujie Liu, Mu Li, and Ming Zhou. Implicit Distortion and Fertility Models for Attention-based Encoder-Decoder NMT Model. *arXiv preprint arXiv:1601.03317*, 2016.

Ganchev, Kuzman, João Graça, Jennifer Gillenwater, and Ben Taskar. Posterior Regularization for Structured Latent Variable Models. *J. Mach. Learn. Res.*, 11:2001–2049, Aug. 2010. ISSN 1532-4435. URL `http://dl.acm.org/citation.cfm?id=1756006.1859918`.

Koehn, Philipp. Europarl: A parallel corpus for statistical machine translation. In *MT summit*, volume 5, pages 79–86, 2005.

Luong, Minh-Thang, Hieu Pham, and Christopher D. Manning. Effective Approaches to Attention-based Neural Machine Translation. In *Proceedings of the 2015 Conference on Em-*

pirical Methods in Natural Language Processing, pages 1412–1421, Lisbon, Portugal, September 2015. Association for Computational Linguistics. URL http://aclweb.org/anthology/D15-1166.

Mi, Haitao, Baskaran Sankaran, Zhiguo Wang, and Abe Ittycheriah. A Coverage Embedding Model for Neural Machine Translation. *arXiv preprint arXiv:1605.03148*, 2016a.

Mi, Haitao, Zhiguo Wang, and Abe Ittycheriah. Supervised Attentions for Neural Machine Translation. *arXiv preprint arXiv:1608.00112*, 2016b.

Och, Franz Josef and Hermann Ney. A Systematic Comparison of Various Statistical Alignment Models. *Computational Linguistics*, 29(1):19–51, 2003.

Sankaran, Baskaran, Haitao Mi, Yaser Al-Onaizan, and Abe Ittycheriah. Temporal Attention Model for Neural Machine Translation. *arXiv preprint arXiv:1608.02927*, 2016.

Tamura, Akihiro, Taro Watanabe, and Eiichiro Sumita. Recurrent Neural Networks for Word Alignment Model. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1470–1480, Baltimore, Maryland, June 2014. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/P14-1138.

Tu, Zhaopeng, Zhengdong Lu, Yang Liu, Xiaohua Liu, and Hang Li. Modeling Coverage for Neural Machine Translation. In *54th Annual Meeting of the Association for Computational Linguistics*, 2016.

Van Merriënboer, Bart, Dzmitry Bahdanau, Vincent Dumoulin, Dmitriy Serdyuk, David Warde-Farley, Jan Chorowski, and Yoshua Bengio. Blocks and fuel: Frameworks for deep learning. *arXiv preprint arXiv:1506.00619*, 2015.

Zeiler, Matthew D. ADADELTA: an adaptive learning rate method. *arXiv preprint arXiv:1212.5701*, 2012.

Zhang, Biao, Deyi Xiong, and Jinsong Su. Recurrent Neural Machine Translation. *arXiv preprint arXiv:1607.08725*, 2016.

Address for correspondence:
Jan-Thorsten Peter
peter@cs.rwth-aachen.de
Human Language Technology and Pattern Recognition Group
RWTH Aachen University
Ahornstr. 55, 52056 Aachen, Germany

PBML

The Prague Bulletin of Mathematical Linguistics
NUMBER 108 JUNE 2017 37–48

Convolutional over Recurrent Encoder for Neural Machine Translation

Praveen Dakwale, Christof Monz

Informatics Institute, University of Amsterdam

Abstract

Neural machine translation is a recently proposed approach which has shown competitive results to traditional MT approaches. Standard neural MT is an end-to-end neural network where the source sentence is encoded by a recurrent neural network (RNN) called encoder and the target words are predicted using another RNN known as decoder. Recently, various models have been proposed which replace the RNN encoder with a convolutional neural network (CNN). In this paper, we propose to augment the standard RNN encoder in NMT with additional convolutional layers in order to capture wider context in the encoder output. Experiments on English to German translation demonstrate that our approach can achieve significant improvements over a standard RNN-based baseline.

1. Introduction

Recently proposed neural machine translation (NMT) has shown competitive results over traditional MT approaches such as Phrase-based MT. The most successful of these neural network approaches is the encoder-decoder framework of Bahdanau et al. (2015) in which the source sentence is converted into a vector representation by a recurrent neural network (RNN) called encoder, then another RNN called decoder generates a target sentence word by word based on the source representation and target history. Besides Machine translation, RNNs have shown promising results in modelling various other NLP tasks such as language modelling (Mikolov et al., 2010) and text similarity (Mueller and Thyagarajan, 2016). The strength of using RNNs for language processing lies in their ability to recurrently maintain a history for large input sequences, thus capturing the long distance dependencies which is an important occurrence in natural language texts.

 Corresponding author: p.dakwale@uva.nl
Cite as: Praveen Dakwale, Christof Monz. Convolutional over Recurrent Encoder for Neural Machine Translation. The Prague Bulletin of Mathematical Linguistics No. 108, 2017, pp. 37–48. doi: 10.1515/pralin-2017-0007.

Although, modelling sequences using the recurrence property is important for most NLP tasks, there is a critical limitation in relying solely on the strengths of the RNN. In an RNN, at each timestep the encoder output is a global representation in which the information about the current word and the previous history are represented compositely. Although RNNs effectively model interdependence of words, they cannot capture phrases without prefix context and often capture too much of last words in the final vector.

To overcome the problem of compact fixed length vectors in neural MT, Bahdanau et al. (2015) and Luong et al. (2015) proposed an attention mechanism which is a very effective approach to solve this problem by representing the source sentence as a weighted average of the encoder outputs corresponding to each source word.

In this paper, we propose to modify the RNN encoder-decoder framework by adding multiple convolutional layers on top of the RNN output. Since the convolutional neural networks (CNNs) apply to a fixed-size window of the input sentence, at each layer, each output represents a relatively uniform composition of information from multiple words. This provides effective guidance to the network to focus on the relevant parts of the source sentence. At the same time, sequence to sequence modelling as in RNNs is necessary to capture the long-distance dependencies between the segments of the source sentence itself. Thus, in our model, a convolutional encoder complements the standard RNN encoder. Such a combination of RNN and CNN has successfully been used in various tasks such as saliency detection for image recognition (Tang et al., 2016), document modeling (Tang et al., 2015) and music classification (Choi et al., 2016).

We first briefly discuss properties of RNNs, the neural MT framework of Bahdanau et al. (2015) and convolutional neural networks in Section 2 and subsequently discuss the related work on Convolutional neural networks in machine translation in Section 3. We introduce our model in Section 4 and discuss the its details. Experiments and results are discussed in Sections 5 and 6 respectively.

2. Background

2.1. Recurrent Neural Network

Given a sequence $[(x_1, x_2, ..., x_n)]$ of length *'n'*, at any timestep *'i'*, an RNN represents the hidden state output as function of the previous hidden state h_{i-1} output and the current input x_i

$$h_i = f(h_{i-1}, x_i) \tag{1}$$

f is commonly a nonlinear function. Thus RNNs represent a sequence as a vector by a function of previous history and current input. It is this recurrence property of RNNs that makes them capable to capture larger context such as long distance dependencies commonly observed in variable length texts.

A common problem observed while training RNN is the decay of gradient over long distance dependencies. To resolve this problem, Hochreiter and Schmidhuber (1997) proposed long-short term memory networks (LSTM) which use input, output, and forget gates to control the amount of information that can pass through a cell unit in the RNN.

2.2. Neural Machine Translation

We employ an NMT system based on Luong et al. (2015) which is a simple encoder-decoder network. The encoder is a multi-layer recurrent network (we use LSTMs) which converts an input sentence $[(x_1, x_2, ..., x_n)]$ into a sequence of hidden states $[(h_1, h_2, ..., h_n)]$.

$$h_i = f_{enc}(x_i, h_{i-1}) \tag{2}$$

Here, f_{enc} is an LSTM unit. The decoder is another multi-layer recurrent network which predicts a target sequence $y = (y_1, y_2, ...y_m)$. Each word in the sequence is predicted based on the last target word y_{i-1}, the current hidden state of the decoder s_j and the context vector c_j. The probability of the sentence is modelled as product of the probability of each target word.

$$p(\mathbf{y}) = \prod_{j}^{m} p(y_j|y_1, ...y_{j-1}, \mathbf{x}) = \prod_{j}^{m} g(y_j, s_j, c_j) \tag{3}$$

where g is a multi-layer feed forward neural network with nonlinear transformation and a softmax layer which generates the probability of each word in the target vocabulary. The end-to-end network is trained by maximizing log-likelihood over the training data. In Equation 3, s_j is the decoder hidden state generated by LSTM units similar to the encoder.

$$s_j = f_{dec}(s_{j-1}, y_{j-1}, c_j) \tag{4}$$

The context vector c_j in turn is calculated using an attention mechanism (Luong et al., 2015) as weighted sum of annotations of the encoder states h_i's.

$$c_j = \sum_{i=1}^{n} \alpha_{ji} h_i \tag{5}$$

where α_{ji} are attention weights corresponding to each encoder hidden state output h_i calculated as follows :

$$\alpha_{ji} = \frac{exp(z_i)}{\sum_{k=1}^{n} exp(z_k)} \tag{6}$$

Activations $z_k = a(s_{j-1}, h_k)$ are calculated by using a context function such as the dot product between the current decoder state s_{j-1} and each of the hidden states

of the encoder h_k's. Figure 1 shows the NMT framework with the encoder-decoder architecture and attention modeling.

In order to reduce the memory requirement for softmax operation on large number of words, source and target vocabulary are usually clipped to a fixed number of most frequent words.Translation is performed by a simple left-to-right beam search algorithm which maintains a small set of *'b'* best hypotheses for each target word. A hypothesis is complete as soon as end of sentence ("$< EOS >$") symbol is produced. A more detailed description of the decoding algorithm can be found in Sutskever et al. (2014).

2.3. Convolutional Neural Networks

Unlike recurrent neural networks, which are applied to a sequence of inputs, feeding the hidden layer from one timestep to the next, convolutional neural networks apply filters of fixed length over a window of inputs and generate outputs of fixed size. As discussed in (Kim, 2014), a narrow convolution operation involves applying a filter θ over a window of *'w'* inputs in order to generate a new feature. *'w'* is known as the width of the filter. The new feature CN_i applied to input window x_i to x_{i+w} is then defined as :

$$CN_i = \sigma(\theta \cdot x_{i-[(w-1)/2]:i+[(w-1)/2]} + b) \tag{7}$$

This feature extraction capability of CNNs makes them suitable for image processing. In NLP, CNNs have been used for tasks such as sentence classification which require computation of all possible phrases or segments of the input sentence regardless of their grammaticality.

3. Related Work

Although recurrent neural networks are very popular for NLP tasks, CNNs have also been used to model tasks such as text or sentence classification (Kim, 2014), sentiment analysis (dos Santos and Gatti, 2014), document modeling (Tang et al., 2015) and sentence modeling (Kalchbrenner et al., 2014) where specific features such as n-grams and phrases are more important than location specific or grammatical features of the sentence.

Similarly, the standard approach to neural Machine translation is the RNN based encoder-decoder network. However, there have been various attempts recently towards using convolutional networks in neural MT as well as additional models in Phrase-based MT. The first attempt to use convolutional networks in an end-to-end NMT framework is Cho et al. (2014). They fully replace the recurrent encoder with a gated recursive convolutional network whose weights are recursively applied to the input sequence until it outputs a single fixed-length vector. However, their experiments demonstrate that translation performance of such a network cannot surpass that of fully recurrent encoder.

Recently, Gehring et al. (2016) also proposed a similar architecture where the recurrent encoder is again fully replaced by a deep convolutional neural network. An important feature in their architecture is the use of a position embedding which encodes the relative position of each word in the source sentence. Their experiments demonstrate that while translation performance of the network is improved by using a very deep convolutional network, without the position embeddings, it drops substantially below the standard RNN/LSTM encoder baseline. This implies that a CNN encoder by itself with simple word embeddings alone cannot encode position-dependent features which are otherwise efficiently captured by an RNN encoder. Another recent approach using convolutional networks in neural MT is the *ByteNet* system by Kalchbrenner et al. (2016). They attempt to replace both the encoder and decoder with dilated convolutional networks stacked on each other.

All of the above approaches either aim to fully replace the recurrent encoders with convolutional encoders with which they aim to reduce the complexity of the network and the training speed, or to address the variable lengths of input sequences. In order to achieve performance comparable to RNN encoders, these approaches have to employ different mechanisms such as position embeddings to effectively capture the long distance dependencies and position-dependent features.

A related line of research is the character-level approach to NMT (Lee et al., 2016) where the main idea is to model the words as a combination of characters using convolutions and then feed the output as word embeddings to the RNN encoder. Their aim is to avoid the constraint on limited vocabulary by character modeling.

An approach which has shown the strength of convolutional network as an additional feature for Phrase-based MT is Meng et al. (2015). They show improvements over a standard Phrase-based MT by encoding the source sentence with a convolutional network and using it as a neural language model as an additional feature. To the best of our knowledge ours is the first attempt to combine recurrent and convolutional networks to model an encoder for neural machine translation.

4. Convolutional over Recurrent model (CoveR):

As discussed in Section 2, using the standard RNN framework, the context vector is a weighted sum of encoder hidden states h_i. The attention weights as in Equation (6), are also calculated by a similarity function between the decoder state s_j and encoder states $h_i s$. The attention weights mainly score how well the inputs around position *j* and the output at position *i* match (Bahdanau et al., 2015). Since each of these vectors h_i is compact summary of the source sentence up to word *i*, the previous or future context available to the alignment function is only given by these compact global representations. We propose that instead of relying only on these single recurrent outputs, a composition of multiple hidden state outputs of the encoder can provide the attention function with additional context about the relevant features of the source sentence.

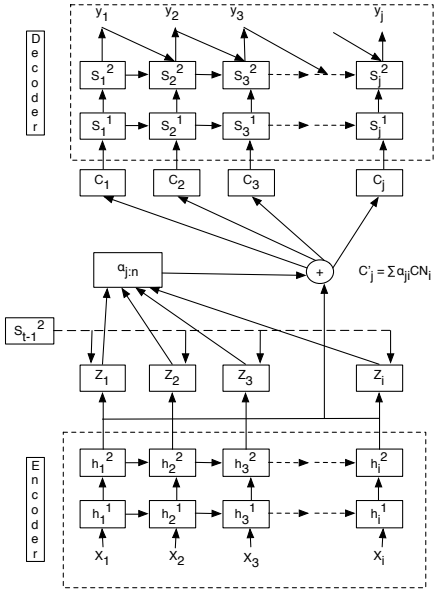


Figure 1. NMT encoder-decoder framework

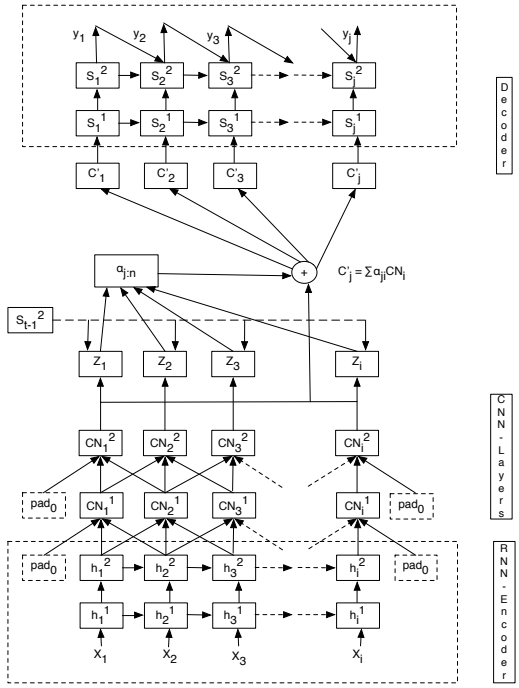


Figure 2. Convolution over Recurrent model

In order to do this, we apply multiple layers of fixed size convolution filters over the output of the RNN encoder at each time step. As shown in Figure 2, for our model the input to the first convolution layer is the hidden state output of the RNN encoder. Thus CN_i^1 is defined as:

$$CN_i^1 = \sigma(\theta \cdot h_{i-[(w-1)/2]:i+[(w-1)/2]} + b) \tag{8}$$

At each layer, we apply a number of filters equal to the original input sentence length. Each filter is of width 3. Note that the length of the output of the convolution filters reduces depending on the input length and the kernel width. In order to retain the original sequence length of the source sentence we apply padding at each layer. That is, for each convolutional layer, the input is zero-padded so that the output length remains the same. The output of the final convolution layer is a set of vectors $[CN_1, CN_2, ...CN_n]$ generated by multiple convolution operations. The modified context vectors c'_i are then calculated similar to c_i using an attention mechanism by

calculating the context function between CN_is and s_j.

$$\alpha_{ji} = \frac{exp(a(s_{j-1}, CN_i))}{\sum_{k=1}^{n} exp(a(s_{j-1}, CN_k))} \tag{9}$$

$$c'_j = \sum_{i=1}^{n} \alpha_{ji} CN_i \tag{10}$$

Finally, the decoder is provided with the context vectors c'_i as follows:

$$p(y_i|y_1, ...y_{i-1}, \mathbf{x}) = g(y_i, s_i, c'_i) \tag{11}$$

Note that each of the vectors CN_i now represents a feature produced by multiple kernels over h_i. Thus each CN_i represents a wider context as compared to h_i.

It is common practice to use pooling along with convolutional filters in order to down-sample the features. However, since in the proposed model, we want to widen the context of the encoder output while still retaining the information represented in the RNN output h_i, and also retaining the original sequence length, we do not apply pooling in our model.

With the increasing depth of the network, the training of the network becomes unstable. In order to ease and stabilize the training with multiple layers, we use residual connections (He et al., 2015) between the input and output of each convolutional layer.

5. Experimental Set-Up

5.1. Data

We conduct experiments on English-German translation. We use the translation data provided for WMT-2015 (Bojar et al., 2015). The training data provided for the task is approximately 4.2 million sentence pairs. We keep source sentences with a maximum sequence length of 80 words. After filtering out sentences longer than this limit and also removing duplicate sentence pairs, we are left with a parallel text of approximately 4 million sentence pairs. We reserve 5000 sentence from this bitext for perplexity validation and use the rest for training. We use wmt-newstest2013 as development set and wmt-newstest2014 and wmt-newstest2015 as test-sets. Results are reported in terms of case-insensitive BLEU-4 (Papineni et al., 2002). Approximate randomization (Noreen., 1989; Riezler and Maxwell, 2005) is used to detect statistically significant differences.

5.2. Baselines

We train a baseline NMT system based on Luong et al. (2015) using the Torch deep learning framework. It is a two layer unidirectional LSTM encoder-decoder with an

	newstest'13 (dev)	newstest'14	newstest'15
Baseline	17.9	15.8	18.5
Deep RNN encoder	18.3	16.2	18.7
CoveR	**18.47**$^{\triangle}$	**16.9**$^{\triangle}$	**19.00**

Table 1. BLEU scores over dev and test sets for Baseline, Deep RNN and CoveR (proposed model). Results marked with $\triangle$ are statistically significant at $p < 0.05$ over baseline

attention (dot product) mechanism. Both the encoder and decoder have input embedding and hidden layer of size 1000. As it is common practice, we limit the vocabulary sizes to 60k for the source and 40k for the target side. Parameters are optimized using stochastic gradient descent. We set the initial learning rate as 1 with a decay rate of 0.5 for each epoch after 5th epoch. Model weights are initialized uniformly within [-0.02, 0.02]. A dropout value of 0.2 is applied for each layer. We train for maximum of 20 epochs and decode with standard beam search with beam size of 10. All models are trained on NVIDIA Titan-X (Pascal) devices.

5.3. CoveR model

As discussed in Section 3, our model is a simple extension of the standard NMT model in which the RNN encoder is extended with additional convolution layers. We add three convolution layers on top of the output of the second RNN layer of the encoder. Note that similar to the baseline system, the RNN decoder has the same number of layers as the RNN encoder i.e., 2. For all layers we apply convolution filters of fixed width 3. The number of filters at each layer is same as the input sequence length. Each filter operates on a window of 3 consecutive inputs and generates a single output with a dimension equal to the input. Thus at each layer the output sequence length is reduced by 2 as compared to input as shown in Figure 2. In order to retain the full sequence length, we apply one zero-padding on both sides of the input. All other optimization parameters are the same as for the baseline.

5.4. Deep RNN encoder

In order to verify that the improvements achieved by the proposed model are due to the convolutions and not just because of the increased number of parameters, we also compare our model to another RNN baseline with an increased number of recurrent layers for encoder. Since we added three convolution layers to the encoder in our proposed CoveR model resulting in a total of 5 layers (2 recurrent + 3 convolution), for a fair comparison, we train a deep encoder with five recurrent layers. For this deep NMT system, the number of layers in decoder remains the same as for the baseline i.e., 2. The initial states of the decoder layers are initialized through a nonlinear transformation of all layers of the encoder RNN. This is done by concatenation of

Example 1:	
Source :	as the reverend martin luther king jr. said fifty years ago
Reference :	wie pastor martin luther king jr. vor fünfzig jahren sagte :
Baseline :	wie der martin luther king jr. sagte
Cover :	wie der martin luther king jr. sagte **vor fünfzig jahren** :
Example 2:	
Source :	he said the itinerary is still being worked out .
Reference :	er sagte , das genaue reiseroute werde noch ausgearbeitet .
Baseline :	er sagte , dass die strecke noch <unk> ist .
Cover :	er sagte , die **reiseroute** wird noch **ausgearbeitet** .

Table 2. Translation examples. Words in bold show correct translations produced by our model as compared to the baseline.

the final states of all the five layers of the encoder resulting in a vector of size 5xD (*'D'* is the dimension of the hidden layer) and then downgrading it to size 2xD by a simple non-linear transformation and finally splitting it in two vectors of size 'D' which are used to initialize each of the layers of the decoder.

6. Results

Table 1 shows the results for our English-German translations experiments. The first column indicates the best BLEU scores on the development set newstest'13 for all three models after 20 epochs. Results are reported on the newstest'14 and newstest'15 test sets. Our CoveR model shows improvements of 1.1 and 0.5 BLEU points respectively over the two test sets. Although the deep RNN encoder performs better than the baseline, the improvements achieved are lower than that of the CoveR model.

6.1. Qualitative analysis and discussion

Table 2 provides some of the translation examples produced by the baseline system and our CoveR model. A general observation is the improved translations by our model over the baseline with regard to the reference translation which is also reflected by the improved BLEU scores.

More specifically, Example 1 shows instances where the baseline suffers in some cases from incomplete coverage of the source sentence. One reason for such incomplete translations is the lack of coverage modeling which has been handled using coverage embeddings (Tu et al., 2016). We observe this problem frequently in instances where a specific word might signal completion of a sentence despite more words in the sequence remain to be be translated. These words can cause the generation of next target word as the end-of-sentence *'EOS'* symbol. Since the beam search decoding algorithm considers a hypothesis complete when the end of sentence is generated, in

such instances search stops, aborting further expansions, while ignoring the remaining words. For instance in Example 1 in Table 2, by relying on the attention mechanism, the baseline system generates the translation of *'said'* as *'sagte'*, the model might give a preference to the generation of an end-of-sentence *'EOS'* symbol immediately following the verb. On the other hand, for our CoveR model, at target position 8, a wider context is available to the model through convolutional layers from both directions signalling the presence of other words remaining in the input sentence, thus producing a more complete translation. Another difference between the baseline model

Figure 3. Attention distribution for Baseline

Figure 4. Attention distribution for CoveR model

and our CoveR model that can be observed in Example 2 is that attention weights are distributed more uniformly among the source words. Specifically, for target position 6, as shown in Figure 3 the baseline model pays attention mainly to *'itinerary'* and

'is' resulting in the generation of target word *'strecke'* which is a more common translation for the English word *'route'*. On the other hand as shown in Figure 4, for the same position, the CoveR model pays attention to *'itinerary'* as well as the last three words *'being worked out'*. This allows for the generation of the more specific and correct target word *'reiseroute'*. Note that the <unk> symbols produced are a result of the vocabulary restriction.

7. Conclusion

In this paper, we proposed a convolutional over recurrent network encoder model for neural machine translation. The model involves feeding outputs of the RNN encoder to multiple convolutional layers of fixed kernel size. Our experiments on English-German translation demonstrate that the proposed model improves translation as compared to a standard RNN encoder. An improvement of 0.5 to 1 BLEU points is observed on different test sets. A qualitative analysis of the translations of our model shows that CNNs capture the smaller context corresponding to each word more effectively while RNNs model the global information thus capturing grammaticality and dependencies with the source sentence.

Bibliography

Bahdanau, Dzmitry, Kyunghyun Cho, and Yoshua Bengio. Neural Machine Translation by Jointly Learning to Align and Translate. In *ICLR*, 2015.

Bojar, Ondřej, Rajen Chatterjee, Christian Federmann, Barry Haddow, Matthias Huck, Chris Hokamp, Philipp Koehn, Varvara Logacheva, Christof Monz, Matteo Negri, Matt Post, Carolina Scarton, Lucia Specia, and Marco Turchi. Findings of the 2015 Workshop on Statistical Machine Translation. In *Proceedings of the Tenth Workshop on Statistical Machine Translation*, September 2015.

Cho, Kyunghyun, Bart van Merrienboer, Dzmitry Bahdanau, and Yoshua Bengio. On the Properties of Neural Machine Translation: Encoder-Decoder Approaches. In *Proceedings of SSST@EMNLP 2014, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation, Doha, Qatar, 25 October 2014*, 2014.

Choi, Keunwoo, George Fazekas, Mark B. Sandler, and Kyunghyun Cho. Convolutional Recurrent Neural Networks for Music Classification. *CoRR*, abs/1609.04243, 2016.

dos Santos, Cicero and Maira Gatti. Deep Convolutional Neural Networks for Sentiment Analysis of Short Texts. In *Proceedings of COLING 2014*, Dublin, Ireland, August 2014. ACL.

Gehring, Jonas, Michael Auli, David Grangier, and Yann N. Dauphin. A Convolutional Encoder Model for Neural Machine Translation. *CoRR*, abs/1611.02344, 2016.

He, Kaiming, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. *CoRR*, abs/1512.03385, 2015.

Hochreiter, Sepp and Jürgen Schmidhuber. Long Short-Term Memory. *Neural Comput.*, 9(8): 1735–1780, 1997. ISSN 0899-7667.

Kalchbrenner, Nal, Edward Grefenstette, and Phil Blunsom. A Convolutional Neural Network for Modelling Sentences. *CoRR*, abs/1404.2188, 2014.

Kalchbrenner, Nal, Lasse Espeholt, Karen Simonyan, Aäron van den Oord, Alex Graves, and Koray Kavukcuoglu. Neural Machine Translation in Linear Time. *CoRR*, abs/1610.10099, 2016.

Kim, Yoon. Convolutional Neural Networks for Sentence Classification. *CoRR*, abs/1408.5882, 2014.

Lee, Jason, Kyunghyun Cho, and Thomas Hofmann. Fully Character-Level Neural Machine Translation without Explicit Segmentation. *CoRR*, abs/1610.03017, 2016.

Luong, Thang, Hieu Pham, and Christopher D. Manning. Effective Approaches to Attention-based Neural Machine Translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, Lisbon, Portugal, 2015. ACL.

Meng, Fandong, Zhengdong Lu, Mingxuan Wang, Hang Li, Wenbin Jiang, and Qun Liu. Encoding Source Language with Convolutional Neural Network for Machine Translation. In *Proceedings of the 53rd Annual Meeting of the ACL*, Beijing, China, July 2015. ACL.

Mikolov, Tomáš, Martin Karafiát, Lukáš Burget, Jan Černocký, and Sanjeev Khudanpur. Recurrent neural network based language model. In *Proceedings of the 11th Annual Conference of the International Speech Communication Association (INTERSPEECH 2010)*, pages 1045–1048. International Speech Communication Association, 2010. ISBN 978-1-61782-123-3.

Mueller, Jonas and Aditya Thyagarajan. Siamese Recurrent Architectures for Learning Sentence Similarity. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, 2016.

Noreen., Eric W. *Computer Intensive Methods for Testing Hypotheses. An Introduction*. Wiley- Interscience, 1989.

Papineni, Kishore, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a Method for Automatic Evaluation of Machine Translation. In *Proceedings of 40th Annual Meeting of the ACL*, 2002.

Riezler, Stefan and John T. Maxwell. On some pitfalls in automatic evaluation and significance testing for MT. In *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Ma- chine Translation and/or Summarization*, 2005.

Sutskever, Ilya, Oriol Vinyals, and Quoc V. Le. Sequence to Sequence Learning with Neural Networks. *CoRR*, abs/1409.3215, 2014.

Tang, Duyu, Bing Qin, and Ting Liu. Document Modeling with Gated Recurrent Neural Network for Sentiment Classification. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, Lisbon, Portugal, September 2015. ACL.

Tang, Youbao, Xiangqian Wu, and Wei Bu. Deeply-Supervised Recurrent Convolutional Neural Network for Saliency Detection. *CoRR*, abs/1608.05177, 2016.

Tu, Zhaopeng, Zhengdong Lu, Yang Liu, Xiaohua Liu, and Hang Li. Coverage-based Neural Machine Translation. *CoRR*, abs/1601.04811, 2016.

Address for correspondence:
Praveen Dakwale
p.dakwale@uva.nl
C3.230, Science Park 904, Amsterdam 1098XH, The Netherlands

PBML

The Prague Bulletin of Mathematical Linguistics
NUMBER 108 JUNE 2017 49–60

Learning Morphological Normalization for Translation from and into Morphologically Rich Languages

Franck Burlot, François Yvon

LIMSI, CNRS, Université Paris-Saclay, France

Abstract

When translating between a morphologically rich language (MRL) and English, word forms in the MRL often encode grammatical information that is irrelevant with respect to English, leading to data sparsity issues. This problem can be mitigated by removing from the MRL irrelevant information through normalization. Such preprocessing is usually performed in a deterministic fashion, using hand-crafted rules and yielding suboptimal representations. We introduce here a simple way to automatically compute an appropriate normalization of the MRL and show that it can improve machine translation in both directions.

1. Introduction

Translating from a morphologically rich language (MRL) like Czech or Russian into a more analytical language like English leads to several issues, due to important divergences in their respective linguistic systems. The MRLs considered in this study have synthetic tendencies, which means that they often encode grammatical information in the endings of words, notably case marks which signal the grammatical function of a word in the sentence. There is no such phenomena in English, where the function of a word is instead encoded in a specific word order or expressed in prepositions. This results in an obvious lack of symmetry between those two types of languages. For instance, while on the MRL side adjectives may vary in gender, number and case, their English translation is invariable. Such differences can impact machine translation (MT) quality in several ways:

 Corresponding author: franck.burlot@limsi.fr
Cite as: Franck Burlot, François Yvon. Learning Morphological Normalization for Translation from and into Morphologically Rich Languages. The Prague Bulletin of Mathematical Linguistics No. 108, 2017, pp. 49–60.
doi: 10.1515/pralin-2017-0008.

- The increase of word forms in the MRL means that each form has a smaller occurrence count than its English counterpart(s), yielding poor probability estimates for infrequent words;
- An even more extreme case is the translation of word forms unseen in training. Even if other forms of the same lemma are known, the MT system cannot generalize and will produce an erroneous output.

A well-known way to mitigate this problem is to "simplify" the MRL by removing information that is deemed redundant with respect to English. This solution has been repeatedly used to translate into the MRL (eg. in (Ney and Popovic, 2004; Durgar El-Kahlout and Yvon, 2010) for German, (Goldwater and McClosky, 2005) for Czech), and is adopted in recent systems competing at WMT (e.g. (Allauzen et al., 2016; Lo et al., 2016) for Russian), as well as in the reverse direction (Minkov et al., 2007; Toutanova et al., 2008; Fraser et al., 2012) with the additional complexity that the simplified MT output needs to be augmented with the missing information ("reinflected" in the MT jargon). One downside of these procedures is that they are entirely dependent on the language pairs under study, and rely on hand-crafted rules that need to be adapted for each new language. It is also likely that rule-based normalization is suboptimal with respect to the task, as it does not take the peculiarities of the training data into account.

We introduce (Section 3) a new way to automatically perform such normalization, by clustering together MRL forms.[1] Clustering is performed on a per lemma basis and groups together morphological variants that tend to translate into the same target word(s). We show in Section 4 that this normalization helps when translating into English. A second contribution is a new neural reinflection system, which is crucially able to also take advantage of source-side information, yielding significant improvements when translating into a MRL (Section 5).

2. Related Work

The normalization of the vocabulary on the MRL side mostly consists in removing word information that is deemed redundant with respect to English. Most of the time, normalization relies on expert knowledge specifying which MRL words can be merged without generating confusion in English, (see eg. (Ney and Popovic, 2004; Goldwater and McClosky, 2005; Durgar El-Kahlout and Yvon, 2010)). An alternative, which does not require user expertise is introduced by Talbot and Osborne (2006), who proposed to use model selection techniques to identify useful clusters in the MRL vocabulary. Even though we start from the same intuition (to cluster forms having similar translation distributions), our model is much simpler and more explicitly oriented toward morphological variation, which makes it also easier to invert.

[1] Our implementation is available at `https://github.com/franckbrl/bilingual_morph_normalizer`.

The same kind of solution is also useful when translating in the reverse direction; it additionally requires a two-step MT architecture addressing morphology as a post-processing step. Minkov et al. (2007) and Toutanova et al. (2008) translate from English into Russian and Arabic stems, which are used to generate full paradigms, then disambiguated using a classifier. Similarly, Chahuneau et al. (2013) augment the translation model with synthetic phrases obtained by re-inflecting target stems. Bojar (2007) cascade two Statistical MT systems: the first one translates from English into Czech lemmas decorated with source-side information and the second one performs a monotone translation into fully inflected Czech.

Fraser et al. (2012) represent German words as lemmas followed by a sequence of tags and introduce a linguistically motivated selection of these in order to translate from English. The second step consists in predicting the tags that have been previously removed, using a dedicated model for each morphological attribute. Finally, word forms are produced by looking-up in a morphological dictionary. El Kholy and Habash (2012a; 2012b) propose a similar approach for Arabic.

3. Source-side Clustering

3.1. Information Gain

Our goal is to cluster together MRL forms that translate into the same target word(s). We assume that each MRL form f is a combination of a lemma, a part of speech (PoS) and a sequence of morphological tags,[2] and that a word aligned parallel corpus is available, from which lexical translation probabilities $p(e|f)$ and unigram probabilities $p(f)$ can be readily computed. We first consider the simple case where the corpus contains one single lemma for each PoS. We denote respectively $\mathbf{f}$ the set of word forms (or, equivalently, of positions in the paradigm) for this lemma, and $\mathbf{E}$ the complete English vocabulary. The conditional entropy (CE) of the translation model is:

$$H(\mathbf{E}|\mathbf{f}) = \sum_{f \in \mathbf{f}} p(f) H(\mathbf{E}|f) = \sum_{f \in \mathbf{f}} \frac{p(f)}{\log_2 |\mathbf{E}_{a_f}|} \sum_{e \in \mathbf{E}_{a_f}} -p(e|f) \log_2 p(e|f), \quad (1)$$

where $\mathbf{E}_{a_f}$ is the set of words aligned with f. The normalizer ($\log_2 |\mathbf{E}_{a_f}|$) ensures that all the entropy values are comparable, no matter the number of aligned target words.

From an initial state where each form is a singleton cluster, and proceeding bottom-up, we repeatedly try to merge cluster pairs (f_1 and f_2) so as to reduce the CE. We therefore compute the information gain (IG) of the merge operation:

$$IG(f_1, f_2) = p(f_1)H(\mathbf{E}|f_1) + p(f_2)H(\mathbf{E}|f_2) - p(f')H(\mathbf{E}|f') \quad (2)$$

[2] For instance, the Czech *autem* (by car) is represented as: *auto + Noun + neutral + singular + instrumental*.

where f' is the resulting aggregate. IG ($\in [-1, +1]$) measures the difference between the combined CEs of clusters f_1 and f_2 before and after merging in f'. If the corresponding forms have similar translation distributions, the information gain is positive; conversely when their translations are different, it is negative and the merge leads to a loss of information. Note that the total entropy $H(\mathbf{E}|\mathbf{f})$ of the translation model can be recomputed *incrementally* after merging (f_1, f_2) by:

$$H(\mathbf{E}|\mathbf{f}) \leftarrow H(\mathbf{E}|\mathbf{f}) - IG(f_1, f_2) \tag{3}$$

IG can also be interpreted as a measure of similarity between two word forms and can be readily used in any clustering model, such as *k-means*. Doing so would however require to fix the total number of clusters, which we would rather like to determine based on the available data. We have therefore opted for an agglomerative clustering procedure, which we now fully describe.

3.2. Clustering Paradigm Cells

In practice, our algorithm is applied at the level of PoS, rather than individual lemmas: we therefore assume that for a given PoS p, all lemmas have the same number n_p of possible morphological variants (cells in their paradigm). This means that IG computations will be aggregated over all lemmas of a given PoS, based on statistics maintained on a per lemma basis. For each lemma of PoS p, the starting point is a matrix $L_l \in [-1:1]^{n_p \times n_p}$, with $L_l(i,j)$ the IG resulting from merging forms l_i and l_j of lemma $\mathbf{l}$. The average of these matrices over all lemmas defines *the PoS level matrix* $M_p \in [-1:1]^{n_p \times n_p}$ containing the average information gain resulting from merging two cells.

Algorithm 1: A bottom-up clustering algorithm

```
C(p) ← {1, ..., n_p}
i, j ← arg max_{i',j' ∈ C(p)^2} M_p(i', j')
repeat
    Merge i and j in C(p)
    for l ∈ V_lem do
        Remove L_l(i, j), create L_l(ij)
        Compute p(ij), p(E|ij) and H(E|ij)
        Compute L_l(ij, k) for k ∈ C(p)
    M_p ← Σ_{l ∈ V_lem} L_l
    i, j ← arg max_{i',j' ∈ C(p)^2} M_p(i', j')
until M_p(i, j) < m or |C(p)| = 1
```

The clustering procedure is described in Algorithm 1. It starts with n_p classes for each PoS and iteratively performs merge operations, as long as the cumulated information gain for the merge exceeds a minimum threshold m. After each merge,

the statistics for the new cluster (unigram probability, translation probability and entropy) are recomputed *for all lemmas* and used to update the PoS-level IG matrix M_p. When the procedure halts, a clustering $C(p)$ is obtained for PoS p, which can then be applied to normalize the source data in various ways (see Section 4.3).

In practice, we obtained slightly better results and a much better runtime than the exact computation of algorithm 1 with an alternative update regime for the IG Matrix M_p, which dispenses with the costly update of all the matrices L_l (lines 5–8). Once initialized, M_p is treated like a similarity matrix and updated using a procedure reminiscent of the linkage clustering algorithm. The aggregated matrix cell for clusters c_1 and c_2 is thus computed as the average IG of all possible 2-way merging operations:

$$M_p(c_1, c_2) = \frac{\sum_{f_1 \in c_1} \sum_{f_2 \in c_2} M(f_1, f_2)}{|c_1| \times |c_2|}. \quad (4)$$

4. Translating from and into a normalized MRL

We assess the normalization model on MT tasks for three language pairs in both directions: Czech-English, Russian-English and Czech-French; note that the latter involves two MRLs.

4.1. Experimental Setup

Tokenization of English and French uses in-house tools. We used the script from the Moses toolkit (Koehn et al., 2007) for Czech and TreeTagger (Schmid, 1994) for Russian. The MT models are trained using Moses with various datasets from WMT 2016[3] (Table 1). 4-gram language models were trained with KenLM (Heafield, 2011) over the monolingual datasets. These systems are optimized with KB-MIRA (Cherry and Foster, 2012) using WMT Newstest-2015 and tested on Newstest-2016. The Czech-French systems were tuned on Newstest-2014 and tested on Newstest-2013.

	cs2en		**en2cs**		**cs2fr**		**fr2cs**		**ru2en**		**en2ru**	
Setup	parall	mono	parall	mono	parall	mono	parall	mono	parall	mono	parall	mono
`Small`	190k	150M	190k	8.4M	622k	12.3M	622k	8.4M	190k	150M	190k	9.6M
`Larger`	1M	150M	1M	34.4M								
`Largest`	7M	250M	7M	54M								

Table 1. Datasets used to train the MT systems

The source-side normalization is performed independently for each dataset, using the training set of the MT system, except for the `Larger` and `Largest` Czech systems for which the parallel data of the `Larger` system was used. The lemmas and tags are

[3] `http://www.statmt.org/wmt16`

obtained with Morphodita (Straková et al., 2014) for Czech and TreeTagger (Schmid, 1994; Sharoff et al., 2008) for Russian. Filtering the MRL lemmas when performing clustering yields better results and we have excluded lemmas appearing less than 100 times, as well as word forms occurring less than 10 times in the training set in order to mitigate the noise in the initial alignments. When clustering paradigm cells (see Section 3.2), we set the minimum IG value $m = 0$.

4.2. A qualitative assessment of normalized Czech

The clustering learned over the `Small` Czech-English data led to a drastic reduction of the source vocabulary. Starting with 158,914 distinct character strings, corresponding to 237,378 fully disambiguated word forms (represented as lemmas and morphological information), we ended up with a set of 90,170 normalized entries.

The resulting clusters confirm some linguistic intuitions. First, nouns turned out to be distinguished only by their number, a property that is also marked for English nouns. We also observed a small number of singleton noun classes, mainly at the instrumental case which often corresponds to the English prepositions *by* and *with* (including the dual number for *rukama* → *with [my] hands*), as well as the vocative case. All possessive pronouns were distinguished only by their person, as is also the case in English; adjectives were clustered separately according to their degree of comparison, verbs are clustered by time, the third person singular of the present tense being separated, since it is marked in English (*I cluster, he cluster**s***). We only noticed a small residual noise with negative verbs, sometimes clustered with affirmative ones. This might be due to alignment errors where an English negation particle is not linked to a Czech negative verb, a typical issue for this language pair (Rosa, 2013). Our model thus seems to be able to capture subtle linguistic phenomena that would require a large amount of rules if such normalization had to be performed manually.

4.3. MT experiments

The results for all Czech systems are in Table 2 and are reported based on different applications of the normalization model. Indeed, normalization can be used to train both the alignment (ali cx) or the full system (cx2en), yielding a total improvement of 1.36 BLEU in the `Small` conditions. Using it only for alignments or only for the MT system gives worse results, still outperforming the baseline (cs2en). This shows that both tasks take advantage of the source normalization. Another way to apply the clustering model is to exclude from normalization the 100 most frequent lemmas (100 freq), which gives the best result for this setup. For the other direction (en2cs), the Czech normalization was used to train the alignments and gives only a slight improvement over the baseline. Results for the translation into normalized Czech (en2cx) after a reinflection step are reported in Section 5.2.

The same tendency holds for the `Larger` Czech-English system, even though the contrasts in BLEU scores are slightly less visible, due to the larger amount of training

	Small System		Larger System		Largest System	
System	BLEU	OOV	BLEU	OOV	BLEU	OOV
cs2en (ali cs)	21.26	2189	23.85	1878	24.99	1246
cx2en (ali cx)	22.62 (+1.36)	1888	24.57 (+0.72)	1610	24.65 (-0.43)	988
cs2en (ali cx)	22.19 (+0.93)	2152	24.14 (+0.29)	1832	**25.35 (+0.36)**	1212
cx2en (ali cs)	22.34 (+1.08)	1914	24.36 (+0.51)	1627		
cx2en (100 freq)	**22.82 (+1.56)**	1893	**24.85 (+1.00)**	1614		
cx2en ($\mathfrak{m} = -10^{-4}$)			24.44 (+0.59)	1604		
cx2en ($\mathfrak{m} = 10^{-4}$)			24.05 (+0.20)	1761		
cx2en (manual)			24.46 (+0.61)	1623		
en2cs (ali cs)	15.21		17.42		19.14	
en2cs (ali cx)	15.54 (+0.33)		17.55 (+0.13)		19.23 (+0.09)	

Table 2. Czech-English Systems

data, which reduces sparsity. For this setup, we also have tried different values of the minimum IG $\mathfrak{m}$ (see Section 3.2). Our results suggest that the optimal value for $\mathfrak{m}$ is close to 0. Indeed, higher values produce more clusters, which leads to more OOVs (1761 OOVs for 10^{-4}, vs. 1604 for $\mathfrak{m} = -10^{-4}$), thus hurting the overall performance.

In the `Largest` Czech-English setup, using normalization to train both the alignments and the translation system hurts the performance (-0.43 BLEU). On the other hand, using it only to train the alignments does give a small improvement. In the reverse direction (en2cs), training the alignments over normalized Czech does not give any significant improvement.

Results for a manual normalization (manual) are also reported. The normalization rules are close to the ones used in (Burlot et al., 2016) where nouns are distinguished by number and negation, adjectives by negation and degree of comparison, etc. We also applied rules for verb clusters that are distinguished by tense and negation, except the singular third person present tense that is kept. This manual normalization improves the baseline (+0.61), but not as much as our best system (+1.00).

System	BLEU	OOV
cs2fr (ali cs)	19.57	1845
cx2fr (ali cx)	**20.19 (+0.62)**	1592
fr2cs (ali cs)	13.36	
fr2cs (ali cx)	13.18 (-0.18)	

Table 3. Czech-French systems

System	BLEU	OOV
ru-en (ali ru)	19.76	2260
rx-en (ali rx)	**21.02 (+1.26)**	2033
rx-en (ali ru)	20.92 (+1.16)	2033
ru-en (ali rx)	20.53 (+0.77)	2048
rx-en (100 freq)	20.89 (+1.13)	2026
en-ru (ali ru)	16.59	
en-ru (ali rx)	16.95 (+0.36)	

Table 4. Russian-English systems

The results for Russian-English follow the same tendency as Czech-English, except that keeping the word forms for the 100 most frequent lemmas did not improve over

the full normalization of the training set. Finally, we note in Table 3 that the Czech normalization towards French also helps to improve the translation, even though the target language is morphologically richer than English. The improvements are smaller, though, than when translating into English. We assume that this is due to a degree of normalization that is lower when the source shares certain properties with the target, such as adjective inflection, which leads our model to create more classes. Indeed, the model distinguishes nouns by their number, just like with English, but moreover creates separate clusters for each adjective gender. This reduced degree of normalization did not help the training of alignments when translating into Czech (fr2cs).

5. Morphological Reinflection

When translating into a MRL, using normalization to train just the alignments did not prove very helpful (Section 4.3). We now consider using it for the complete translation system. Translating from English into fully inflected Czech however requires a non-trivial post-processing step for reinflection. In this section, we introduce our solution to this problem and provide results for several English-Czech systems.

5.1. A Morphological Reinflection Model

We view the reinflection of the normalized MT output as predicting the fine-grained PoS tag for each output token. Knowing the normalized word and its PoS tag is sufficient to recover the fully inflected word form by dictionary lookup.

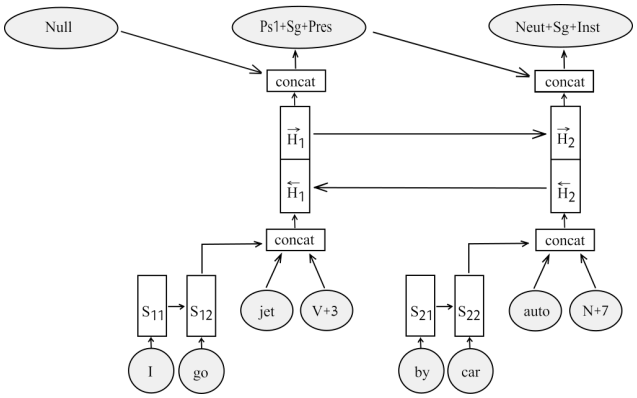


Figure 1. RNN architecture for target-side morphology prediction.

For this sequence labeling task, we used a bidirectional recurrent neural network (RNN) that considers both normalized Czech words as well as source-side English

tokens to make its predictions (see Figure 1). It computes a probability distribution over the output PoS tags $\mathbf{y}$ at time t, given both the Czech ($\mathbf{f}$) and the English ($\mathbf{e}$) sentences, as well as the previous prediction: $p(\mathbf{y}_t|\mathbf{f}, \mathbf{e}, \mathbf{y}_{t-1})$.

For each word f_t in the Czech sentence, we need to encode the English words that generated f_t during the translation process. As there can be an arbitrary number of them (denoted I_t below), we used a RNN layer,[4] where each state S_i inputs a source token representation $a_{t,i}$ and the previous hidden state S_{i-1}. The last state (at time I_t) of that layer is used to represent the sequence of aligned tokens: $S_{t,I} = \mathcal{A}(S_{t,I_t-1}, a_{t,I_t})$.

Each normalized Czech word representation is decomposed into a lemma embedding l_t and a cluster embedding c_t, which are represented in distinct continous spaces. These vectors are concatenated with the source representation S_{t,I_t}, defining the input to the bidirectional RNN[5] performing PoS tagging. A forward layer hidden state H at time t is therefore computed as: $\overrightarrow{H}_t = \mathcal{R}(H_{t-1}, [S_{t,I_t}; l_t; c_t])$. Finally, both forward and backward layers are concatenated with the representation of the preceding PoS tag y_{t-1} [6] and the result is passed through a last feed-forward layer to which a softmax is applied. All the model parameters, including embeddings, are trained jointly in order to maximize the log-likelihood of the training data.

5.2. Experimental Results

The reinflection systems introduced in this section were trained with the parallel English-Czech data used for the `Small` setup (News-Commentary). The fine-grained PoS tags are the same as the ones used to train the normalization in Section 4 (Straková et al., 2014).[7] The word alignments used for the training and validation sets were obtained with fast_align (Dyer et al., 2013). At test time, we use the alignments produced by the MT decoder. Since the Czech side of the parallel data must be normalized prior to training, the results below were obtained with two versions of the RNN model: with the `Small` data normalization and with the `Larger` data one (see Section 4).

Each normalized Czech word is associated with a sequence of source English words that we collect as follows: using word alignments, we take the English words that are linked to the current position, as well as surrounding unaligned words. These unaligned words often contain essential information: as shown in (Burlot and Yvon, 2015), many of them have a grammatical content that is helpful to predict the correct inflection on target side. For instance, the English preposition *of* is an important pre-

[4] Encoding the sequence of aligned tokens with a "bag of words" model, where we just sum the embeddings, performed worse in our experiments.

[5] The RNN layers for English and normalized Czech contain gated recurrent units (Cho et al., 2014).

[6] Representing the full left-side target context with an additional RNN did not bring any improvement.

[7] Our attempts to use the manually annotated data from the Universal Dependency Treebank project (`http://universaldependencies.org`) to train a monolingual variant of our model turned out to give worse results, supposedly because this data is not entirely in-domain.

dictor of the Czech genitive case. This type of grammatical information is the only one that matters for this task, since the lexical content of the Czech words is already computed by the MT system and can not be changed. In fact, replacing the English content words by their PoS and keeping only words in a list of stopwords proved to work better than keeping all the words. Decoding used a beam search of size 5, and the final lookup uses the Morphodita morphological generator.

We consider here three English-Czech MT systems with reinflection. The training data is the same as the `Small`, `Larger` and `Largest` systems described in Section 4, except that the Czech target side is now normalized. The reinflection model can also be used in different ways. One can use it to process the one-best hypothesis of the MT system, or the n-best hypotheses ($n = 300$ in our experiments). A third approach reinflects n-best lists and outputs k-best hypotheses from the reinflection model ($k = 5$ in our experiments). These are finally scored by a language model trained on the same data as the one used in the MT system – albeit with fully inflected words. This score is added to the ones given by the MT system. With nk-best reinflection, we also add the scores given by the reinflection model (log-probability of the predicted sequence). All these scores are finally interpolated using Mira optimization over Newstest-2015 set and produce a single best output sentence.

	Small System			**Larger System**			**Largest System**		
	BLEU ↑	BEER ↑	CTER ↓	BLEU ↑	BEER ↑	CTER ↓	BLEU ↑	BEER ↑	CTER ↓
en2cs (ali cs)	15.21	0.512	0.624	17.42	0.531	0.602	19.14	0.543	0.582
en2cs (ali cx)	15.54	0.516	0.617	17.55	0.532	0.597	19.23	0.544	0.578
en2cx (1-best)	16.07	0.520	0.606	18.00	0.535	0.589	19.19	0.545	0.573
en2cx (n-best)	16.37	0.521	**0.601**	17.41	0.529	0.591	19.48	0.547	**0.570**
en2cx (nk-best)	**16.93**	**0.525**	0.602	**18.81**	**0.540**	**0.588**	**19.95**	**0.548**	0.572

Table 5. BLEU scores for English-Czech

Results are in Table 5, where we provide, in addition to BLEU, scores computed by BEER (Stanojević and Sima'an, 2014) and CharacTER (Wang et al., 2016). These two metrics proved to be more adapted to MRLs by Bojar et al. (2016). We observe a slight improvement when reinflecting the 1-best hypothesis in the `Small` data conditions. With the `Largest` dataset, the reinflection has nearly no impact on the translation quality according to BLEU and BEER. Like for the reverse direction, the improvements of normalization get lower as the size of the dataset grows. We were nevertheless able to obtain a reasonable improvement of 0.81 BLEU points over the baseline in the `Largest` data conditions, which shows that even when a huge quantity of data is available, a specific handling of morphology on target side can still be useful.

6. Conclusion

We have introduced a simple language agnostic way to automatically infer the normalization of a morphologically rich language with respect to the target language that consists in clustering together words that share the same translation, and have shown that it improves machine translation in both directions. Future work will consist in testing our model on neural machine translation systems.

Acknowledgments

This work has been partly funded by the European Union's Horizon 2020 research and innovation programme under grant agreement No. 645452 (QT21).

Bibliography

Allauzen, Alexandre, Lauriane Aufrant, Franck Burlot, Ophélie Lacroix, Elena Knyazeva, Thomas Lavergne, Guillaume Wisniewski, and François Yvon. LIMSI@WMT16: Machine Translation of News. In *Proc. WMT*, pages 239–245, Berlin, Germany, 2016.

Bojar, Ondřej. English-to-Czech Factored Machine Translation. In *Proc. of the 2nd WMT*, pages 232–239, Prague, Czech Republic, 2007.

Bojar, Ondřej, Yvette Graham, Amir Kamran, and Miloš Stanojević. Results of the WMT16 Metrics Shared Task. In *Proc. WMT*, pages 199–231, Berlin, Germany, 2016.

Burlot, Franck and François Yvon. Morphology-Aware Alignments for Translation to and from a Synthetic Language. In *Proc. IWSLT*, pages 188–195, Da Nang, Vietnam, 2015.

Burlot, Franck, Elena Knyazeva, Thomas Lavergne, and François Yvon. Two-Step MT: Predicting Target Morphology. In *Proc. IWSLT*, Seattle, USA, 2016.

Chahuneau, Victor, Eva Schlinger, Noah A. Smith, and Chris Dyer. Translating into Morphologically Rich Languages with Synthetic Phrases. In *EMNLP*, pages 1677–1687, 2013.

Cherry, Colin and George Foster. Batch Tuning Strategies for Statistical Machine Translation. In *Proceedings of the NAACL-HLT*, pages 427–436, Montreal, Canada, 2012.

Cho, Kyunghyun, Bart van Merrienboer, Dzmitry Bahdanau, and Yoshua Bengio. On the Properties of Neural Machine Translation: Encoder-Decoder Approaches. In *Proc. SSST@EMNLP*, pages 103–111, Doha, Qatar, 2014.

Durgar El-Kahlout, Ilknur and François Yvon. The pay-offs of preprocessing for German-English Statistical Machine Translation. In *Proc. IWSLT*, pages 251–258, Paris, France, 2010.

Dyer, Chris, Victor Chahuneau, and Noah A. Smith. A Simple, Fast, and Effective Reparameterization of IBM Model 2. In *Proc. NAACL*, pages 644–648, Atlanta, Georgia, 2013.

El Kholy, Ahmed and Nizar Habash. Translate, Predict or Generate: Modeling Rich Morphology in Statistical Machine Translation. In *Proc. EAMT*, pages 27–34, Trento, Italy, 2012a.

El Kholy, Ahmed and Nizar Habash. Rich Morphology Generation Using Statistical Machine Translation. In *Proc. INLG*, pages 90–94, 2012b.

Fraser, Alexander, Marion Weller, Aoife Cahill, and Fabienne Cap. Modeling Inflection and Word-Formation in SMT. In *Proc. EACL*, pages 664–674, Avignon, France, 2012.

Goldwater, Sharon and David McClosky. Improving Statistical MT through Morphological Analysis. In *Proc. HLT–EMNLP*, pages 676–683, Vancouver, Canada, 2005.

Heafield, Kenneth. KenLM: Faster and Smaller Language Model Queries. In *Proc. WMT*, pages 187–197, Edinburgh, Scotland, 2011.

Koehn, Philipp, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondřej Bojar, Alexandra Constantin, and Evan Herbst. Moses: Open Source Toolkit for Statistical MT. In *Proc. ACL:Systems Demos*, pages 177–180, Prague, Czech Republic, 2007.

Lo, Chi-kiu, Colin Cherry, George Foster, Darlene Stewart, Rabib Islam, Anna Kazantseva, and Roland Kuhn. NRC Russian-English Machine Translation System for WMT 2016. In *Proc. WMT*, pages 326–332, Berlin, Germany, 2016.

Minkov, Einat, Kristina Toutanova, and Hisami Suzuki. Generating Complex Morphology for Machine Translation. In *Proc. ACL*, pages 128–135, Prague, Czech Republic, 2007.

Ney, Hermann and Maja Popovic. Improving Word Alignment Quality using Morpho-syntactic Information. In *Proc. COLING*, pages 310–314, Geneva, Switzerland, 2004.

Rosa, Rudolf. Automatic post-editing of phrase-based machine translation outputs. Master's thesis, Institute of Formal and Applied Linguistics, Charles University, 2013.

Schmid, Helmut. Probabilistic Part-of-Speech Tagging Using Decision Trees. In *Proceedings of International Conference on New Methods in Language Processing*, Manchester, UK, 1994.

Sharoff, Serge, Mikhail Kopotev, Tomaz Erjavec, Anna Feldman, and Dagmar Divjak. Designing and Evaluating a Russian Tagset. In *Proc. LREC*, pages 279–285, Marrakech, Marocco, 2008.

Stanojević, Miloš and Khalil Sima'an. Fitting Sentence Level Translation Evaluation with Many Dense Features. In *Proc. EMNLP*, pages 202–206, Doha, Qatar, 2014.

Straková, Jana, Milan Straka, and Jan Hajič. Open-Source Tools for Morphology, Lemmatization, POS Tagging and Named Entity Recognition. In *Proc. ACL: System Demos*, pages 13–18, Baltimore, MA, 2014.

Talbot, David and Miles Osborne. Modelling Lexical Redundancy for Machine Translation. In *Proc. ACL*, pages 969–976, Sydney, Australia, 2006.

Toutanova, Kristina, Hisami Suzuki, and Achim Ruopp. Applying Morphology Generation Models to Machine Translation. In *Proc. ACL-08: HLT*, pages 514–522, Columbus, OH, 2008.

Wang, Weiyue, Jan-Thorsten Peter, Hendrik Rosendahl, and Hermann Ney. CharacTer: Translation Edit Rate on Character Level. In *Proc. WMT*, pages 505–510, Berlin, Germany, 2016.

Address for correspondence:
Franck Burlot
franck.burlot@limsi.fr
LIMSI-CNRS
Campus Universitaire Orsay, 91 403 Osay, France

PBML

The Prague Bulletin of Mathematical Linguistics
NUMBER 108 JUNE 2017 61–72

Integration of a Multilingual Preordering Component into a Commercial SMT Platform

Anita Ramm,[a] Riccardo Superbo,[b] Dimitar Shterionov,[b] Tony O'Dowd,[b] Alexander Fraser[c]

[a] IMS, University of Stuttgart
[b] KantanMT.com
[c] CIS, LMU Munich

Abstract

We present a multilingual preordering component tailored for a commercial Statistical Machine translation platform. In commercial settings, issues such as processing speed as well as the ability to adapt models to the customers' needs play a significant role and have a big impact on the choice of approaches that are added to the custom pipeline to deal with specific problems such as long-range reorderings.

We developed a fast and customisable preordering component, also available as an open-source tool, which comes along with a generic implementation that is restricted neither to the translation platform nor to the Machine Translation paradigm. We test preordering on three language pairs: English→Japanese/German/Chinese for both Statistical Machine Translation (SMT) and Neural Machine Translation (NMT). Our experiments confirm previously reported improvements in the SMT output when the models are trained on preordered data, but they also show that preordering does not improve NMT.

1. Introduction

Statistical Machine Translation (SMT) is still the most widely used machine translation paradigm in commercial translation services. Unlike previous approaches, SMT models can be trained very fast and do not require any language-specific knowledge, but only parallel bilingual data. Translation quality in SMT typically depends on the quality and quantity of the training data, but also on the syntactic and morphological

 Corresponding author: ramm@ims.uni-stuttgart.de
Cite as: Anita Ramm, Riccardo Superbo, Dimitar Shterionov, Tony O'Dowd, Alexander Fraser. Integration of a Multilingual Preordering Component into a Commercial SMT Platform. The Prague Bulletin of Mathematical Linguistics No. 108, 2017, pp. 61–72. doi: 10.1515/pralin-2017-0009.

differences between the source and the target languages. One of the most common and well analysed problems in SMT is how to place translated words in the correct order with respect to the target language. Often, when the source language (SL) and the target language (TL) have a different syntax, SMT places the TL words in incorrect positions or even omits them. The former case hinders translation fluency, but usually does not strongly affect the meaning of the translation. The latter case, however, damages translation adequacy and may have a negative effect on the conveyed meaning, because specific information given in the source may be missing in the translation.

One of the simplest, yet most effective ways to deal with reordering problems in SMT is to move the words in the SL to positions that are typical of the TL prior to training and translation. This approach, called *preordering*, is performed using rules which describe movements of words or word sequences, typically expressed in terms of part-of-speech (POS) tags or syntactic subtrees. Preordering decreases the syntactic differences between SL and TL sentences and allows for a correct alignment of words in discontiguous phrases. By making the SL and TL look more similar, the long-range reorderings, which are troublesome for automatically-learnt lexicalised reordering models, become much less problematic. The work published on preordering (see Section 2) reports very impressive improvements in the translation quality.

In this paper, we describe the design and the implementation of a preordering approach as well as its integration into KantanMT[1], a commercial custom MT platform. Both the implementation and the integration need to observe the following set of requirements: (i) the implementation must be customisable according to the clients' needs; (ii) the integration of preordering into the training and translation pipelines of the platform should happen seamlessly and sustain backward compatibility; and (iii) the newly integrated preordering component should add as little overhead as possible to the total training/translation time. We focus on the extendible implementation of a preordering component and show how it can be tailored to each user's individual needs. We tested our preordering component on three language pairs: English (EN)→German (DE)/Japanese (JA)/Chinese (ZH)[2], and report results gained when different parsers are used. Despite the fact that our preordering component is inspired by SMT, it can seamlessly be applied to NMT as well. Our experiments will however show that preordering does not improve NMT.

The remainder of the paper is structured as follows. Section 2 briefly outlines the relevant previous work and motivates the development of the preordering component in the present work. In Section 3, we present the reordering rules for the language pairs under consideration. In Section 4, we describe in detail the implementation of the preordering component. In Section 5, we evaluate its effects within the extended MT platform. Finally, we draw conclusions in Section 6.

[1] `https://kantanmt.com/`

[2] By Chinese, we mean Simplified Mandarin Chinese.

2. Related work

Many approaches have been proposed to deal with reordering problems within SMT. One of the simplest, yet most effective methods is *preordering*, which involves a modification of the SL data prior to training and translation. The reordering rules are usually defined on the basis of POS tags and/or syntactic node labels in the source language parse trees. The rules may be hand-crafted or automatically derived from the word-aligned parallel texts. They may be deterministic (i.e., leading to a single reordered variant of the given source sentence) or non-deterministic (i.e., leading to several variants of the source sentence). An extensive overview of different preordering approaches is presented by Bisazza and Federico (2016).

Xu et al. (2009) and Nakagawa (2015) proposed preordering methods which can be applied to many different language pairs. In a multilingual environment, such methods are certainly very convenient. Moreover, the method advanced by Nakagawa (2015) is very fast, as it does not require any linguistic preprocessing (e.g., tagging or parsing) of the training data. Thus, it additionally fits the speed requirements of commercial MT software. However, the approach discussed by Nakagawa (2015) is non-deterministic: a single source sentence is transformed into a number of different reordered variants. As such, it requires lattice-based tuning and decoding which is not supported by the in-house pipeline that we aim to extend. When many different rules can be applied to a single sentence, it also becomes difficult to track errors in the MT output which may be caused by incorrect reordering rules. In the context of commercial settings, however, we need to have the possibility to (manually) improve the rules in order to further increase the quality of the generated translations.

To allow for modification and adaptation of the reordering rules, and encouraged by the simplicity of the deterministic preordering approaches as well as by the improvements reported for the deterministic rules, we present the implementation and integration of the deterministic preordering approach into a commercial, multilingual MT platform. Like Xu et al. (2009), we work with a single source language (English) which translates into three different target languages: German, Japanese and Chinese. Our method relates to the approaches proposed by Gojun and Fraser (2012) for EN→DE and Lee et al. (2010) for EN→JA translation. Both approaches use a set of deterministic hand-crafted reordering rules and apply them to the source-side (i.e., English) constituency parse trees. Both works report on significant improvements in the MT outputs.

3. Reordering rules

Our rule sets are hand-crafted by the language-pair experts taking into account the rules described by Gojun and Fraser (2012) and Lee et al. (2010).

English-German. The main syntactic difference between English and German is the

position of the verbs. Depending on the clause type, the verbs in German may be in the second position (SVO) or in the last position (SOV) in a clause, while in English the sentence structure is always SVO. Our rule set is based on the rules described by Gojun and Fraser (2012). We defined nine reordering rules which move the verbal elements of the English verbal phrases, as well as the negation particle *not*. The rules are conditioned by the clause types (e.g., S, SBAR) since the position of the German verbs depends on the type of clause in which they occur.

English-Japanese. English and Japanese differ in many syntactic aspects: the order of the clauses is different, as well as the order of the words within the clauses. An extensive overview of the differences on various syntactic levels can be found in Bisazza and Federico (2016). The rule set for Japanese is taken from Lee et al. (2010). We only applied context-free grammar (CFG) rules and omitted context-sensitive grammar (CSG) rules, since the information required for such rules is not given in our parse trees. In total, we defined seven reordering rules which change the position of specific subordinate clauses and parts-of-speech (i.e., verbs or conjunctions). Additionaly, we defined one insertion rule to handle null subjects in Japanese.

English-Chinese. To our knowledge, there is no previous work on preordering for EN→ZH. Wu (2016) manually inspected Chinese SMT output and categorised errors related to reordering problems. Relying on this work, as well as on the work on preordering for ZH→EN proposed by Wang et al. (2007), we defined a set of rules which include clause movements, such as moving subordinate clauses before main clauses, as well as various phrase movements. However, this set of rules did not lead to satisfying results and, after further investigation, we reduced the set to only two different rules: (i) moving all PPs before the modifying noun, (ii) moving only PPs with the preposition *of* in front of the modifying noun (this is a subset of the movements defined by the preceding rule). We report results for using either rule (i) or rule (ii)).

4. Implementation

The KantanMT platform is based on the SMT Moses toolkit (Koehn et al., 2007). The preordering component acts as part of the data preprocessing step in the training and translation pipelines.[3] It is applied to the training/testing data before all the other corpus-processing steps, such as lowercasing, cleansing, etc. Preordering is invoked only if reordering rules for the given language pair are provided. Otherwise, the processing pipeline simply skips the preordering step.

[3]A simplified version of the preordering component is freely available for research purposes: https://github.com/KantanLabs/KantanPreorder.

4.1. Pipeline overview

The preordering component is implemented as a three-step process that uses the training, tuning and testing data in the SL (English) as input data, and reorders it sentence by sentence. The processing steps are illustrated in Figure 1. For a general sentence ω, we first generate the constituent parse tree T_ω. We then apply the tree modifications according to our reordering rules to generate the reordered tree T^r_ω. Finally, we read out the reordered sentence ω^r from the modified tree T^r_ω.

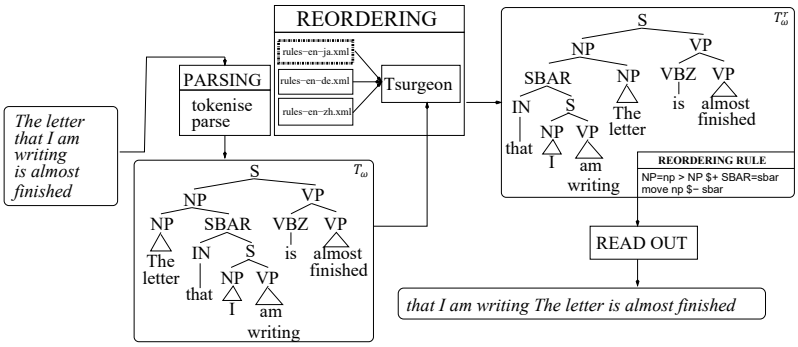


Figure 1. Pipeline for the multilingual preordering. The figure shows tree modification for one of the rules used for Japanese.

Parsing of SL data. Our approach employs reordering of English constituent parse trees. We use the Stanford Shift-Reduce (SR) constituency parser (Zhu et al., 2013) to generate these trees mainly because of its speed (see Section 5). But our implementation also works with two other parsers: the Charniak-Johnson parser (Charniak and Johnson, 2005) and the Stanford PCFG constituency parser (Klein and Manning, 2003). The preordering component does not parse nor reorder sentences that are longer than 60 words, shorter than 5 words or contain many special characters. We impose this restriction because parsers may generate incorrect parse trees or take too long to parse such sentences.

Tsurgeon-based reordering. For modifications of the parse trees, we employ Tsurgeon (Levy and Andrew, 2006) – a tool for parse tree editing based on regular expressions. Tsurgeon first uses a pattern, defined as a Tregex expression, to identify specific subtrees. These expressions make use of relational dependencies between tree nodes, such as *immediate dominance* and *precedence*. Once a subtree matches the pattern, Tsurgeon applies basic tree transformations to it (e.g., move, insert, etc.). An example is given in Figure 1: the applied rule shows the movement (*move np $- sbar*) of the relative clause under the SBAR node (*SBAR=sbar*) in front (*$-*) of the modifying

noun phrase (*NP=np*). Before applying reordering rules to a parse tree, we modify the tree to ensure that reordering operations will not cross the clause boundaries. That is to say, no word movement will place a word outside of the corresponding clause. Afterwards, we apply the language-pair specific reordering operations.

Reading out the reordered sentences. Given the modified parse trees, the reordered sentences are read out by gathering the terminal nodes. Subsequently, the entire source language data undergoes the tokenisation and lowercasing steps.

4.2. Optimised performance and scalability

Training and translation speeds are crucial for a commercial MT system's quality of service. To perform preordering efficiently, we developed the preordering component with a distributed software architecture, and optimised both the parser and Tsurgeon.

First, we run the parser as a simple web service on the machine used for training or translation. This ensures that the parsing model is loaded into the memory prior to parsing any sentence. Furthermore, running the parser as a simple web service makes the implementation independent from the parsing software used. Next, we modify Tsurgeon and introduce a limited-depth search in order to avoid infinite loops and ensure that the reordering of a single sentence will always terminate. In case either the parser or Tsurgeon fails, we output the original sentence. This way, we preserve coherence within the training/translation data.

Our preordering component uses GNU parallel (Tange, 2011) to distribute the workload on all available cores. In particular, we divide the data into as many parts as the CPUs and run preordering for each part in parallel and asynchronously. The GNU parallel tool orchestrates the execution and ensures that the output is serialised correctly. The parallel architecture leads to a substantially lower reordering time as compared to the non-parallel implementation. In a particular test case for EN→DE, involving the reordering of 5000 segments with the Stanford shift-reduce parser, the parallel implementation on 8 cores took 46.10s, whereas the serial took 263.24s.

The run-time depends on the complexity of the sentences, number of the rules, the parser, as well as parsing model used, etc. Analysis of the effects of these factors on the efficiency is out of the scope of this paper and shall be addressed in future work.

4.3. Customisability

Since the tree modifications are based on Tsurgeon, they are independent of the language pair for which the reordering is to be performed (as long as there is a corresponding constituent parse tree). Thanks to the well-defined syntax of the rules, it is easy to extend and/or modify the existing rule sets. In order to add reordering for a new target language (and English as the source language), it is only necessary to specify the new reordering rules. That is, no further adaptation of the training/translation

pipelines is needed. Applying reordering to a new source language would, however, require the pipelines to be adapted and new parsers or models to be incorporated. In principle, this is an easy task thanks to the generic implementation of the preordering component.[4] Furthermore, since some rules are shared across languages, there are cases where already existing rules can be re-used for new language pairs. For instance, a rule for moving verbs at the end of the clause can be used for all SVO-SOV language pairs. Rules can be developed by anyone familiar with Tsurgeon syntax. However, language proficiency and translation experience are required in order to create a valid set of rules.

5. Evaluation

We evaluated our preordering component in different settings and examined the benefit of preordering on both SMT and NMT. All models were trained on the same sets of data from the legal domain. The German models were trained on 1,018,738 parallel sentences, while for Japanese and Chinese, the training data consisted of 213,592 and 387,275 sentences, respectively. The models were tuned and tested on 500 in-domain sentences. The MT quality is evaluated in terms of BLEU (Papineni et al., 2002), TER (Snover et al., 2006) and F-Measure (Melamed et al., 2003). In addition, we give the time required to preorder the SL data used to build the translation models.

5.1. Preordering and SMT

The models were trained with the SMT Moses toolkit (Koehn et al., 2007). We used Moses default settings, including the lexicalised reordering model with distortion limit of 6 words. The 5-gram language models were trained with the target side of the parallel training data. We used *fast_align* for word alignment (Dyer et al., 2013). Model weights were tuned with MERT (Och, 2003) with a maximum of 25 iterations.

The evaluation results, as well as the total training time (including reordering and tuning time), are given in Table 1. The scores show that the quality of the translations varies when different parsers are used. For all target languages, the MT output improves for all parsers. The highest MT improvement for German (+1,39 BLEU) is obtained when the BLLIP parser is used, while the Japanese translations improve the most when reordering is performed on the output produced through the SR parser (+1,89 BLEU). The Chinese MT output profits the most from the reordering of *of*-PPs in the PCFG trees (+0,13 BLEU).

Parsing accuracy depends on the domain and type of training data (Kummerfeld et al., 2012). It may thus be interesting to extend the preordering component with domain-dependent parsing software, as well as domain-specific parsing models.

[4] In Section 5, we present our experiments with two different parsers and three different models, which is evidence of the extendibility of our tool.

	Baseline				SR					PCFG					BLLIP				
	TER	F	BLEU	t_t	TER	F	BLEU	t_r	t_t	TER	F	BLEU	t_r	t_t	TER	F	BLEU	t_r	t_t
EN→DE	51.73	64.21	40.1	187	49.57	65.53	40.74	97	254	49.7	65.57	41.17	372	579	50.45	64.6	41.49	1279	1468
EN→JA	54.02	78.22	49.44	135	51.87	76.34	51.33	25	155	52.04	77.43	50.29	413	544	54.54	76.04	51.33	372	492
EN→ZH (ppNP)	66.27	61.04	24.99	197	66.1	60.55	24.4	50	245	65.97	61.01	24.47	252	460	66.76	60.75	24.66	627	819
EN→ZH (ofPP)					66.69	61.50	25.09	49	240	66.11	61.53	25.22	269	464	67.32	61.36	25.05	633	820

Table 1. Automatic evaluation scores (given as percentages) for the SMT models together with the time (given in minutes) for reordering (t_r) and the total training time (t_t), including tuning. The run times relate to the 8-core CPU machines. Parsers: SR: Stanford shift-reduce parser, PCFG: Stanford PCFG parser, BLLIP: Charniak/Johnson parser.

5.2. Preordering and NMT

Training setting. Our NMT models are built on the same data as our SMT models, after removing duplicates of source-target sentences. We used the open-source toolkit OpenNMT (Klein et al., 2017) to train a single RNN (Recurrent Neural Network) encoder-decoder model (Cho et al., 2014; Sutskever et al., 2014) with attention mechanism (Bahdanau et al., 2014). We used a word-segmentation with byte pair encoding (BPE) (Sennrich et al., 2016) of 25,000 operations for English and German. We built the BPE dictionary from normal-cased (i.e., lower- and upper-cased) tokens bypassing the requirement for a recasing model. For Chinese and Japanese, we used a character-based segmentation (Chung et al., 2016). Each network was trained on one GPU (NVIDIA K520, 4GB RAM) for a maximum of 15 epochs, using the ADAM (Kingma and Ba, 2015) learning optimisation function with initial learning rate of 0.005.

We need to highlight that our NMT pipeline is optimised for speed (e.g., EN→JA models are built in less than 8 hours); within the scope of this work, we did not aim to build NMT models that perform better than SMT (according to the automatic metrics), but rather to explore the impact of preordering on NMT.

Automatic evaluation. The evaluation scores are presented in Table 2. In addition to TER, F-Measure and BLEU, we also give the models' perplexity during training to assess the effect of reordering on NMT engines. The scores indicate that, overall, preordering does not improve the quality of NMT models. On the contrary, all metrics, including perplexity, are better for the baseline models. However, we ought to note that the EN-ZH (*of*PP) NMT model has the highest BLEU score for EN→ZH. This result, although episodic for our data, indicates that preordering can have a positive effect under certain conditions. This calls for further, in-depth analysis, which we plan to address in future work.

Human evaluation. Automatic evaluation metrics often tend to misjudge NMT quality (Shterionov et al., 2017). Therefore, we carried out human evaluation tests on Chinese (80 sentences) and German (250 sentences) MT output. For each of the two

	Baseline						SR					
	Perplexity	TER	F	BLEU	Human	t_e	Perplexity	TER	F	BLEU	Human	t_e
EN-DE	2.83	54.63	63.07	38.26	49.2	123	2.94	54.84	61.42	36.74	50.8	123
EN-JA	1.41	27.44	84.54	67.66	–	31	1.5	35.28	80.62	60.77	–	31
EN-ZH (ppNP)	3.46	63.34	61.01	27.65	36.9	91	3.71	67.15	59	26.67	30.7	91
EN-ZH (ofPP)							3.66	65.48	60.37	28.75	32.4	91

Table 2. Scores (given in percentages) together with training time (given in minutes) for one epoch (t_e) for the baseline and reordered NMT models. The human evaluation indicates the percentage of sentences for which the translation is deemed better.

EN	The Commission *may*, in any case, *withdraw* **such products** or substances in accordance with Article37(2).
ENr	The Commission *may*, in any case, **such products** or substances in accordance with Article37(2) *withdraw*.
B	Die Kommission *kann* in jedem Fall **diese Produkte** oder Stoffe gemäß Artikel37 Absatz2 *zurückziehen*.
R	Die Kommission *kann* in jedem Fall **solche Erzeugnisse** oder Stoffe gemäß Artikel37 Absatz2 *zurückziehen*.
REF	Kommission *kann* in jedem Fall **solche Erzeugnisse** oder Stoffe gemäß Artikel37 Absatz2 *zurückziehen*.

Table 3. Example of baseline (B) and reordered (R) translation of a sentence EN and its reordered version ENr. The verbs are indicated in italic, while the differing object NPs are given in bold. The German reference is indicated as REF.

languages, two reviewers compared randomly selected MT sentence pairs (obtained using reordered (R) and baseline (B) training data). For EN-DE they had to indicate which of the two translations was better or whether they were the same; for EN-ZH they had to compare three translations (B, ppNP and ofPP) and score each of them on the scale of 1 to 5. We mainly notice: (i) the translation quality of the B translations is slightly better than that of the R translations, (ii) reordering does not seem to impact the placement of (single) words in the NMT output, but it may lead to syntactically completely different translations, as well as different lexical choices, and (iii) often the B models already correctly translate sentences which our preordering aims to correct.

Discussion. Bentivogli et al. (2016) showed that NMT deals very well with word order issues for EN→DE. Mainly, this is because the RNN encoder-decoder model encapsulates knowledge of the complete input sentence. That is, a complete input sentence is mapped to a complete output sentence, contrary to phrase-based SMT where one sentence is handled phrase by phrase. This allows NMT to deal with both short- and long-distance order issues much more efficiently. Despite showing great improvement when compared to SMT, NMT still makes some mistakes in relation to the placement of words in the translations. We applied preordering on NMT to examine the possibility to reach further improvement in the NMT quality. Our experiments showed that preordering is not beneficial for NMT based on RNNs. This may be explained by the fact that preordering is applied on some, but not all source sentences (depending on the parser's accuracy and coverage of the preordering rules), which leads to noisy training data. Although adding noise to a neural network may improve the generalisation abilities of the network (Jim et al., 1994; Bishop, 1995), in our experiments we did not use any technique to accommodate any excessive noise introduced by the reordering, which may result in a lower network performance.

For future works, we plan to investigate in depth this hypothesis, and upgrade our preordering component to address performance issues, aiming to improve the translation quality of NMT models.

5.3. Processing time vs. translation quality improvement

Since the processing time plays an important role for commercial MT, we ought to investigate whether the improvements reported in Section 5.1 justify the longer processing time. Given the baseline training time (see Table 1), the total training time increases by 36% for EN→DE and 15% for EN→JA when the fastest parser (SR parser) is used. BLEU improvements, and even more importantly, positive feedback of our clients, justify longer processing time for both language pairs. For Chinese, the increase is 24% for EN→ZH (*of*PP) and 22% for EN→ZH (ppNP). On the other hand, the PCFG (4-6 hours) and BLLIP (6-20 hours) parsers lead to a non-acceptable increase of the training time, although for German and Chinese, the best translations are obtained using the BLLIP and PCFG parser, respectively. Future work will aim at making these parsers faster so as to be usable within our commercial SMT platform.

In some settings, however, an increase of processing time may be acceptable if it promises high-quality translations. For example, for translation via API, where only a few segments are translated at once, the increase in time is negligible. Furthermore, if a single model is to be used for many decoding iterations, one could consider training it using a slower, but better parser. Ultimately, it is up to the clients to decide how fast the translations of the provided test sets are to be generated. Given the evaluation results and the training times of the SMT models, we suggest employing the fastest SR parser.

6. Conclusion

We presented a generic component for preordering that is integrated in the corpus preprocessing step for a commercial MT platform. Reordering of the SL sentences is based on Tsurgeon, a tool for editing parse trees based on regular expressions. Thanks to the well-defined syntax of the Tsurgeon expressions, the preordering component is easy to maintain and to extend to other language pairs.

We implemented deterministic rule-based reordering because it performs well, and we can control and adapt it, if needed, to maximise the translation quality. Furthermore, due to its deterministic character, we are not forced to choose between different reorderings of a single sentence or to modify the pipeline into which the preordering component has been integrated.

We described how to achieve preordering speeds that can satisfy the high performance demands of commercial MT software. We showed that a single preordering pipeline can successfully be applied to three different language pairs. Furthermore, the EN→ZH language pair has not been handled this way before. Our experiments

confirmed previously reported improvements for combining preordering with SMT. Additionally, we applied preordering to NMT and observed that NMT does not generally benefit from the reordering of the source training data. In the future work, we will further investigate impact of the preordering approach on NMT.

Acknowledgements
This research was supported by the European Association for Machine Translation.

Bibliography

Bahdanau, Dzmitry, Kyunghyun Cho, and Yoshua Bengio. Neural Machine Translation by Jointly Learning to Align and Translate. *ICLR*, May 2014.

Bentivogli, Luisa, Arianna Bisazza, Mauro Cettolo, and Marcello Federico. Neural versus Phrase-Based Machine Translation Quality: a Case Study. In *EMNLP*, November 2016.

Bisazza, Arianna and Marcello Federico. A Survey of Word Reordering in Statistical Machine Translation: Computational Models and Language Phenomena. *Computational linguistics*, 42(2), 2016.

Bishop, Chris M. Training with Noise is Equivalent to Tikhonov Regularization. *Neural Computation*, 7(1), January 1995.

Charniak, Eugene and Mark Johnson. Coarse-to-fine *n*-best parsing and MaxEnt discriminative reranking. In *ACL*, June 2005.

Cho, Kyunghyun, Bart van Merriënboer, Çağlar Gülçehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation. In *EMNLP*, 2014.

Chung, Junyoung, Kyunghyun Cho, and Yoshua Bengio. A Character-level Decoder without Explicit Segmentation for Neural Machine Translation. In *ACL*, 2016.

Dyer, Chris, Victor Chahuneau, and Noah A. Smith. A Simple, Fast, and Effective Reparameterization of IBM Model 2. In *Proceedings of the NAACL-HLT*, Atlanta, USA, June 2013.

Gojun, Anita and Alexander Fraser. Determining the Placement of German Verbs in English-to-German SMT. In *EACL*, 2012.

Jim, Kam, Bill G Horne, and C Lee Giles. Effects of noise on convergence and generalization in recurrent networks. In *NIPS*, 1994.

Kingma, Diederik P. and Jimmy Ba. Adam: A Method for Stochastic Optimization. *ICLR*, 2015.

Klein, Dan and Christopher D. Manning. Accurate Unlexicalized Parsing. In *ACL*, 2003.

Klein, Guillaume, Yoon Kim, Yuntian Deng, Jean Senellart, and Alexander M. Rush. OpenNMT: Open-Source Toolkit for Neural Machine Translation. 2017.

Koehn, Philipp, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondrej Bojar, Alexandra Constantin, and Evan Herbst. Moses: Open Source Toolkit for Statistical Machine Translation. In *ACL*, 2007.

Kummerfeld, Jonathan K., David Hall, James R. Curran, and Dan Klein. Parser Showdown at the Wall Street Corral: An Empirical Investigation of Error Types in Parser Output. In *EMNLP-CoNLL*, 2012.

Lee, Young-Suk, Bing Zhao, and Xiaoqiang Luo. Constituent Reordering and Syntax Models for English– to–Japanese Statistical Machine Translation. In *COLING*, 2010.

Levy, Roger and Galen Andrew. Tregex and Tsurgeon: tools for querying and manipulating tree data structures. In *LREC*, 2006.

Melamed, I. Dan, Ryan Green, and Joseph P. Turian. Precision and Recall of Machine Translation. In *NAACL-HLT*, 2003.

Nakagawa, Tetsuji. Efficient Top-Down BTG Parsing for Machine Translation Preordering. In *ACL-NLP*, 2015.

Och, Franz J. Minimum Error Rate Training in Statistical Machine Translation. In *ACL*, 2003.

Papineni, Kishore, Salim Roukos, Todd Ward, and Wei-Jing Zhu. BLEU: A Method for Automatic Evaluation of Machine Translation. In *ACL*, 2002.

Sennrich, Rico, Barry Haddow, and Alexandra Birch. Neural Machine Translation of Rare Words with Subword Units. In *ACL*, 2016.

Shterionov, Dimitar, Pat Nagle, Laura Casanellas, Riccardo Superbo, and Tony O'Dowd. Empirical Evaluation of NMT and PBSMT Quality for Large-scale Translation Production. In *EAMT*, 2017.

Snover, Matthew, Bonnie Dorr, Richard Schwartz, Linnea Micciulla, and John Makhoul. A study of translation edit rate with targeted human annotation. In *AMTA*, 2006.

Sutskever, Ilya, Oriol Vinyals, and Quoc V. Le. Sequence to Sequence Learning with Neural Networks. In *NIPS*, 2014.

Tange, Ole. GNU Parallel: The Command-Line Power Tool. *;login: The USENIX Magazine*, February 2011.

Wang, Chao, Michael Collins, and Philipp Koehn. Chinese Syntactic Reordering for Statistical Machine Translation. In *EMNLP*, 2007.

Wu, Peiyu. Word order errors in Simplified Chinese MT. *MultiLingual*, October/November 2016.

Xu, Peng, Jaeho Kang, Michael Ringgaard, and Franz J. Och. Using a Dependency Parser to Improve SMT for Subject-object-verb Languages. In *NAACL-HLT*, 2009.

Zhu, Muhua, Yue Zhang, Wenliang Chen, Min Zhang, and Jingbo Zhu. Fast and Accurate Shift-Reduce Constituent Parsing. In *ACL*, 2013.

Address for correspondence:
Anita Ramm
ramm@ims.uni-stuttgart.de
University of Stuttgart, Institute for Natural Language Processing,
Pfaffenwaldring 5b, 70569 Stuttgart, GERMANY

PBML

The Prague Bulletin of Mathematical Linguistics
NUMBER 108 JUNE 2017 73–84

Maintaining Sentiment Polarity in Translation of User-Generated Content

Pintu Lohar, Haithem Afli, Andy Way

ADAPT Centre, School of Computing, Dublin City University, Dublin, Ireland

Abstract

The advent of social media has shaken the very foundations of how we share information, with Twitter, Facebook, and Linkedin among many well-known social networking platforms that facilitate information generation and distribution. However, the maximum 140-character restriction in Twitter encourages users to (sometimes deliberately) write somewhat informally in most cases. As a result, machine translation (MT) of user-generated content (UGC) becomes much more difficult for such noisy texts. In addition to translation quality being affected, this phenomenon may also negatively impact sentiment preservation in the translation process. That is, a sentence with positive sentiment in the source language may be translated into a sentence with negative or neutral sentiment in the target language. In this paper, we analyse both sentiment preservation and MT quality *per se* in the context of UGC, focusing especially on whether sentiment classification helps improve sentiment preservation in MT of UGC. We build four different experimental setups for tweet translation (i) using a single MT model trained on the whole Twitter parallel corpus, (ii) using multiple MT models based on sentiment classification, (iii) using MT models including additional out-of-domain data, and (iv) adding MT models based on the phrase-table fill-up method to accompany the sentiment translation models with an aim of improving MT quality and at the same time maintaining sentiment polarity preservation. Our empirical evaluation shows that despite a slight deterioration in MT quality, our system significantly outperforms the Baseline MT system (without using sentiment classification) in terms of sentiment preservation. We also demonstrate that using an MT engine that conveys a sentiment different from that of the UGC can even worsen both the translation quality and sentiment preservation.

1. Introduction

The world of social media has experienced significant growth in the last decade. With the advent of Web 2.0, we are all publishers these days, which means that the

 Corresponding author: pintu.lohar@adaptcentre.ie
Cite as: Pintu Lohar, Haithem Afli, Andy Way. Maintaining Sentiment Polarity in Translation of User-Generated Content. The Prague Bulletin of Mathematical Linguistics No. 108, 2017, pp. 73–84.
doi: 10.1515/pralin-2017-0010.

amount of UGC created is enormous, multilingual, diverse and of varying quality. Accordingly, building robust, high-quality MT engines can be problematic, especially when users deliberately decide to violate linguistic norms in the languages they speak (cf. Jiang et al. (2012)). Twitter, one of the largest social media websites, enables people throughout the world to share information and express their opinion (in the form of tweets) in the language of their choice. Many Twitter users follow others who do not tweet in their preferred language. In such a case, tweets in a specific language need to be translated into the language of choice of such users. As well as the 140-character restriction mentioned above, tweets are often generated using mobile devices, which contributes further to the poor quality of language, including spelling and other errors, omission of diacritics etc. Tweets also contain hashtags, user handles, retweets etc., all of which makes tweet translation a difficult task. This task can be done directly (tweet-to-tweet), or indirectly via tweet normalization (Kaufmann and Kalita, 2010; Jiang et al., 2012).

Leaving quality *per se* to one side for one moment, errors in translation can negatively impact the sentiment of the source-language tweet, e.g. a tweet in English conveying positive sentiment may not retain its positivity after being translated into Japanese. Especially in business contexts, where large multinational companies want to find out what their users think of their products and services, sentiment preservation of the original tweets is arguably as important as the overall translation quality. Accordingly, in this work, we mainly focus on incorporating sentiment classification within our MT systems to investigate the extent to which the sentiment of tweets in the source language is preserved in the target language. Our aim is to improve sentiment preservation from source-to-target language tweets while at the same time minimizing any performance degradation in translation. We use parallel Twitter data set consisting of $4,000$ English tweets from the FIFA World Cup 2014 and there translations into German. We conduct four experiments on tweet translation: (i) a Baseline translation model built from the whole parallel corpus of tweets is used to translate the tweets, (ii) the data is divided according to specific sentiment classes to build different translation models for positive, negative and neutrally-sentimented tweets, (iii) the Twitter data is amalgamated with comparably much larger out-of-domain data sets,[1] and the sentiment translation model is combined with (a) small and (b) large out-of-domain models in order to apply phrase-table fill-up (Bisazza et al. (2011)).

The remainder of this paper is organized as follows. Section 2 highlights related work in this area. We describe our sentiment classification system in Section 3. Section 4 presents the different experiments, while Section 5 provides the empirical evaluation results, together with an analysis of our findings. Finally, we conclude and outline possible future work in Section 6.

[1]They *are* UGC, but the domains are not football-related.

2. Related Work

A significant amount of work has been done in the area of translation of UGC, and especially sentiment translation. The earliest work we are aware of is that of Kanayama et al. (2004), who use a transfer-based MT engine to translate text documents to a set of sentiment units. A graph-based approach using SimRank to transfer sentiment information from a source language to a target language is presented in Scheible et al. (2010). Saif et al. (2016) examine sentiment analysis in Arabic, a (relatively) resource-poor language. They use two approaches to examining the sentiment of Arabic social media posts: (i) translate the focus language text into a resource-rich language such as English, and apply a powerful English sentiment analysis system on the text, and (ii) translate resources such as sentiment-labeled corpora and sentiment lexicons from English into the focus language, and use them as additional resources in the focus-language sentiment-analysis system. They show that the sentiment analysis of English translations of Arabic texts produces competitive results, with respect to the Arabic sentiment analysis, and the Arabic sentiment analysis systems benefit from the use of automatically translated English sentiment lexicons. Balahur and Turchi (2012) deal with the problem of sentiment detection in three different languages (French, German and Spanish) using three distinct MT systems: Bing,[2] Google,[3] and Moses (Koehn et al., 2007). These systems are used to translate the *training* data so that English sentiment analysis can be applied to the output. In a similar vein, Araujo et al. (2016) show that simply translating the input text (the *test* data) from a specific language to English and then using one of the existing methods for English can be better than the existing language-specific efforts evaluated.

In parallel with the area of sentiment translation, crosslingual sentiment analysis (CLSA) has also undergone significant evolution. Lin et al. (2014) develops a model to carry out aspect-specific sentiment analysis in a target language using the knowledge learned from a source language. The task of crosslingual sentiment lexicon learning by automatically generating target-language sentiment lexicons from available English sentiment lexicons is addressed in Gao et al. (2015). Jain and Batra (2015) use the recursive auto-encoder architecture to develop a CLSA tool using sentence-aligned corpora between a resource-rich (English) and a resource-poor (Hindi) language. He et al. (2015) propose a semi-supervised learning approach with "space transfer" to tackle the task of cross-language sentiment classification. The work in Balahur and Turchi (2013) shows that the joint use of training data from multiple languages (especially those pertaining to the same family of languages) significantly improves the results of the sentiment classification. Baker et al. (2012) incorporate related aspects of meaning such as modality into the translation process in order to both maintain semantics across translation and improve translation quality. However, to the best

[2] https://www.bing.com/translator

[3] https://translate.google.com/

of our knowledge, none of the work to date has attempted a sentiment classification approach aimed at preserving the sentiment in translation. Our proposed method integrates the sentiment classification approach in building different translation models based on specific sentiment classes. Then the particular sentiment-translation model is used to translate the tweets with that sentiment polarity. This output is compared against a Baseline system built with all Twitter data, as well as systems based on phrase-table fill-up method.

3. Sentiment classification

3.1. Manual sentiment classification

We use a Twitter data set comprising 4,000 English tweets from the FIFA World Cup 2014 , their manual translations into German and the annotated sentiment scores (prepared by anon). As might be expected, these tweets are rather informal in nature e.g. the English tweet "GOAAAAL ♡ ♥ ♡ ♥" is translated as "TOOOOR ♡ ♥ ♡ ♥" in German in order to emphasize the positive emotion in the target language. We consider the tweets with manually annotated sentiment scores as our 'gold standard' data. The tweets are categorised into the following three classes: (i) negative tweets with sentiment score ≤ 0.4, (ii) neutral tweets with sentiment score $\simeq 0.5$ and (iii) positive tweets with sentiment score ≥ 0.6. Once the tweet categorization was complete, we held out a very small subset – 50 tweets per sentiment (negative, neutral and positive) – for tuning and testing purposes because we wanted to maintain as large an amount as possible for training the MT systems. For phrase-table fill-up, we include parallel sentence pairs from (i) an English–German parallel Flickr data set[4]o train a small out-of-domain model, and (ii) a much larger data set, namely the English-German parallel "News-Commentary" corpus[5] to build a large out-of-domain model. These data are also merged with the Twitter data to create additional training resources. The objectives here were to see the effects on both MT quality and sentiment preservation when the out-of-domain data is included.

The statistics of the number of parallel data used for training, tuning and testing is shown in Table 1. Of course, 3,700 training examples (tweets in this case) is not a large amount of data in the first place, and in our non-Baseline models we reduce this data size still further. Nonetheless, as will be seen in Section 4, good results *can* be achieved with such very small amounts of training data – albeit on admittedly small test sets – contrary to the perceived wisdom in the field.

The manually annotated sentiment scores are available only for the Twitter data because the Flickr and the News data are much larger, and so their manual annotation

[4] http://www.statmt.org/wmt16/multimodal-task.html#task1

[5] http://data.statmt.org/wmt16/translation-task/training-parallel-nc-v11.tgz

Data	Train	Development			Test		
		#negative	#neutral	#positive	#negative	#neutral	#positive
Twitter	3,700	50	50	50	50	50	50
Flickr	29,000	50	50	50	50	50	50
News_comm	235,843	50	50	50	50	50	50

Table 1: Data statistics

is practically infeasible. Therefore we apply an automatic sentiment analysis tool (see Section 3.2) to extract the sentiment scores for these data sets.

3.2. Automatic Sentiment classification

This approach involves automatic extraction of the sentiment scores of the tweets (or sentences) and their classification into negative, neutral and positive tweets (sentences) with the same criteria for scoring discussed in Section 3.1. We use a lexicon-based sentiment analysis (SA) system especially designed for tweets in low-resourced languages (Afli et al., 2017). This system makes use of SentiWordNet (Esuli and Sebastiani, 2006), an opinion lexicon derived from WordNet (Miller, 1995) where each word is associated with numerical scores (from zero to one) indicating the strength of being positive, negative or neutral. SentiWordNet word values have been semi-automatically computed based on training a set of ternary classifiers, each capable of deciding the polarity of the synset. The process begins with pre-processing of the raw tweets in following three modules: (i) **tokenization**: splitting the tweet into very simple tokens such as numbers, punctuation and words of different types; (ii) **sentence splitting**: segmenting the text into sentences, if there is more than one in the tweet. This module is required for the part of speech (PoS) tagger. (iii) **PoS tagging**: producing a PoS tag as an annotation on each word or symbol. Afterwards, SentiWordNet is used to score each PoS-tagged word in the tweet. Subsequently, exponential weighting and the words magnitude scoring techniques are applied on the tokenised and split text. Finally, in order to obtain the overall sentiment score of each tweet, the scores are added and normalized by the number of tweet words.

We evaluate the performance of the sentiment analysis tool of Afli et al. (2017) in classifying sentiments correctly. Out of the 4,000 tweets, 2,994 tweets are correctly classified when compared to the gold standard manual sentiment classification, giving a performance accuracy of 74.85%.

4. Experiments

4.1. Sentiment translation

We consider German as the source language and English as the target in order to be able to use the English SA tool for the English translation of the German tweets. For the Twitter data, we divide the train data of 3,700 tweet pairs into negative, neutral

Data	Sentiment Classification	#Negative	#Neutral	#Positive	#Total
Twitter	manual	919	1,308	1,473	3,700
Twitter	automatic	630	1,343	1,727	3,700
Flickr	automatic	9,677	11,065	8,258	29,000
News_comm	automatic	111,337	14,306	113,200	238,843

Table 2: Data distribution after sentiment classification

and positive tweet pairs using both the manual and automatic sentiment classification approaches. In contrast, since the manually annotated versions of Flickr and News data are unavailable, we apply the automatic SA tool on these data in order to extract the sentiment scores. Table 2 shows the distribution of negative, neutral and positive tweet/sentence pairs after manual and automatic sentiment classification.

In order to build the translation models, we use the Moses statistical MT (SMT) toolkit, which uses Giza++ (Och and Ney, 2003) for word and phrase alignment. The models are tuned using minimum error rate training (Och, 2003). Each of the translation models is built from the parallel data with a specific sentiment category and sentiment classification approach, respectively, e.g. a 'positive sentiment' translation model (see Table 2) is built from the 1,727 positive tweet pairs. The translation models conveying the particular sentiment types are referred to as "negative", "neutral" and "positive", respectively, whereas the single model trained on the whole 3,700-tweet pairs is termed the "Baseline". Note that our smallest system is built with just 630 tweet-pairs. Despite the fact that this may be the smallest SMT system ever published, as will be seen in Table 4, good results can nonetheless be achieved. The architectural overview of the sentiment translation system is shown in Figure 1. Note that the two boxes 'Output combination1/2' only merge the different polarity translations at tweet-level prior to the whole 150-tweet test set being sent for evaluation; there is no intention to suggest that parts of individual tweets are reassembled here into 'whole tweet' translations. More precisely, the whole 150-tweet test set is comprised of 50-tweets per sentiment class and each of them is translated using the corresponding translation model and the outputs are combined.

The main experiment consists of three different approaches; (i) translation without sentiment classification, (ii) translation with manual sentiment classification and (iii) translation with automatic sentiment classification which are discussed in the following sections. We also conduct experiments on data concatenation, and use phrase table fill-up method (Bisazza et al., 2011) to see whether it is possible to increase the translation quality and at the same time maintain the sentiment preservation.

4.1.1. **Translation without sentiment classification**

In this set-up, we build three translation models: (i) one with Twitter data, (ii) one with Twitter data combined with Flickr data, and (iii) one with Twitter, Flickr and News data combined.

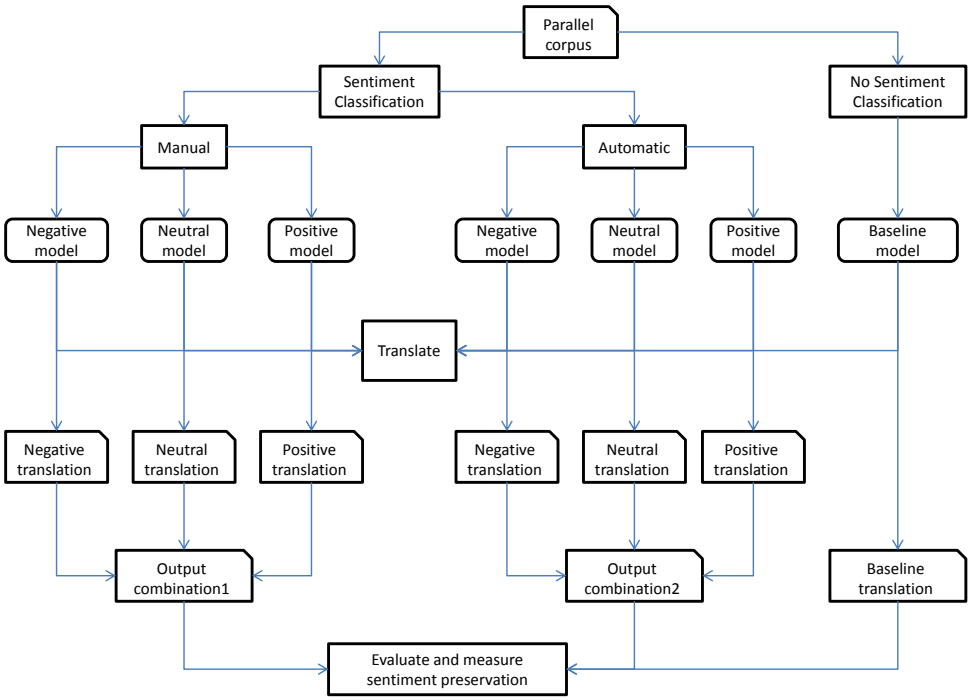


Figure 1: Architecture of the Sentiment Translation System

4.1.2. **Translation with manual sentiment classification**

In this approach, the three sentiment-translation models (with negative, neutral and positive sentiments) trained on the Twitter data with the gold standard sentiment annotations (the 'oracle', henceforth) translate the appropriate test set with the same sentiment polarity.

4.1.3. **Translation with automatic sentiment classification**

Here, we apply the SA tool to all the data sets and then train the sentiment-translation models under each sentiment class. This experiment is designed to test the expected fall off in accuracy with automatic sentiment classification.

In addition to this, we also make use of the phrase-table fill-up method using (i) one with Flickr data, and (ii) one with News data.

Translation model	Oracle	Sent_Clas.	BLEU	METEOR	TER	Sent_Pres.
Twitter	✓	✓	48.2	59.4	34.2	72.66%
Twitter	×	✓	48.1	58.9	34.6	68.0%
Twitter (Baseline)	✓	×	50.3	60.9	31.9	66.66%
Twitter + Flickr	×	✓	48.5	59.8	33.9	71.33%
Twitter + Flickr	×	×	50.7	62.0	31.3	62.66%
Twitter + Flickr + News_Comm	×	✓	50.3	62.3	31.0	**75.33%**
Twitter + Flickr + News_Comm	×	×	**52.0**∗	**63.4**∗	**30.1**∗	73.33%
Twitter (wrong MT engine)	✓	✓	46.9	57.9	35.4	47.33%

Table 3: Experimental evaluation: With data concatenation

5. Results

We conduct our experiments taking into account both the translation quality *per se* as well as the sentiment polarity preservation. The results are summarized in Table 3 which shows that, where only the Twitter data is used, the best BLEU, METEOR and TER scores are obtained when no sentiment classification (referred to as "Sent_Clas.") is applied ('Twitter (Baseline)'), i.e. when all Twitter data is merged, regardless of sentiment. The scores improve further when the Flickr data is used as additional training data, despite the fact that it is out-of-domain; when no sentiment classification is applied, the improvements here are 0.4, 1.1 and 0.6 BLEU, METEOR and TER points, respectively (see output rows 3 and 5 of Table 3). Moreover, further addition of out-of-domain News data produces the best BLEU, METEOR and TER scores of 52.0,63.4 and 30.1, respectively (row 7). We also perform statistical significance test with MultEval (Clark et al., 2011). The systems that perform significantly better than the Baseline with $p < 0.05$ are marked with "$*$".

However, we note that for Twitter data, the sentiment preservation score (termed as "Sent_Pres.") is higher when using the SMT systems in combination with the sentiment classification approach (72.66% for the Twitter oracle data). Without the oracle sentiment analysis, sentiment preservation dips to 68% (with sentiment classification), but when sentiment classification is switched off altogether in the Baseline model, the score is reduced further to only 66.66%. When the Flickr data is made available as additional training data, similar behaviour is seen; if we look at row 5, we can see that the sentiment preservation score is a full 10% less (a 16% relative reduction) than in row 1. When all the data merged together, using sentiment classification produces the highest sentiment preservation score of 75.33% (see row 6).

As might be expected, dividing an already tiny Twitter parallel corpus into different parts for translation model training causes a degradation in MT quality, but not by much: just 2.1 BLEU points compared to the Baseline (see row 1 and 3). When Flickr data is added, the BLEU, METEOR and TER scores decrease by 2.2, 2.2 and 2.6 points, respectively, but the sentiment preservation score increases by 8.67% (from 62.66% to 71.33%). When all data are concatenated, the BLEU, METEOR and TER scores decrease here too but the sentiment preservation score increases from 73.33% to 75.33%.

The last row in Table 3 shows that the wrong MT engines[6] produces the lowest MT evaluation and sentiment preservation scores. As is well-known, using the phrase-table fill-up method can improve MT quality, as this is used to plug the gaps of the smaller in-domain MT system (Bisazza et al., 2011). Accordingly, we conduct experiments with an aim to increasing the translation quality and observing any accompanying degradation in sentiment polarity preservation. The results are shown in Table 4. It can be observed that the scores remain similar (almost no improvement) in all cases. The probable reason is that the addition of Flickr and News data adds certainty in terms of the probabilities in the phrase-table in the data concatenation approach, which do not effectively carry over in the phrase-table fill-up method. However, the sentiment preservation scores decrease in both cases. Additionally, Table 5 shows some of the interesting results obtained. We can compare the translations generated by combining outputs by sentiment classification with the translations produced using the Baseline model. Example 1 (the reference) is a tweet with negative sentiment but both of the two systems fail to produce proper translation because the word "terrible" which is the main word representing negative emotion still remains untranslated in both cases. In general,

Data	Fill-up	BLEU	METEOR	TER	Sent_Pres.
Twitter	×	48.2	59.4	34.2	**72.66%**
Flickr	✓	48.0	59.0	34.4	69.33%
News_Comm	✓	48.4	59.4	34.3	71.33%

Table 4: Experiment evaluation using fill-up method

Ex.	Reference	sentiment translation models	Baseline model
1	*Howard Webb is a terrible ref #WorldCup*	*Howard Webb is a schrecklicher ref #WorldCup*	*Howard Webb is a schrecklicher ref #WorldCup*
2	*injured Neymar out of World Cup 2014*	*verletzter Neymar out the WC2014*	*verletzter Neymar out of World Cup 2014*
3	*penalty shootouts are too intense !*	*penalty shoot is to intensiv !*	*penalties is to intensiv !*
4	*damn chile is nice !!!! #WorldCup*	*freeking Chile is good !!! #WorldCup*	*damn Chile is good !!! #WorldCup*
5	*a bit boring ...*	*a little boring ...*	*some boring ...*
6	*im with Germany*	*I stand to Deutschlands side*	*I stand to Germany's side*
7	*as getting I, GO CHILE !*	*completely mache I it GO CHILE !*	*as getting I, GO CHILE !*

Table 5: Comparison of translations by sentiment translation models and Baseline model

the Baseline model produces better translations as compared to sentiment-specific models (see examples 2, 4, 6 and 7 in Table 5). However, there are few cases where

[6]We perform a test by translating (i) negative tweets by positive model, (ii) neutral tweets by negative model, and (iii) positive tweets by neutral model. However, any of the different combination can be applied; our objective is to arbitrarily choose one of them and investigate the effect on translation and change in sentiment polarity

the sentiment-classified models outperforms the Baseline model (examples 3 and 5). This is a very interesting observation that can motivate the application of sentiment classification approach towards improving not only the sentiment preservation but also the MT quality for particular texts. Finally, Table 6 shows some results on how

Ex.	Reference	Right MT engine	Wrong MT engine
1	*little break on the #WorldCup for an amazing #Wimbledon final!*	*small Pause from the #WorldCup for a amazing #Wimbledon final!*	*kleine Pause of the #WorldCup for a erstaunliches #Wimbledon final!*
2	*yes !!!!!*	*yes !!!!!*	*so !!!!!*
3	*a bit boring ...*	*a little boring ...*	*some was ...*

Table 6: Comparison between sentiment polarities using the right and wrong MT engine

the sentiment polarity can change by using wrong MT engines. The tweet in example 1 with positive sentiment, when translated by a wrong MT engine produces an incomprehensible translation that makes it very difficult to identify its sentiment polarity. Furthermore, for the tweets in examples 2 and 3, using wrong MT engines produces semantically very much different translation from the reference and can not be assigned either of the positive or negative sentiment. These results imply that it is essential to translate the tweets by the MT engines conveying the same sentiment.

6. Conclusion and Future Work

In this paper we investigated the performance of the sentiment classification approach in order to measure the MT quality and sentiment preservation for CLSA. We propose a strategy of dividing the data used to train the Baseline SMT system into different subsets based on specific sentiment categories – positive, negative and neutral – to build a suite of sentiment translation engines. We showed that, despite a small deterioration in translation quality, the sentiment classification approach significantly improves sentiment preservation. We would argue that this trade-off is well worth making, especially in industrial sectors where it is critical that user sentiment in one (less spoken) language is accurately rendered when translated into the language of choice (typically, English). Further experiments also suggest that it is essential to carefully select the proper MT engine conveying the same sentiment polarity as that of the UGC in order to improve the accuracy of sentiment polarity preservation in the target language. In future, we would like to make use of the SA tools for both the source and the target languages and then apply our proposed approach. Another possibility is to further refine the sentiment classes with additional sentiment categories (strong positive, strong negative etc.,) in order to build more specific translation models and combine their output for evaluation.

Acknowledgements

This research is supported by Science Foundation Ireland in the ADAPT Centre (Grant 13/RC/2106) (www.adaptcentre.ie) at Dublin City University.

Bibliography

Afli, Haithem, Sorcha McGuire, and Andy Way. Sentiment Translation for low resourced languages: Experiments on Irish General Election Tweets. In *18th International Conference on Computational Linguistics and Intelligent Text Processing*, Budapest, Hungary, 2017.

Araujo, Matheus, Julio Reis, Adriano Pereira, and Fabricio Benevenuto. An Evaluation of Machine Translation for Multilingual Sentence-level Sentiment Analysis. In *Proceedings of the 31st Annual ACM Symposium on Applied Computing*, pages 1140–1145, New York, USA, 2016.

Baker, Kathryn, Michael Bloodgood, Bonnie J. Dorr, Chris Callison-Burch, Nathaniel W. Filardo, Christine Piatko, Lori Levin, and Scott Miller. Modality and Negation in Simt Use of Modality and Negation in Semantically-informed Syntactic Mt. *Computational Linguistics*, 38(2):411–438, June 2012. ISSN 0891-2017.

Balahur, Alexandra and Marco Turchi. Multilingual Sentiment Analysis Using Machine Translation? In *Proceedings of the 3rd Workshop in Computational Approaches to Subjectivity and Sentiment Analysis*, pages 52–60, Jeju, Republic of Korea, 2012.

Balahur, Alexandra and Marco Turchi. Improving Sentiment Analysis in Twitter Using Multilingual Machine Translated Data. In *International Conference on Recent Advances in Natural Language Processing*, pages 49–55, Hissar, Bulgaria, 2013.

Bisazza, Arianna, Nick Ruiz, and Marcello Federico. Fill-up versus interpolation methods for phrase-based SMT adaptation. In *Proceedings of the International Workshop on Spoken Language Translation*, pages 136–143, San Francisco, USA, 2011.

Clark, Jonathan H., Chris Dyer, Alon Lavie, and Noah A. Smith. Better Hypothesis Testing for Statistical Machine Translation: Controlling for Optimizer Instability. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies:*, pages 176–181, Portland, USA, 2011. Association for Computational Linguistics.

Esuli, Andrea and Fabrizio Sebastiani. SENTIWORDNET: A Publicly Available Lexical Resource for Opinion Mining. In *In Proceedings of the 5th Conference on Language Resources and Evaluation*, pages 417–422, Genoa, Italy, 2006.

Gao, Dehong, Furu Wei, Wenjie Li, Xiaohua Liu, and Ming Zhou. Cross-lingual Sentiment Lexicon Learning with Bilingual Word Graph Label Propagation. *Computational Linguistics*, 41(1):21–40, Mar. 2015. ISSN 0891-2017.

He, Xiaonan, Hui Zhang, Wenhan Chao, and Deqing Wang. Semi-supervised Learning on Cross-Lingual Sentiment Analysis with Space Transfer. In *Proceedings of the IEEE First International Conference on Big Data Computing Service and Applications*, pages 371–377, Washington, DC, USA, 2015.

Jain, Sarthak and Shashank Batra. Cross Lingual Sentiment Analysis using Modified BRAE. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 159–168, Lisbon, Portugal, 2015.

Jiang, Jie, Andy Way, and Rejwanul Haque. Translating User-Generated Content in the Social Networking Space. In *Proceedings of the Tenth Biennial Conference of the Association for Machine Translation in the Americas*, pages 1–9, San Diego, USA, 2012.

Kanayama, Hiroshi, Nasukawa Tetsuya, and Watanabe Hideo. Deeper Sentiment Analysis Using Machine Translation Technology. In *Proceedings of the 20th International Conference on Computational Linguistics*, pages 494–500, Geneva, Switzerland, 2004.

Kaufmann, Max and Jugal Kalita. Syntactic normalization of Twitter messages. In *Proceedings of the 8th International Conference on Natural Language Processing*, pages 149–158, Kharagpur, India, 2010.

Koehn, Philipp, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondřej Bojar, Alexandra Constantin, and Evan Herbst. Moses: Open Source Toolkit for Statistical Machine Translation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics*, pages 177–180, Prague, Czech Republic, 2007.

Lin, Zheng, Xiaolong Jin, Xueke Xu, Weiping Wang, Xueqi Cheng, and Yuanzhuo Wang. A Cross-Lingual Joint Aspect/Sentiment Model for Sentiment Analysis. In *Proceedings of the 23rd ACM International Conference on Information and Knowledge Management*, pages 1089–1098, Shanghai, China, 2014.

Miller, George A. WordNet: A Lexical Database for English. *Journal of Communications of the ACM*, 38(11):39–41, November 1995. ISSN 0001-0782.

Och, Franz Josef. Minimum Error Rate Training in Statistical Machine Translation. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics*, pages 160–167, Sapporo, Japan, 2003.

Och, Franz Josef and Hermann Ney. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29:19–51, March 2003. ISSN 0891-2017.

Saif, Mohammad M., Mohammad Salameh, and Svetlana Kiritchenko. How Translation Alters Sentiment. *Journal of Artificial Intelligence Research*, 55(1):95–130, January 2016. ISSN 1076-9757.

Scheible, Christian, Florian Laws, Lukas Michelbacher, and Hinrich Schütze. Sentiment Translation Through Multi-edge Graphs. In *Proceedings of the 23rd International Conference on Computational Linguistics*, pages 1104–1112, Beijing, China, 2010.

Address for correspondence:
Pintu Lohar
pintu.lohar@adaptcentre.ie
ADAPT Centre, School of Computing, Dublin City University,
Glasnevin, Dublin 9,
Dublin, Ireland

PBML

The Prague Bulletin of Mathematical Linguistics
NUMBER 108 JUNE 2017 85–96

Using Word Embeddings to Enforce Document-Level Lexical Consistency in Machine Translation

Eva Martínez Garcia,[a] Carles Creus,[b] Cristina España-Bonet,[c] Lluís Màrquez[d]

[a] TALP Research Center, Universitat Politècnica de Catalunya
[b] Universitat Politècnica de Catalunya
[c] University of Saarland; DFKI, German Research Center for Artificial Intelligence
[d] ALT Group, Qatar Computing Research Institute, HBKU, Qatar Foundation

Abstract

We integrate new mechanisms in a document-level machine translation decoder to improve the lexical consistency of document translations. First, we develop a document-level feature designed to score the lexical consistency of a translation. This feature, which applies to words that have been translated into different forms within the document, uses word embeddings to measure the adequacy of each word translation given its context. Second, we extend the decoder with a new stochastic mechanism that, at translation time, allows to introduce changes in the translation oriented to improve its lexical consistency. We evaluate our system on English–Spanish document translation, and we conduct automatic and manual assessments of its quality. The automatic evaluation metrics, applied mainly at sentence level, do not reflect significant variations. On the contrary, the manual evaluation shows that the system dealing with lexical consistency is preferred over both a standard sentence-level and a standard document-level phrase-based MT systems.

1. Introduction

Statistical Machine Translation (SMT) systems have been traditionally designed at sentence level, without paying special attention to document-level information. However, taking into account some linguistic phenomena that go beyond the sentence boundaries, such as coreference or discourse markers, can be useful to improve the quality of the translation. Lexical coherence and consistency are also expected in a

 Corresponding author: emartinez@cs.upc.edu
Cite as: Eva Martínez Garcia, Carles Creus, Cristina España-Bonet, Lluís Màrquez. Using Word Embeddings to Enforce Document-Level Lexical Consistency in Machine Translation. The Prague Bulletin of Mathematical Linguistics No. 108, 2017, pp. 85–96. doi: 10.1515/pralin-2017-0011.

document, but they are difficult to attain if the document is translated in a sentence by sentence basis.

In this paper we focus on improving the quality of the translations by handling lexical selection consistency across sentences in the document. The hypothesis is that making translations more consistent will lead to more coherent documents, perceived as globally better translations by humans (Carpuat, 2009; Carpuat and Simard, 2012). We tackle this problem by integrating new mechanisms inside a document-level decoder based on Docent (Hardmeier et al., 2013), which evaluate lexical consistency at the document level, and which provide translation changes oriented to improve it.

First, we design and implement a new document-level feature. Our feature scores the document lexical consistency by measuring how suitable the translation of a word is according to its context and to the other possible translations for that word found within the document. The feature uses word embeddings to make these adequacy assessments.

Second, we design a new change operation affecting how the translation search space is explored by the document-level decoder. This operation guides the translation process to improve lexical consistency. In particular, our operation detects those words that present translation inconsistencies within a document and proposes alternative, consistent translations for them.

Finally, we evaluate our system on benchmark datasets for English to Spanish translation, comparing its results to a phrase-based MT system. Although the usual automatic MT evaluation metrics are mostly insensitive to the changes introduced by our document-based MT system, a manual evaluation conducted on the output shows that those changes are important and noticeable by humans when assessing the quality of the document translations.

2. Related Work

In recent years, several efforts have been devoted to deal with document-level translation. Usually, authors focus on a particular phenomenon, such as pronominal anaphora (Hardmeier and Federico, 2010; Nagard and Koehn, 2010), topic cohesion (Gong et al., 2011), or topic coherence (Xiong et al., 2015). Lexical consistency has also been addressed before. For instance, Xiao et al. (2011) and Martínez Garcia et al. (2014a) used a post-process to re-translate source words that have been translated in different ways in a document. This is similar to our work in the sense that they consider inconsistent terms to be those words translated in different ways throughout a document, but differs from ours in that we want to consider the consistency information at decoding time and not as a post-process. The way we measure the consistency also differs: we use (bilingual) distributed word representations for this purpose.

Distributed word representation or word embeddings (WE) models have been successfully applied to several different NLP tasks. An efficient implementation of the Context Bag of Words (CBOW) and the Skipgram algorithms is presented

in Mikolov et al. (2013a) and implemented in the word2vec toolkit. These models proved to be robust and powerful to predict semantic relations between words even across languages. However, they are unable to handle lexical ambiguity as they conflate word senses of polysemous words into one common representation. This limitation is discussed in Mikolov et al. (2013b) and Wolf et al. (2014), where bilingual extensions of the standard architecture are also proposed. Another bilingual approach is presented in Martínez Garcia et al. (2014b), where the resulting models are also evaluated in a cross-lingual lexical substitution task. Recently, WEs have been used in Pu et al. (2017) to improve the consistency of noun translations by means of a post-editing/re-ranking procedure with a phrase-based SMT system.

Closely related to our work, Hardmeier et al. (2012) used distributional vector models to implement semantic space language models (SSLM) within a document-oriented MT decoder. When working with SSLMs, the decoder uses the information of the word vector model to evaluate the adequacy of a word inside a translation by calculating the distance between the current word and its preceding context. In a similar way, Martínez Garcia et al. (2015) used, as SSLMs, bilingual and monolingual embedding models obtained with word2vec. Both studies used Docent (Hardmeier et al., 2013), a document-oriented SMT decoder that implements the algorithms in Hardmeier et al. (2012) and offers the possibility of using word embeddings as SSLMs. For our work, we use an in-house implementation of Hardmeier et al. (2012), named Lehrer as a homage to Docent.[1] These decoders work by performing hill climbing in a translation search space. This space can be seen as a graph where nodes are full-document translations and an edge connects two nodes when one translation can be transformed into the other. This transformation depends on the change operations provided by the decoders, which in general are simple operations such as changing the translation of a phrase, swapping phrase-pairs, or resegmenting the data. At each step of the search a full-document translation is available to the decoder. Thus, it is possible to develop features that capture properties of document-level phenomena. This makes these decoders flexible frameworks to develop and test different document-level strategies at translation time.

3. Lexical Consistency Feature

We strive to obtain translations where the same word appears translated into similar forms and with similar or related meanings throughout a document. In other terms, we want to avoid inconsistent translations for the same word. Thus, we are tackling a lexical-choice problem. Inspired by the SSLMs and with these aims, we develop a new lexical consistency feature that uses a Semantic Space to measure the Lexical Consistency of a document translation (SSLC).

[1] "*Lehrer*" means "teacher" in German. Source code at: `http://www.cs.upc.edu/~emartinez/lehrer.tgz`

Intuitively, SSLC scores each occurrence of an inconsistently translated source word with a value in $[-\infty, 0]$. For each such occurrence, this value is intended to measure how worse (in terms of adequacy) the current translation option is when compared to the other translation options seen in the document. More precisely, this value is computed as a subtraction between two numbers: the first one represents the adequacy of the current translation option, and the second one represents the best adequacy that could be obtained if another translation option (among the ones used somewhere in the document) had been used there instead. We consider a translation option to have better adequacy the more semantically similar it is to the context surrounding the occurrence being scored, and we compute it with WEs as the cosine similarity between the translation option and the context. Overall, note that SSLC does not try to enforce a strict lexically consistent translation, as long as lexical inconsistencies are semantically similar to their surrounding context.

To formalize SSLC we require some preliminary artillery. Let the source and target documents be the sequences of words $s_1, s_2, \ldots, s_N$ and $t_1, t_2, \ldots, t_M$, respectively, for some $N, M > 0$. Let $\tau : \{1, \ldots, N\} \rightarrow \{1, \ldots, M\}$ be a partial, injective mapping that associates to a source word index its corresponding target word index according to the current translation, if any.[2] In order to detect inconsistencies we need a way to identify whether two source or two target words must be considered to be the same word or not. To this end, we introduce the normalization functions $\mathtt{norm}_{src}$ and $\mathtt{norm}_{tgt}$ that take as input a source or target word, respectively, and return a normalized version of it. Then, two source or two target words are considered the same if they have the same normalized form through $\mathtt{norm}_{src}$ or $\mathtt{norm}_{tgt}$, respectively. In our experiments, $\mathtt{norm}_{src}$ and $\mathtt{norm}_{tgt}$ are implemented by, first, lower-casing the word and, second, by stemming it with the Snowball library.[3] Let $\mathtt{occ} : \{1, \ldots, N\} \rightarrow 2^{\{1,\ldots,N\}}$ be the function that associates to each source word index i the set of indexes of the source words that have the same normalized form as s_i, i.e., $\mathtt{occ}(i) = \{j \in \{1, \ldots, N\} \mid \mathtt{norm}_{src}(s_j) = \mathtt{norm}_{src}(s_i)\}$. Let $\tau\mathtt{occ}(i)$ be a shorthand for $\tau(\mathtt{occ}(i) \cap \mathtt{dom}(\tau))$, where the intersection with $\mathtt{dom}(\tau)$ is only necessary since τ is partial. We say that the ith source word is *inconsistent* in the current translation, denoted $\mathtt{incons}(i)$, if the source words s_j that have the same normalized form as s_i have been translated into more than two distinct normalized targets. Formally:

$$\mathtt{incons}(i) = \left(\left|\{\mathtt{norm}_{tgt}(t_j) \mid j \in \tau\mathtt{occ}(i)\}\right| > 2\right)$$

Let μ be the mapping defined by the word vector model in use by the decoder, i.e., a function that maps a word to a vector in a certain space $\mathbb{R}^n$ for some $n > 0$. Let $C > 0$ be the size of the context to either side of the target word, possibly crossing sentence

[2]Recall that phrase-based decoders perform translations by, in particular, using arbitrary alignments from source words to target words. For the SSLC feature we consider only the one-to-one word alignments.

[3]http://snowballstem.org/

boundaries. We tried several values for C and decided to fix $C = 15$ in the experiments as a good trade-off between performance and results. For each target word index $j \in \tau\mathrm{occ}(i)$, where the source word index i satisfies that $\mathrm{incons}(i)$ is true, we define its associated *score*, denoted $\mathrm{score}(j)$, depending on the cosine similarity between the context and the current used translation option, and the cosine similarity between the context and the other translation options in the document. More precisely:

$$\mathrm{score}(j) = \mathrm{sim}(\mathrm{ctxt}(j), \mu(t_j)) - \max_{k \in \tau\mathrm{occ}(i)} \mathrm{sim}(\mathrm{ctxt}(j), \mu(t_k))$$

where $\mathrm{ctxt}(j)$ is the sum of the vector representations of the words in the context of the jth target word, i.e., $\mathrm{ctxt}(j) = \sum_{k \in \{\max(1, j-C), \ldots, \min(j+C, M)\} \setminus \{j\}} \mu(t_k)$, and sim of two vectors is the natural logarithm of their cosine similarity linearly scaled to the range $[0, 1]$, i.e., $\mathrm{sim}(\vec{a}, \vec{b}) = \ln\big((\vec{a} \cdot \vec{b}/(\|\vec{a}\|\|\vec{b}\|) + 1)/2\big)$. Note that sim ranges in $[-\infty, 0]$, with $-\infty$ corresponding to the case where the vectors are diametrically opposed (semantically distant) and 0 to the case where they have the same orientation (semantically close). The final SSLC score for the whole document simply adds together the individual scores: $\sum_{i \in \mathrm{dom}(\tau),\ \mathrm{incons}(i)} \mathrm{score}(\tau(i))$.

As a final remark, note that for ease of presentation we have assumed that the word vector model is monolingual. If it were bilingual, the expressions like $\mu(t_j)$ would be $\mu(t_j, s_{\tau^{-1}(j)})$ instead. Also, unknown words for the vector model, i.e., words w such that $\mu(w)$ is undefined, are ignored when computing the scores, and not taken into account when considering the C-sized context of the target word.

4. Lexical Consistency Change Operation

Recall that the decoding process of Lehrer performs a hill climbing in a translation search space. At each step, the decoder explores the neighbourhood of the current translation by randomly applying to it one of the available change operations. The default operations perform simple modifications such as changing the translation of a phrase, swapping phrase-pairs, or resegmenting the data. Unfortunately, these simple operations do not aid in our goal of reaching more lexically consistent translations. The reason for this fact is twofold. On the one hand, to increase the consistency it is in general necessary to perform multiple changes within the document and, since the default change operations only perform one change at a time, it would take several steps to fix one of the lexical choice inconsistencies. On the other hand, since hill climbing only performs a step when it strictly increases the score, each of the intermediate steps that try to fix an inconsistency would need to increase the score. To ameliorate this limitation on the hill climbing, we introduce the Lexical Consistency Change Operation (LCCO) that shortcuts the process by, at a single step, performing simultaneous changes that fix inconsistent translations of the same source word.

Intuitively, LCCO randomly selects an inconsistently translated source word, randomly chooses one of its translation options used in the document, and re-translates

its occurrences throughout the document to match the chosen translation option. Both random decisions follow uniform distributions (the first one is uniform on all the distinct source words that appear inconsistently translated in the document, and the second one is uniform on all the distinct translation options seen in the document for the selected source word) in order to allow the hill climbing to fully explore the neighbourhood (given enough time) while minimizing the repetition of failed steps.

To formalize LCCO we need a more refined view of the source and target documents than in Section 3. Nevertheless, we reuse some of the previous definitions. Since the decoder works at phrase level, the documents are processed as sequences of phrases. Hence, we now consider that all the s_i and t_j are phrases instead of words. The definition of τ is still the same (although we can now guarantee that it is a total bijection since the decoder works with phrase-pairs) and $\mathit{norm}_{src}, \mathit{norm}_{tgt}$ are similar to before but have phrases as input and output instead of single words. The goal of LCCO is to change the translation of inconsistently translated words but, since the decoder works at phrase level, it can only change them safely when the inconsistent word appears alone in a phrase (otherwise LCCO would need to resegment the data too). For this reason, let us consider a more restricted definition of occ that only deals with indexes of source phrases having a single word. That is, for any $i \in \{1, \ldots, N\}$:

$$\mathit{occ}(i) = \{j \in \{1, \ldots, N\} \mid \mathit{norm}_{src}(s_j) = \mathit{norm}_{src}(s_i) \wedge |s_j| = 1\}$$

where $|s_j|$ is the number of words in the source phrase s_j. Using this redefined occ, we can keep the same definition for $\tau\mathit{occ}$ and incons as before.

LCCO works as follows. First, it selects a source phrase index $i \in \{1, \ldots, N\}$ such that $\mathit{incons}(i)$ is true. This is done by uniformly drawing that i from $\{\min(\mathit{occ}(k)) \mid k \in \{1, \ldots, N\} \wedge \mathit{incons}(k)\}$, where min is used to pick a representative from $\mathit{occ}(k)$. Second, it selects an occurrence $j \in \mathit{occ}(i)$ of that source phrase and considers $t_{\tau(j)}$ as the translation to use in the other occurrences. This is done by uniformly drawing that j from $\{k \in \mathit{occ}(i) \mid \nexists k' \in \mathit{occ}(i) : (k' < k \wedge \mathit{norm}_{tgt}(t_{\tau(k')}) = \mathit{norm}_{tgt}(t_{\tau(k)}))\}$. The new document translation $t'_1, t'_2, \ldots, t'_M$ is obtained by setting for each $k \in \{1, \ldots, M\}$:

$$t'_k := \begin{cases} t_k & \text{if } k \notin \tau\mathit{occ}(i) \\ t_k & \text{if } k \in \tau\mathit{occ}(i) \wedge \mathit{norm}_{tgt}(t_k) = \mathit{norm}_{tgt}(t_{\tau(j)}) \\ t_k & \text{if } k \in \tau\mathit{occ}(i) \wedge \nexists t \in \rho(s_{\tau^{-1}(k)}) : \mathit{norm}_{tgt}(t) = \mathit{norm}_{tgt}(t_{\tau(j)}) \\ t & \text{otherwise, with random } t \in \rho(s_{\tau^{-1}(k)}),\ \mathit{norm}_{tgt}(t) = \mathit{norm}_{tgt}(t_{\tau(j)}) \end{cases}$$

where ρ maps a source phrase to the set of target phrases that are its possible translations according to the phrase table in use by the decoder. Note that we do not change all the target phrases in $\tau\mathit{occ}(i)$, as in some of them we might already have a phrase with the same normal form as $t_{\tau(j)}$ (second case of the definition) and in some others the phrase table might not contain any entry with the same normal form as $t_{\tau(j)}$ (third case). The third case would never arise if norm_{src} had been defined as the identity.

5. Experiments

We use as baselines a standard sentence-level SMT system based on Moses (Koehn et al., 2007) and our document-level Lehrer system implementing the algorithms in Hardmeier et al. (2012). We use the Europarl corpus (Koehn, 2005) for training an English to Spanish translation system, GIZA++ (Och and Ney, 2003) for word alignments, and the 5-gram language model described in Specia et al. (2013). We build monolingual and bilingual WEs as described in Martínez Garcia et al. (2014b, 2015) using the CBOW implementation in word2vec. We use newsCommentary2009 as development set and newsCommentary2010 as test set.

Weight optimization for the baseline Moses system is done with MERT (Och, 2003) against the BLEU metric (Papineni et al., 2002). The same weights are used for the baseline Lehrer system. Since automatic weight optimization for document-level features is not straightforward (Smith, 2015), we optimize the weights for the document-level features of non-baseline Lehrer system variants with manual grid searches.

We analyze the performance of 17 systems: the standard baseline Moses, 8 variants of Lehrer, and another 8 analogous variants of Lehrer+LCCO. More precisely, the first mentioned 8 system variants are: a baseline Lehrer system, three systems that implement the SSLMs within Lehrer using either the bilingual (+SSLMbi), the monolingual (+SSLMmo), or both (+SSLMbi&mo) embeddings, two systems implementing our SSLC feature within Lehrer using the bilingual embeddings (+SSLCbi) and its combination with the SSLM features (+SSLMbi&mo+SSLCbi), and finally, two more systems with the monolingual embeddings in SSLC (+SSLCmo) and its combination with the SSLMs (+SSLMbi&mo+SSLCmo). For Lehrer+LCCO, its 8 system variants are analogous and we denote them with equivalent names.

5.1. Automatic Evaluation

We use the Asiya toolkit (González et al., 2012) for automatic evaluation and include several lexical metrics (TER, BLEU, NIST, METEOR).

In Table 1 we show the performance of the systems. On the development set, results without LCCO show that bilingual information in SSLM appears to be more helpful than monolingual, but it also seems that both kinds of models can work together to improve the final system output. Looking at the scores of both SSLC systems, there are almost no noticeable improvements with respect to baseline Lehrer. The best results have been obtained combining all the information: bilingual and monolingual SSLMs with either of the SSLCs. When introducing LCCO, we observe roughly the same trends as before, except that combining SSLC and SSLM does not seem to provide the same benefit. On the test set we observe a similar behaviour, although differences among system scores are smaller. In this occasion both SSLC appear to improve the baseline Lehrer. Note that, in contrast with the trend observed on the development set, now both SSLC seem to work better alone than combined with SSLM.

	Development set				Test set			
System	**TER↓**	**BLEU↑**	**NIST↑**	**METEOR↑**	**TER↓**	**BLEU↑**	**NIST↑**	**METEOR↑**
Moses	58.28	24.27	6.826	46.84	53.70	27.52	7.323	50.02
Lehrer	58.34	24.28	6.820	46.92	53.78	27.58	7.313	50.08
+SSLMbi	58.08	24.35	6.845	46.93	**53.49**	27.60	**7.349**	**50.13**
+SSLMmo	58.28	24.27	6.827	46.89	53.70	27.57	7.319	50.07
+SSLMbi&mo	58.01	24.36	6.854	46.91	**53.49**	27.48	7.344	50.10
+SSLCbi	58.38	24.26	6.817	46.90	53.77	**27.61**	7.315	50.07
+SSLCmo	58.37	24.24	6.818	46.91	53.78	27.59	7.313	50.07
+SSLMbi&mo+SSLCbi	**57.99**	**24.39**	6.861	**46.95**	53.50	27.50	7.344	50.07
+SSLMbi&mo+SSLCmo	**57.99**	24.37	**6.863**	**46.95**	53.51	27.51	7.347	50.08
Lehrer+LCCO	58.36	24.27	6.819	46.92	53.77	27.57	7.308	50.07
+SSLMbi	58.04	**24.38**	6.849	**46.94**	53.45	**27.61**	7.352	50.14
+SSLMmo	58.29	24.27	6.825	46.91	53.71	27.58	7.320	50.09
+SSLMbi&mo	58.04	24.35	6.848	46.92	**53.43**	27.60	**7.355**	**50.15**
+SSLCbi	58.36	24.25	6.819	46.89	53.81	27.59	7.310	50.07
+SSLCmo	58.35	24.27	6.819	46.91	53.77	27.59	7.311	50.07
+SSLMbi&mo+SSLCbi	58.06	24.34	6.846	46.93	53.46	27.57	7.351	50.12
+SSLMbi&mo+SSLCmo	**58.03**	24.36	**6.851**	46.92	53.47	27.57	7.348	50.12

Table 1. Scores of the automatic evaluation of the systems.

As a general remark, the differences between most of the systems are not statistically significant.[4] Several causes contribute to this effect. On the one hand, a pairwise comparison of all the system outputs shows that the amount of different sentences is only between 8% and 42%. On the other hand, SSLC and LCCO deal with very sparse phenomena, and thus, they cannot have a huge impact on the automatic metrics. For instance, in average, LCCO is applied on 8% of the documents on the development and test sets, and in those cases it comprises between 4% and 9% of the total amount of change operation applications. Nevertheless, this does not necessarily hinder our goals, as consistent lexical selection improvements can also be introduced by the default change operations (although taking more search steps in decoding than LCCO, as the latter performs several modifications at once), which are boosted by SSLC.

These results make necessary a human evaluation of the translations, since we expect that the few changes induced by SSLC and LCCO will be appreciated by humans.

5.2. Human Evaluation

We carry out two distinct evaluation tasks. The first one tries to assess the quality of the different systems, working with and without LCCO. The second one is a small document-level evaluation task that compares the adequacy of the lexical choices between pairs of system variants that differ on whether they use LCCO or not.

For the first evaluation task, we select a common subset of sentences from the test set translated by the Moses system and by the 8 variants of the Lehrer system. More

[4] According to bootstrap resampling (Koehn, 2004) over BLEU and NIST metrics with a p-value of 0.05.

ID	System	1	2	3	4	5	6	7	8	9
1	Moses	-	39 / 39	**44 / 43**	35 / 45	38 / 48	37 / 41	**43 / 39**	36 / 47	40 / 46
2	Lehrer	39 / 39	-	28 / 32	24 / 28	37 / 40	11 / 14	**14 / 11**	35 / 45	34 / 44
3	+SSLMbi	43 / 44	**32 / 28**	-	**36 / 33**	34 / 34	33 / 34	**37 / 29**	23 / 34	23 / 34
4	+SSLMmo	**45 / 35**	**28 / 24**	33 / 36	-	31 / 35	**31 / 30**	**32 / 26**	27 / 38	26 / 39
5	+SSLMbi&mo	**48 / 38**	**40 / 37**	34 / 34	**35 / 31**	-	**42 / 36**	**44 / 36**	18 / 27	20 / 25
6	+SSLCbi	**41 / 37**	**14 / 11**	**34 / 33**	30 / 31	36 / 42	-	**13 / 8**	34 / 43	36 / 45
7	+SSLCmo	39 / 43	11 / 14	29 / 37	26 / 32	36 / 44	8 / 13	-	31 / 47	33 / 47
8	+SSLMbi&mo+SSLCbi	**47 / 36**	**45 / 35**	**34 / 23**	**38 / 27**	**27 / 18**	**43 / 34**	**47 / 31**	-	**21 / 18**
9	+SSLMbi&mo+SSLCmo	**46 / 40**	**44 / 34**	**34 / 23**	**39 / 26**	**25 / 20**	**45 / 36**	**47 / 33**	18 / 21	-

ID	System	1	2	3	4	5	6	7	8	9
1	Moses	-	**40 / 38**	44 / 45	39 / 43	41 / 49	36 / 40	39 / 40	40 / 46	**44 / 42**
2	Lehrer+LCCO	38 / 40	-	32 / 40	23 / 32	28 / 38	14 / 19	13 / 19	31 / 41	35 / 38
3	+SSLMbi	**45 / 44**	**40 / 32**	-	38 / 39	21 / 26	**40 / 36**	36 / 36	21 / 28	24 / 26
4	+SSLMmo	**43 / 39**	**32 / 23**	**39 / 38**	-	36 / 37	**31 / 27**	**32 / 26**	34 / 36	**37 / 36**
5	+SSLMbi&mo	**49 / 41**	**38 / 28**	**26 / 21**	**37 / 36**	-	**39 / 34**	**40 / 35**	18 / 24	22 / 23
6	+SSLCbi	**40 / 36**	**19 / 14**	36 / 40	27 / 31	34 / 39	-	**16 / 13**	35 / 40	**36 / 35**
7	+SSLCmo	**40 / 39**	**19 / 13**	36 / 36	26 / 32	35 / 40	13 / 16	-	37 / 44	37 / 37
8	+SSLMbi&mo+SSLCbi	**46 / 40**	**41 / 31**	**28 / 21**	**36 / 34**	**24 / 18**	**40 / 35**	**44 / 37**	-	**21 / 19**
9	+SSLMbi&mo+SSLCmo	42 / 44	**38 / 35**	**26 / 24**	36 / 37	**23 / 22**	35 / 36	37 / 37	19 / 21	-

Table 2. The two pairwise system comparisons done in the human evaluation. Each entry is the mean % of times a row system is evaluated better/worse than the column system.

precisely, we randomly choose 100 sentences with at least 5 and at most 30 words, and with at least 3 different translations among all the considered system outputs. We set up an evaluation environment where 3 native-Spanish annotators (including two of the authors) with a high English level have been asked to rank the output of all the systems for each of the 100 selected sentences, from best to worst general translation quality and with possible ties. System outputs were presented in random order to avoid system identification. The same evaluation procedure is also carried out with the 8 variants of Lehrer+LCCO. Table 2 shows the results obtained, where each entry of the table contains the mean number of times that the row system is better/worse than the column system according to the annotators, the remainder being ties. For the ranking with Lehrer variants, (pairs of) annotators agreed 70% of the time when ranking (pairs of) distinct outputs, and with Lehrer+LCCO variants, 72% of the time.

From the results in Table 2, we can say that Lehrer and Lehrer+LCCO are equivalent to Moses: they have a few ties, and either system is considered better than the other in roughly the same amount of cases. On the other hand, most non-baseline variants of Lehrer and Lehrer+LCCO seem to surpass Moses on wins. Translations from the systems including the combination of several features seem to be preferred in general; for instance, annotators prefer the combination SSLMbi&mo over SSLMbi or SSLMmo alone. Another interesting detail is that the SSLC systems seem analogous to the corresponding Lehrer and Lehrer+LCCO baselines, as they have many ties (although the SSLC systems have a slight advantage on wins). Also, SSLCbi and SSLCmo seem analogous, with SSLCbi having a slight win advantage over SSLCmo. This fact

source	[...] Due to the choice of the camera and the equipment, these **portraits** remember the classic photos. [...] The passion for the *portrait* led Bauer to repeat the idea [...]
reference	[...] Son **retratos** que, debido a la selección de la cámara y del material recuerdan la fotografía clásica. [...] La pasión por los *retratos* de Bauer le llevó a repetir la idea [...]
Moses	[...] Debido a la elección de la cámara y el equipo, estos **retratos** recordar el clásico fotos. [...] la pasión por el *cuadro* conducido Bauer a repetir la idea [...]
Lehrer+LCCO	[...] Debido a la elección de la cámara y el equipo, estos **retratos** recordar el clásico fotos. [...] la pasión por el *retrato* conducido Bauer a repetir la idea [...]

Table 3. Systems translation example with (in)consistent lexical choices.

shows that bilingual information has helped SSLC more than monolingual information. Both combinations of SSLMbi&mo with either of the SSLCs also seem analogous. As final remarks, the SSLMbi&mo+SSLCbi variants of Lehrer and Lehrer+LCCO systematically beat the other systems, and the non-baseline Lehrer and Lehrer+LCCO variants beat their respective baseline variant (except for Lehrer+SSLCmo).

The second, small evaluation task is a comparison between three system pairs with and without LCCO: the baseline, +SSLCbi, and +SSLMbi&mo+SSLCbi variants of Lehrer against the analogous variants of Lehrer+LCCO. We selected 10 documents with lexical changes introduced by LCCO, and asked an annotator to choose the translation with best lexical consistency and adequacy, given the source and two translated documents obtained by a system pair. The annotator preferred the translations of the variants with LCCO 60% of the time, and 20% of the time considered the translations of either system to have the same quality. So, systems with LCCO provided better translations according to the annotator regarding lexical consistency and adequacy.

To conclude, we provide in Table 3 a translation example from a news-piece about a photographer and his portraits work. Moses has not translated consistently an occurrence of the word *portrait* (the one in italics) which wrongly becomes *cuadro* (painting) instead of the correct choice *retrato*. Without LCCO, only the baseline, +SSLMbi, and both SSLC variants of Lehrer correctly produce *retrato* instead of *cuadro*. With LCCO, on the contrary, all the system variants are able to produce the consistent translation.

6. Conclusions

We have presented two new document-level strategies that aid MT systems in producing more coherent translations by improving the lexical consistency of the translations during the decoding process. In particular, we have developed a new document-level feature and change operation. The feature scores the lexical selection consistency of a translation document. To this end, it uses word embeddings to measure the adequacy of word translations given their context, computed on words that have been

translated in several different forms within a document. The change operation helps the decoder explore the translation search space by performing simultaneous lexical changes in a translation step. Since it is able to modify several words at a time, even across sentences, it boosts the process of correcting the lexical inconsistencies. Both the feature and the change operation are implemented within our Lehrer decoder.

Results show that, although differences among systems are not statistically significant for the automatic evaluation metrics, they are noticeable for human evaluators that prefer the outputs from the enhanced systems.

As future work, we plan to study the impact of applying SSLC at lemma and seme levels, and conduct thorougher evaluations. Additionally, we are interested in tackling the same phenomena when using neural machine translation systems (Cho et al., 2014). These systems have recently achieved state-of-the-art results; however most are designed at sentence-level, and thus far, only a handful of works have studied the impact of using context information (Wang et al., 2017; Jean et al., 2017).

Bibliography

Carpuat, M. One Translation Per Discourse. In *Proc. of the Workshop on Semantic Evaluations: Recent Achievements and Future Directions*, DEW '09, pages 19–27, 2009.

Carpuat, M. and M. Simard. The Trouble with SMT Consistency. In *Proc. of the 7th Workshop on Statistical Machine Translation, WMT@NAACL-HLT 2012*, pages 442–449, 2012.

Cho, K., B. van Merrienboer, D. Bahdanau, and Y. Bengio. On the Properties of Neural Machine Translation: Encoder-Decoder Approaches. In *Proc. of SSST-8*, pages 103–111, 2014.

Gong, Z., M. Zhang, and G. Zhou. Cache-based document-level statistical machine translation. In *Proc. of the 2011 Conference on Empirical Methods in NLP*, pages 909–919, 2011.

González, M., J. Giménez, and L. Màrquez. A Graphical Interface for MT Evaluation and Error Analysis. In *Proc. of the 50th ACL, System Demonstrations*, pages 139–144, 2012.

Hardmeier, C. and M. Federico. Modelling pronominal anaphora in statistical machine translation. In *Proc. of the 7th IWSLT*, pages 283–289, 2010.

Hardmeier, C., J. Nivre, and J. Tiedemann. Document-Wide Decoding for Phrase-Based Statistical Machine Translation. In *Proc. of EMNLP-CoNLL*, pages 1179–1190, 2012.

Hardmeier, C., S. Stymne, J. Tiedemann, and J. Nivre. Docent: A Document-Level Decoder for Phrase-Based Statistical Machine Translation. In *Proc. of the 51st ACL Conference*, pages 193–198, 2013.

Jean, S., S. Lauly, O. Firat, and K. Cho. Does Neural Machine Translation Benefit from Larger Context? *CoRR*, abs/1704.05135, 2017.

Koehn, P. Statistical Significance Tests for Machine Translation Evaluation. In *Proc. of EMNLP*, pages 388–395, 2004.

Koehn, P. Europarl: A Parallel Corpus for Statistical Machine Translation. In *Proc. of the MT Summit X*, pages 79–86, 2005.

Koehn, P., H. Hoang, A. Birch, C. Callison-Burch, M. Federico, N. Bertoldi, B. Cowan, W. Shen, C. Moran, R. Zens, C. Dyer, O. Bojar, A. Constantin, and E. Herbst. Moses: open source toolkit for statistical machine translation. In *Proc. of the 45th ACL*, pages 177–180, 2007.

Martínez Garcia, E., C. España-Bonet, and L. Màrquez. Document-level machine translation as a re-translation process. In *Procesamiento del Lenguaje Natural, Vol. 53*, pages 103–110, 2014a.

Martínez Garcia, E., C. España-Bonet, J. Tiedemann, and L. Màrquez. Word's Vector Representations meet Machine Translation. In *Proc. of SSST-8*, pages 132–134, 2014b.

Martínez Garcia, E., C. España-Bonet, and L. Màrquez. Document-Level Machine Translation with Word Vector Models. In *Proc. of the 18th EAMT*, pages 59–66, 2015.

Mikolov, T., K. Chen, G. Corrado, and J. Dean. Efficient Estimation of Word Representations in Vector Space. In *Proc. of Workshop at ICLR*, 2013a.

Mikolov, T., I. Sutskever, G. Corrado, and J. Dean. Distributed Representations of Words and Phrases and their Compositionality. In *Proc. of NIPS*, 2013b.

Nagard, R. Le and P. Koehn. Aiding pronouns translation with co-reference resolution. In *Proc. of Joint 5th Workshop on Statistical Machine Translation and MetricsMATR*, pages 252–261, 2010.

Och, F. Josef. Minimum Error Rate Training in Statistical Machine Translation. In *Proc. of the ACL 2003*, pages 160–167, 2003.

Och, F. Josef and H. Ney. A Systematic Comparison of Various Statistical Alignment Models. *Computational Linguistics*, 29(1):19–51, 2003.

Papineni, K., S. Roukos, T. Ward, and W.J. Zhu. BLEU: a Method for Automatic Evaluation of Machine Translation. In *Proc. of the 40th ACL*, pages 311–318, 2002.

Pu, X., L. Mascarell, and A. Popescu-Belis. Consistent Translation of Repeated Nouns using Syntactic and Semantic Cues. In *Proc. of the 15th EACL*, 2017.

Smith, A. BLEU Decoding and Feature Weight Tuning in Docent (Master Thesis). Uppsala Universitet, 2015.

Specia, L., K. Shah, J. G. C. de Souza, and T. Cohn. QuEst - A translation quality estimation framework. In *Proc. of ACL Demo Session*, 2013.

Wang, L., Z. Tu, A. Way, and Q. Liu. Exploiting Cross-Sentence Context for Neural Machine Translation. *CoRR*, abs/1704.04347, 2017.

Wolf, L., Y. Hanani, K. Bar, and N. Dershowitz. Joint word2vec Networks for Bilingual Semantic Representations. In *Poster sessions at CICLING*, 2014.

Xiao, T., J. Zhu, S. Yao, and H. Zhang. Document-level Consistency Verification in Machine Translation. In *Proc. of MT Summit XIII*, pages 131–138, 2011.

Xiong, D., M. Zhang, and X. Wang. Topic-Based Coherence Modeling for Statistical Machine Translation. In *IEEE/ACM Trans. on audio, speech & language processing*, pages 483–493, 2015.

Address for correspondence:
Eva Martínez Garcia
emartinez@cs.upc.edu
TALP Research Center – Universitat Politècnica de Catalunya
Jordi Girona, 1-3, 08034 Barcelona, Spain

PBML

The Prague Bulletin of Mathematical Linguistics
NUMBER 108 JUNE 2017 97–108

Comparative Human and Automatic Evaluation of Glass-Box and Black-Box Approaches to Interactive Translation Prediction

Daniel Torregrosa, Juan Antonio Pérez-Ortiz, Mikel L. Forcada

Departament de Llenguatges i Sistemes Informàtics, Universitat d'Alacant, E-03690 Sant Vicent del Raspeig, Spain

Abstract

Interactive translation prediction (ITP) is a modality of computer-aided translation that assists professional translators by offering context-based computer-generated continuation suggestions as they type. While most state-of-the-art ITP systems follow a glass-box approach, meaning that they are tightly coupled to an adapted machine translation system, a black-box approach which does not need access to the inner workings of the bilingual resources used to generate the suggestions has been recently proposed in the literature: this new approach allows new sources of bilingual information to be included almost seamlessly. In this paper, we compare for the first time the glass-box and the black-box approaches by means of an automatic evaluation of translation tasks between related languages such as English–Spanish and unrelated ones such as Arabic–English and English–Chinese, showing that, with our setup, 20%–50% of keystrokes could be saved using either method and that the black-box approach outperformed the glass-box one in five out of six scenarios operating under similar conditions. We also performed a preliminary human evaluation of English to Spanish translation for both approaches. On average, the evaluators saved 10% keystrokes and were 4% faster with the black-box approach, and saved 15% keystrokes and were 12% slower with the glass-box one; but they could have saved 51% and 69% keystrokes respectively if they had used all the compatible suggestions. Users felt the suggestions helped them to translate faster and easier. All the tools used to perform the evaluation are available as free/open–source software.

1. Introduction

Translation technologies such as machine translation (MT) (Hutchins and Somers, 1992) or translation memories (TM) (Somers, 2003) are frequently used by professional

 Corresponding author: dtorregrosa@dlsi.ua.es
Cite as: Daniel Torregrosa, Juan Antonio Pérez-Ortiz, Mikel L. Forcada. Comparative Human and Automatic Evaluation of Glass-Box and Black-Box Approaches to Interactive Translation Prediction. The Prague Bulletin of Mathematical Linguistics No. 108, 2017, pp. 97–108. doi: 10.1515/pralin-2017-0012.

translators to produce a first, usually inadequate suggestion of a target-language equivalent of a source-language sentence. The suggestion is then modified by the professional translator by rearranging or accepting parts of it, or by introducing new words when an appropriate equivalent fragment is not present; this can be perceived as a process in which the computer outputs the translation, and then the professional translator fixes the mistakes (if using MT) or the mismatches (if using TM). This paper focuses however on a different translation technology approach: *interactive translation prediction* (ITP), a human–computer collaborative approach in which computer-generated translation suggestions are offered as the professional translator carries out the translation of the source-language sentence.

The TransType project (Langlais et al., 2000), and its continuation, the TransType2 project (Macklovitch, 2006) were the pioneers of ITP. An automatic best-scenario evaluation with in-domain corpora (Barrachina et al., 2009) showed that it might theoretically be possible to save between 55% and 80% of the keystrokes in comparison with unassisted translation. A number of projects continued the research where TransType2 had left it off. Caitra (Koehn, 2009) is an ITP tool which uses both the phrase table and the decoder of a statistical machine translation (SMT) (Koehn, 2010) system to generate suggestions. Researchers at the Universitat Politècnica de València have also made significant improvements to ITP systems (Barrachina et al., 2009). The CASMACAT project (`casmacat.eu`) followed the same line of research, improving ITP using active and on-line learning (Alabau et al., 2014). More recent works use neural MT systems (NMT) to generate the suggestions, as the decoding procedure can easily be adapted to use a given prefix (Peris et al., 2016; Knowles and Koehn, 2016). All these systems follow a *glass-box* strategy: in the case of SMT, suggestions are obtained by means of a tightly coupled system that is modified or tailor-made to provide additional information such as word alignments, alternative translations, and scores or probabilities for the translation; NMT systems only need to be slightly modified. ITP systems can therefore exploit most (if not all) the information captured in the translation model to generate the ITP suggestions, but inherit common SMT and NMT requirements, such as their dependency on extensive parallel corpora. Integrating other resources (such as commercial, translation-as-a-service engines over which no control is available) as part of the ITP process would be almost impossible, as most of them would not be able to provide the additional information needed to generate the suggestions.

Unlike the previously described glass-box approach, Pérez-Ortiz et al.'s (2014) system follows a *black-box* strategy: suggestions are obtained by splitting the source-language sentence in all possible sub-segments up to a given number of words, querying any available bilingual resource capable of delivering one or more translations into the target language, and eventually offering some of these translated segments as suggestions as the translation is typed. These bilingual resources can be MT systems, but also translation memories, dictionaries, catalogs of bilingual phrases, or any combination of them. The performance of this approach has been explored using rule-

based MT systems (Pérez-Ortiz et al., 2014) and in-domain and out-of-domain SMT systems (Torregrosa et al., 2014); more recently, the performance of the method used for suggestion ranking and selection has been improved by replacing the heuristics used in the early black-box ITP papers (Pérez-Ortiz et al., 2014; Torregrosa et al., 2014) with a neural network working on a set of features extracted from the source sentence, from the current prefix of the target sentence, and from the sub-segments translated with the bilingual resources (Torregrosa et al., 2016). Black-box systems have no access to the internals of the bilingual resource and can only use an approximation of the knowledge contained in the resource by translating each word multiple times in different contexts, that is, as part of the different segments (this means more words are translated overall), but this allows the integration of new resources without modifying how the ITP system works; similarly, the resources used do not need to provide additional information or be modified in any way. This makes it possible to use any resource available to the professional translator in an almost seamless way.

ITP popularity is on the rise and some commercial translation memory systems already integrate some form of ITP as one of their basic features (such as SDL Trados AutoSuggest 2.0, `translationzone.com/products/trados-studio/autosuggest`), and new translation tools such as Lilt (Green et al., 2014) (`lilt.com`) focus on delivering glass-box ITP on a user-friendly computer-assisted translation (CAT) web tool.

A comparison between the glass-box and the black-box approaches is carried out for the first time in this paper, by performing both an extensive automatic evaluation and a preliminary human evaluation. We evaluate both approaches when translating between related language pairs, particularly English–Spanish, and between less related languages such as Arabic–English and Chinese–English. This will help us assess the validity of the approaches for translating between languages that do not share the same syntactical structure, that is, those exhibiting frequent crossed and long-range word-alignments.

The remainder of the paper is organized as follows. In Section 2 we introduce our experimental set-up, and describe the automatic evaluation along with the results. In Section 3 we describe the experimental set-up and the results of the human evaluation. Finally, in Section 4, we discuss the results and propose future lines of research.

2. Experimental setup

2.1. Software used

As glass-box ITP model we will use the free/open-source toolkit Thot (`daormar.github.io/thot`) (Ortiz-Martínez and Casacuberta, 2014), which provides SMT, and ITP as a particular case of SMT where the system is forced to constrain the translation to a given prefix. Thot's ITP generates a word graph with probabilities using a modified version of the SMT decoder, and searches for the most probable translation constrained by the already typed prefix according to the word graph; an error-

	In-domain			Development	Out-of-domain	
Thot	Test	-		Development	Train	-
Forecat	-	Train	Development	-		
Evaluation	Test	-				
Sentences	3 000	10 000	2 000	2 000	1 000 000†	Rest of sentences†

Table 1. Distribution of the corpora. The sentences follow the same order as in the original corpus, except for the sentences tagged with †, which are ordered according to the similarity score of the bitext domain adaptation procedure. The top 1 million sentences for the glass-box training set were selected after filtering with the preprocessing tools in Thot.

correction algorithm is used if the typed prefix is not in the word graph. As black-box ITP model we will use the free/open-source toolkit Forecat (Torregrosa et al., 2016) (`github.com/transducens/forecat`). Forecat creates a pool of suggestions by splitting the source sentence in all the sub-segments up to a given length L, then translating them using any available bilingual resource. A set of features extracted from the source sentence, from the current prefix of the target sentence, and from the translated sub-segments is used by a feedforward neural network to rank the viability of the suggestions that are compatible (if the last word of the already typed prefix is the prefix of a suggestion, the suggestion is compatible); the top M suggestions are then offered to the user. In order to perform a fair comparison unaffected by the quality of the translation models, Forecat will use the same Thot SMT system as bilingual resource for translating the sub-segments in our experiments.

2.2. Corpora and model training

Parts of the Arabic–English (ar–en), English–Chinese (en–zh) and English–Spanish (en–es) bitexts from the United Nations Parallel Corpus 1.0 (Ziemski et al., 2016) have been used to train Thot models and the Forecat neural network, as well as to provide a test set for the automatic evaluation. Due to processing resources and time limitations, we had to reduce the size of the corpora used to train Thot models; to this end, we used the bitext domain adaptation procedure described by Axelrod et al. (2011) as implemented in XenC (Rousseau, 2013). This technique minimizes the impact of reducing the size of the training set by keeping the sentences that are more similar to the ones in the test set. The distribution of the corpus is shown in Table 1.

Thot's training and development sets were lowercased to reduce data sparsity and tokenized; those sentence pairs that could hinder the training procedure, such as extremely long sentences (more than 80 words) or sentence pairs with disparate lengths, were removed using the preprocessing tools in Thot, as described in its manual (`daormar.github.io/thot/docsupport/thot_manual.pdf`); however, the Stanford Tokenizer (`nlp.stanford.edu/software/tokenizer.shtml`) was used for the tokeniza-

tion of Chinese, as Thot does not support this task. The Simplified Chinese corpus was transliterated to the corresponding sequences for the Pinyin input method using Python's `pinyin 0.4.0` (`pypi.python.org/pypi/pinyin`), as Simplified Chinese characters are seldom directly typed. Thot was compiled to use IBM2 alignment models, and the training procedure used the parameter values in the user manual; a trigram language model and a maximum phrase length of 10 tokens were used. The reader may refer to the paper by Ortiz-Martínez and Casacuberta (2014) for more information about Thot's architecture. The BLEU (Papineni et al., 2002) scores for the resulting models (computed using the Thot tookit over the evaluation set) are: 0.49 for en→es, 0.47 for es→en, 0.43 for en→ar, 0.33 for ar→en, 0.23 for en→zh and 0.19 for zh→en.[1]

The Forecat feedforward neural network had one unit per feature in the input layer, 128 units in a single hidden layer, all fully connected to the input layer, and a single output unit fully connected to the hidden layer; it has a relatively small number of parameters, in the order of magnitude of 10^4. The training was performed via backpropagation with a learning rate of 10^{-3}, using the mean squared error (MSE) as the error function to optimize and no momentum or regularization; each model was trained five times with different weight initializations, and the one that results in a lower MSE was used in both the automatic and human evaluations. The reader may refer to the paper by Torregrosa et al. (2016) for more information about Forecat's architecture and for a description of the features.[2]

2.3. Automatic evaluation

The automatic evaluation model is similar to the one described by Langlais et al. (2000). A reference translation T is provided to the automatic system, which proceeds to "type" it; after each character, the system evaluates all the suggestions offered and chooses the suggestion or suggestion prefix that locally saves the most keystrokes and exactly matches the following words in T. Accepted suggestions or prefixes need to be full-word translations: if the word of T currently being translated is "thesaurus", a suggestion "the" will not be accepted. Accepting a full suggestion costs one keystroke, and accepting a suggestion prefix costs one keystroke per word in the selected prefix plus one keystroke for accepting the prefix (simulating the behaviour of the interface the human translators use, as described in Section 3). In order to measure the performance, we use the keystroke ratio (KSR), the ratio between the actual number of keys pressed for typing the translation and the length of the translation in characters; lower KSR values mean the suggestions were more useful while typing T. The glass-box model always offers one suggestion that completes the translation, and the user

[1] Even though 1 million sentences are too few for SMT, each of the resulting models use around 6 GB of RAM when loaded into the ITP server, most of the 8 GB available in the system used for human evaluation.

[2] The specific feature that takes the value of the starting letter (f_{26} in the paper by Torregrosa et al. (2016)) of the suggestion has been reworked for the en→ar task: rather than using the English alphabet, it uses the Arabic one; all the diacritics of the starting letter of the suggestion are removed.

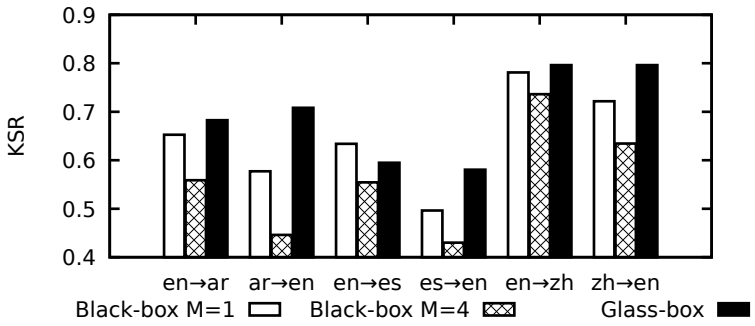


Figure 1. KSR values for the automatic evaluation. All differences between the values are statistically significant ($p \leq 0.05$).

can accept the full suggestion or a prefix of it; the suggestion will therefore be longer at the start of the task, and will shorten as the translation gets carried out. The average length over the 6 different translation tasks of the glass-box model suggestions offered during the automatic evaluation was of 20 words. The black-box model offers at most one suggestion ($M = 1$; if no suggestion is compatible with the typed prefix nothing is offered) with a maximum source sub-segment length of $L = 4$; the final length of the suggestion depends on the language pair and the words being translated. On average, the black-box model offered 2.3 words, or 1.4 words if we also consider the steps in which no suggestion is offered. The results obtained when allowing the black-box model to show up to 4 suggestions ($M = 4$) will also be shown, as this is the value used during the human evaluation; the black-box model with $M = 4$ shows on average 7.5 words (combining the length of the up to 4 suggestions), or 5 words if we also consider those steps where no suggestion is available. In both cases, the user can accept a full suggestion or a prefix of one of them.

2.4. Results of the automatic evaluation

We have performed extensive automatic evaluation for all six language pairs with both the black-box and the glass-box approaches, using all the sentences in the evaluation set described in 2.2. We tested the statistical significance of the results of the different models using paired bootstrap resampling (Koehn, 2004) with 1000 iterations and $p \leq 0.05$. The results of the automatic evaluation are shown in Figure 1. The black-box system using $M = 4$ outperformed the glass-box strategy by a wide margin, even when it had no access to all the information contained in the SMT system and, on average, showed less than half the words to the user as explained in the previous section; the black-box system with $M = 1$ still outperformed the glass-box system for every task but en→es, and showed on average less than a fourth of the words of

the glass-box approach. The black-box and glass-box approaches have closer performances when translating from English, as the corpora was originally written in English then translated; for en→es, the translation process is simpler and the glass-box method offers better suggestions.

3. Human evaluation

3.1. Experimental setup

We performed a human evaluation in order to compare the black-box and the glass-box approaches. To this end, both Forecat and Thot have been integrated into the open-source TM tool OmegaT (`omegat.org`) as plugins `github.com/dtr5/Forecat-OmegaT`, `github.com/dtr5/thot-omegat`). We used a preexistent plug-in to log user actions non-obstructively (`github.com/mespla/OmegaT-SessionLog`). No translation memory was used during testing. The suggestions (either the single sentence completion suggestion offered by the glass-box strategy or the up to $M = 4$ suggestions offered by the black-box strategy) are offered to the users in a drop-down list as they type the translation; these suggestions can then be accepted by selecting them using the arrow keys and pressing the enter key, by using a hot-key combination (`Alt+p`, with p being the position on the list) or with the mouse. Another hot-key (`Tab`) is used to select a prefix of the current suggestion, word by word. All the actions performed by the human translator, such as typing one character or selecting a full suggestion either with the mouse or the keyboard cost one keystroke, but selecting the prefix of a suggestion has a cost of one keystroke per word (`Tab` has to be pressed once per word) in the selected prefix plus one additional keystroke for accepting the prefix.

We have selected the first 20 English sentences with lengths between 15 and 25 words in the English–Spanish test set: this range of lengths excludes those sentences that are too long to be easily understood by non-native speakers and those so short that are hard to translate isolated from their context or do not present any kind of challenge to the translators. The sentences were arranged in 4 blocks SB_1–SB_4 of 5 sentences each, and the blocks were distributed so that each block was translated by two users under each modality. The 4 blocks were presented to the 8 users using 4 different modalities: the *induction task* let them familiarize with the interface and both suggestion models; the *unassisted task* offered no suggestions whatsoever; the *black-box task* used the black-box model, offering up to 4 suggestions ranked using the best neural network configuration, and the *glass-box task* used the glass-box model, offering a sentence-completion suggestion using the typed prefix as a constraint.

All eight test subjects U_1–U_8 were computer science researchers currently working in our university as technical or research staff. All of them except for U_5 claimed to be experienced typists. All of them are native Spanish speakers, and self-assessed themselves to have an R2/R2+ level (limited working proficiency) of English in the Interagency Language Roundtable scale for reading (a proficiency scale available

SB_1	Time	Tc	Tc/s	KS	KS/s	KSR	ESR
U_1							
U_2	996	666	0.67	996	1.00	1.50	–
U_3	**524**	**603**	**1.15**	**830**	**1.58**	**1.38**	**0.74**
U_4	*715*	*567*	*0.79*	*747*	*1.04*	*1.32*	*0.68*
U_5							
U_6	687	736	1.07	996	1.45	1.35	–
U_7	**468**	**604**	**1.29**	**583**	**1.25**	**0.97**	**0.76**
U_8	*602*	*581*	*0.97*	*717*	*1.19*	*1.23*	*0.70*

SB_2	Time	Tc	Tc/s	KS	KS/s	KSR	ESR
U_1	528	637	1.21	996	1.89	1.56	–
U_2	**626**	**636**	**1.02**	**686**	**1.10**	**1.08**	**0.71**
U_3	*576*	*570*	*0.99*	*537*	*0.93*	*0.94*	*0.75*
U_4							
U_5	477	677	1.42	690	1.45	1.02	–
U_6	**642**	**631**	**0.98**	**686**	**1.07**	**1.09**	**0.67**
U_7	*466*	*547*	*1.17*	*548*	*1.18*	*1.00*	*0.65*
U_8							

SB_3	Time	Tc	Tc/s	KS	KS/s	KSR	ESR
U_1	**613**	**677**	**1.10**	**686**	**1.12**	**1.01**	**0.62**
U_2	*732*	*618*	*0.84*	*819*	*1.12*	*1.33*	*0.68*
U_3							
U_4	668	606	0.91	782	1.17	1.29	–
U_5	**542**	**639**	**1.18**	**686**	**1.26**	**1.07**	**0.65**
U_6	*605*	*635*	*1.05*	*819*	*1.35*	*1.29*	*0.77*
U_7							
U_8	595	644	1.08	783	1.32	1.22	–

SB_4	Time	Tc	Tc/s	KS	KS/s	KSR	ESR
U_1	*513*	*615*	*1.20*	*819*	*1.60*	*1.33*	*0.49*
U_2							
U_3	298	646	2.17	765	2.57	1.18	–
U_4	**479**	**612**	**1.28**	**661**	**1.38**	**1.08**	**0.69**
U_5	*525*	*595*	*1.13*	*819*	*1.56*	*1.38*	*0.67*
U_6							
U_7	396	660	1.67	681	1.72	1.03	–
U_8	**392**	**647**	**1.65**	**807**	**2.06**	**1.25**	**0.66**

Table 2. Performance of the users with the different sentence blocks for the unassisted task (in regular typeface), the black box task (in bold) and the glass box task (in italics). The rows corresponding to the induction task are blank, as those results are not relevant.

at `govtilr.org/skills/ILRscale4.htm`). None of them had any kind of translation education or was familiar with the domain of the corpora. All of them resorted to using Google translate (`translate.google.com`) to look up the translation of single words or short phrases, except for U_1, who used the online version of the Cambridge English dictionary (`dictionary.cambridge.org`), and U_7, who preferred Linguee (`linguee.com`). Most users consulted domain-specific terms such as "guidelines", "compliance" or "interim". They were supervised during the test, and encouraged to ask as many questions as they needed to and experiment with the different suggestion systems, but only during the induction task. The instructions included all the ways they could use the suggestions and stressed that users were not obliged to accept one of the suggestions offered, but that they should also avoid ignoring them altogether.

3.2. Results of the human evaluation

We measured the time, the size in characters of the translations (Tc) and the number of keystrokes (KS), and calculated the translation speed (Tc/s), the number of keystrokes per second (KS/s) and the keystroke ratio (KSR=KS/Tc). We also calculated the emulated KSR (ESR) by performing the automatic evaluation described in Subsection 2.3 using the same conditions as the human test and the generated translations as references. The results of the human evaluation are shown in Table 2; an analysis of the differences in translation speeds and KSR of each method and user is shown in Figure 2. Only U_2 managed to translate both faster and with less effort

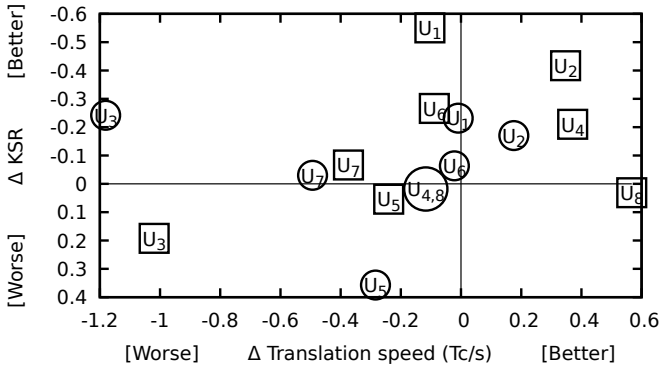


Figure 2. Absolute increase of KSR and Tc/s of the glass-box (○) and the black-box (□) tasks against the unassisted task. U_4 and U_8 got grouped because they attained very similar performances with the glass-box system.

with both techniques; U_4 managed to do so only with the black-box method. On average, when compared to the unassisted task, the evaluators saved 10% keystrokes and were 4% faster with the black-box approach, and saved 15% keystrokes and were 12% slower with the glass-box one; black-box suggestions proved therefore less useful but they allowed translators to perform faster. ESR values show that they could have theoretically saved 51% and 69% respectively if they had used the compatible suggestions. Users only had a few minutes to familiarize with the techniques, and it is expected that the translation speed will rise and the gap between the KSR and the ESR will close (but not completely, as part of this margin can be explained by user mistakes and rethought translations) as users get more and more familiar with the technology; a recent study by Autodesk (`langtech.autodesk.com/productivity.html`) considers experience the most single important factor in translation productivity.

After the tests, users were asked to sort the tasks according to their perceived speed of translation and ease of translation. U_1, U_4 and U_8 perceived the black-box system as faster and more helpful than the rest; the rest preferred the glass-box system; U_4 thought the glass-box system led to faster translations, yet it made the translation task harder than without assistance; finally, U_5 thought the black-box system made the task both harder and slower. Users' perceptions strongly contrasted with the measurements: only U_2 was faster with both methods compared to unassisted translation (0.67 Tc/s), though glass-box (0.84 Tc/s) was incorrectly perceived to be faster than black-box (1.02 Tc/s); and U_4 correctly ranked black-box (1.28 Tc/s) as the fastest task.

Finally, they were asked to provide some open feedback. U_4 strongly disliked the OmegaT tool. Most users were slightly annoyed by the unassisted block after experimenting with the induction block; some of them also said that the unassisted block had the harder sentences to translate, even when the sentences themselves were dif-

ferent from user to user. As none of them are professional translators, most of them expressed that the first full-sentence suggestion that the glass-box system gave them was very useful for planning the translation, but some complained those suggestions were too long and unwieldy. Some users complained about suggestions being offered too often, specially when none of them were useful. Some users praised the tool as they were able to operate it using only the keyboard; they all are experienced coders and most work in environments operable without a mouse. However, none of them used the `Alt+p` option for accepting specific suggestions from the drop-down list. The option for using the prefix of a suggestion by pressing `Tab` was neglected until they reached the glass-box block, as the suggestions were too long to be useful as a whole, but some had an adequate prefix.

4. Conclusions and future work

Interactive translation prediction (ITP) is a computer-assisted translation modality that focuses on offering translation suggestions as the translation is carried out. The automatic evaluation performed on this paper shows that 20%–50% of keystrokes can potentially be saved compared to unassisted translation using either the black-box or the glass-box approaches, regardless of whether the translation task is for related languages such as en–es or more unrelated ones such as ar–en or en–zh. The comparison between the black-box and the glass-box approaches shows that under these particular conditions, the black-box approach consistently outperforms the glass-box one in all but one translation task (en→es), even when the black-box approach does not have access to the internal information of the SMT model and shows to the user less than a fourth of the words offered by the glass-box model. Exhaustive analysis under different conditions needs to be carried out to identify when each system is useful and which one performs the best. Once these conditions are known, a hybrid strategy that chooses the best approach for each task could be devised. Also, even when the black-box strategy shows less words, we do not know the effect this has on the user; a detailed study about the cost of showing words and how many of them the users read before accepting or rejecting the suggestions has to be carried out.

In the human evaluation for en→es, test subjects mostly agreed in that both methods were useful, but were also divided when choosing which system was better for performing the translations; five of them preferred the glass-box approach and three preferred the black-box approach. Only one user managed to save keystrokes and be faster with both approaches. On average, the evaluators saved 10% keystrokes and were 4% faster with the black-box approach, and saved 15% keystrokes and were 12% slower with the glass-box one, but they could have saved 51% and 69% respectively if they used the compatible suggestions; as the users get more comfortable with the tool, the translation speed and the keystroke savings may both improve. Our preliminary human tests can be used to give an indication of how each system performs, but they suffer of two limitations: the size of the task and the profile of the users. A more

extensive evaluation with professional translators, translation students, or both will be carried out to explore the influence of different parameters and translation tasks. One common user complaint was that suggestions were being offered too often. Both models can be improved so they can assess the quality of the suggestions and offer only those that surpass some threshold. The detailed logs of the human evaluation sessions could also be used to tune the automatic evaluation strategies so they better reflect how users interacted with both approaches.

Finally, all the software used in this work is available under a free/open-source license. OmegaT users can now integrate both black-box and glass-box ITP and benefit from the performance improvements; using the plugins as inspiration, developers of other CAT tools can also integrate them into their tools.

Acknowledgments: Work partially funded by the Generalitat Valenciana through grant ACIF/2014/365, the Spanish government through project EFFORTUNE (TIN2015-69632-R), and by the Government of the Republic of Kazakhstan.

Bibliography

Alabau, Vicent, Jesús González-Rubio, Daniel Ortiz-Martínez, Germán Sanchis-Trilles, Francisco Casacuberta, M García-Martínez, Bartolome Mesa-Lao, Dan Cheung Petersen, Barbara Dragsted, and Michael Carl. Integrating online and active learning in a computer-assisted translation workbench. In *Proceedings of the First Workshop on Interactive and Adaptive Statistical Machine Translation, page to appear*, pages 1–8, 2014.

Axelrod, Amittai, Xiaodong He, and Jianfeng Gao. Domain adaptation via pseudo in-domain data selection. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 355–362. Association for Computational Linguistics, 2011.

Barrachina, Sergio, Oliver Bender, Francisco Casacuberta, Jorge Civera, Elsa Cubel, Shahram Khadivi, Antonio Lagarda, Hermann Ney, Jesús Tomás, Enrique Vidal, and Juan-Miguel Vilar. Statistical approaches to computer-assisted translation. *Computational Linguistics*, 35 (1):3–28, 2009.

Green, Spence, Jason Chuang, Jeffrey Heer, and Christopher D Manning. Predictive Translation Memory: A mixed-initiative system for human language translation. In *Proceedings of the 27th annual ACM Symposium on User Interface Software and Technology*, pages 177–187, 2014.

Hutchins, W. John and Harold L. Somers. *An introduction to machine translation*. Academic Press, 1992. ISBN 9780123628305.

Knowles, Rebecca and Philipp Koehn. Neural Interactive Translation Prediction. In *Proceedings of the Conference of the Association for Machine Translation in the Americas (AMTA)*, volume 1: MT researchers track, pages 107–120, 2016.

Koehn, Philipp. Statistical Significance Tests for Machine Translation Evaluation. In *Proceedings of the conference on Empirical Methods on Natural Language Processing (EMNLP 2004)*, pages 388–395, 2004.

Koehn, Philipp. A web-based interactive computer aided translation tool. In *Proceedings of the ACL-IJCNLP 2009 Software Demonstrations*, pages 17–20, 2009.

Koehn, Philipp. *Statistical Machine Translation*. Cambridge University Press, 2010. ISBN 0521874157, 9780521874151.

Langlais, Philippe, Sébastien Sauvé, George Foster, Elliott Macklovitch, and Guy Lapalme. Evaluation of TransType, a computer-aided translation typing system: a comparison of a theoretical-and a user-oriented evaluation procedures. In *Conference on Language Resources and Evaluation (LREC)*, pages 641–648, 2000.

Macklovitch, Elliott. TransType2: The last word. In *Proceedings of the 5th International Conference on Languages Resources and Evaluation (LREC 06)*, pages 167–172, 2006.

Ortiz-Martínez, Daniel and Francisco Casacuberta. The New Thot Toolkit for Fully Automatic and Interactive Statistical Machine Translation. In *Proc. of the European Association for Computational Linguistics (EACL): System Demonstrations*, pages 45–48, Gothenburg, Sweden, April 2014.

Papineni, Kishore, Salim Roukos, Todd Ward, and Wei-Jing Zhu. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318. Association for Computational Linguistics, 2002.

Pérez-Ortiz, Juan Antonio, Daniel Torregrosa, and Mikel L. Forcada. Black-box integration of heterogeneous bilingual resources into an interactive translation system. *EACL 2014 Workshop on Humans and Computer-assisted Translation*, pages 57–65, 2014.

Peris, Álvaro, Miguel Domingo, and Francisco Casacuberta. Interactive neural machine translation. *Computer Speech & Language*, 2016.

Rousseau, Anthony. Xenc: An open-source tool for data selection in natural language processing. *The Prague Bulletin of Mathematical Linguistics*, 100:73–82, 2013.

Somers, Harold L. *Computers and Translation: A Translator's Guide*. Benjamins translation library. John Benjamins Publishing Company, 2003. ISBN 9789027216403.

Torregrosa, Daniel, Mikel L. Forcada, and Juan A. Pérez-Ortiz. An open-source web-based tool for resource-agnostic interactive translation prediction. *The Prague Bulletin of Mathematical Linguistics*, 102(1):69–80, 2014.

Torregrosa, Daniel, Mikel L. Forcada, and Juan A. Pérez-Ortiz. Ranking suggestions for black-box interactive translation prediction systems with multilayer perceptrons. In *Proceedings of the Conference of the Association for Machine Translation in the Americas (AMTA)*, volume 1: MT researchers track, pages 65–78, 2016.

Ziemski, Michał, Marcin Junczys-Dowmunt, and Bruno Pouliquen. The United Nations Parallel Corpus v1.0. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016)*, pages 3530–3534, Paris, France, 2016. European Language Resources Association (ELRA). ISBN 978-2-9517408-9-1.

Address for correspondence:
Daniel Torregrosa
dtorregrosa@dlsi.ua.es
Universitat d'Alacant, E-03690 Sant Vicent del Raspeig, Spain

PBML

The Prague Bulletin of Mathematical Linguistics
NUMBER 108 JUNE 2017 109–120

Is Neural Machine Translation the New State of the Art?

Sheila Castilho,[a] Joss Moorkens,[a] Federico Gaspari,[a] Iacer Calixto,[a]
John Tinsley,[b] Andy Way[a]

[a] ADAPT Centre, Dublin City University
[b] Iconic Translation Machines

Abstract

This paper discusses neural machine translation (NMT), a new paradigm in the MT field, comparing the quality of NMT systems with statistical MT by describing three studies using automatic and human evaluation methods. Automatic evaluation results presented for NMT are very promising, however human evaluations show mixed results. We report increases in fluency but inconsistent results for adequacy and post-editing effort. NMT undoubtedly represents a step forward for the MT field, but one that the community should be careful not to oversell.

1. Introduction

Since its inception, different theories and practices for Machine Translation (MT) have come and gone, with each new wave generating great excitement and anticipation in the field. From the first commercial rule-based systems to more recent statistical models, there has, however, generally been great discrepancy between the high expectation of what MT should accomplish and what it is actually able to deliver. More recently, the neural approach (NMT) has emerged as a new paradigm in MT systems, raising interest in academia and industry by outperforming phrase-based statistical systems (PBSMT), based largely on impressive results in automatic evaluation (Bahdanau et al., 2015; Sennrich et al., 2016; Bojar et al., 2016). But do NMT results also surpass those of SMT when using human evaluation? Can we claim at this stage that NMT is the new state-of-the-art paradigm for production? This paper discusses the quality of NMT systems when compared to the state-of-the-art SMT

 Corresponding author: sheila.castilho@adaptcentre.ie
Cite as: Sheila Castilho, Joss Moorkens, Federico Gaspari, Iacer Calixto, John Tinsley, Andy Way. Is Neural Machine Translation the New State of the Art?. The Prague Bulletin of Mathematical Linguistics No. 108, 2017, pp. 109–120. doi: 10.1515/pralin-2017-0013.

systems, by reporting on three use cases in which human evaluators compared NMT and SMT output for a range of language pairs. Based on the findings, we argue that even though NMT shows significant improvements for some language pairs and specific domains, there is still much room for research and improvement before broad generalisations can be made.

The remainder of the paper is organised as follows: in Section 2, we survey the existing literature concerning NMT systems. In Section 3, we describe three use cases where NMT systems were compared against SMT systems and human evaluation was carried out: Section 3.1 presents a study using images to machine-translate user-generated e-commerce product listings with two NMT and one SMT systems for the English-German language pair; Section 3.2 reports a small-scale human evaluation focusing on the patent domain for the Chinese language, and Section 3.3 describes a large-scale human evaluation for the MOOC domain, considering translations from English into four target languages (German, Greek, Portuguese and Russian). Finally, in Section 4, we discuss the main findings of the use cases, zooming in on how NMT was evaluated, and we draw our main conclusions of interest to the broader MT community, including developers and users.

2. The Rise of Neural Machine Translation Models

Neural models involve building an end-to-end neural network that maps aligned bilingual texts which, given an input sentence X to be translated, is normally trained to maximise the probability of a target sequence Y without additional external linguistic information. Recently, a surge of interest in NMT came with the application of deep neural networks (DNNs) to build end-to-end *encoder–decoder* models (Kalchbrenner and Blunsom, 2013; Sutskever et al., 2014; Cho et al., 2014). Bahdanau et al. (2015) first introduced an attention mechanism into the NMT encoder–decoder framework which is trained to attend to the relevant source-language words as it generates each word of the target sentence. Some important recent developments in NMT involve improving the attention mechanism, including linguistic information or including more languages into the model (Luong et al., 2015; Sennrich and Haddow, 2016)

NMT improvements over PBSMT systems have been reported in shared tasks, where NMT ranked above SMT systems in six of 12 language pairs for translation tasks (Bojar et al., 2016). In addition, for the automatic post-editing task, neural end-to-end systems were found to represent a "significant step forward" over a basic statistical approach. Other recent studies have reported an increase in quality when comparing NMT with SMT using automatic metrics (Bahdanau et al., 2015; Jean et al., 2015) or small-scale human evaluations (Bentivogli et al., 2016; Wu et al., 2016). Wu et al. (2016) report their NMT system outperforming SMT approaches (for English to Spanish, French, simplified Chinese and back), particularly for morphologically rich languages, with impressive human evaluation ratings. Bentivogli et al. (2016) report that English-German NMT post-editing was reduced on average by 26% when

compared with the best-performing SMT system, with fewer word order, lexical, and morphological errors, concluding that NMT has "significantly pushed ahead the state of the art", particularly for morphologically rich languages.

Toral and Sánchez-Cartagena (2017) compare NMT and PBSMT for nine language pairs (English to and from Czech, German, Romanian, Russian, and English to Finnish), with engines trained for the WMT newstest data. Better automatic evaluation results are obtained for NMT output than for PBSMT output for all language pairs other than Russian-English and Romanian-English. NMT systems' increased reordering results in NMT systems performing better than SMT for inflection and reordering errors in all language pairs. However, they also report that SMT appears to perform better than NMT for segments longer than 40 words, when applying the chrF1 automatic evaluation metric (Popović, 2015).

This overview of recent work suggests that NMT has brought great improvement to the field, especially if one considers state-of-the-art automatic evaluation metrics. However, the progress is not always evident. Section 3 presents three use cases in which NMT was compared against SMT and evaluated via human assessments. What emerges is that depending on the different domains and on the various language pairs under study NMT has not always yielded the best results.

3. Use Cases

Each use case focuses on a different domain, and covers a different set of language pairs. First, Section 3.1 looks at NMT for e-commerce, describing important parts of a more extended study that is reported in detail in Calixto et al. (2017b). The second use case (Section 3.2) is an evaluation performed by Iconic Translation Machines Ltd.[1], whose goal was to find out whether NMT could provide better translations for the patent domain than SMT. Finally, the third and last use case (discussed in Section 3.3) is a comparison conducted as part of the EU-funded TraMOOC project on data taken from Massive Open Online Courses (MOOCs) in English.

3.1. NMT for E-Commerce Product Listing

A common use case in e-commerce consists in leveraging MT to make product descriptions, user reviews and comments (e.g. on dedicated forums) as widely accessible as possible, regardless of the customers' native language or country of origin. In previous work, Calixto et al. (2017a) compared the quality of product listings' translations obtained with a multi-modal NMT model against two text-only approaches: a conventional attention-based NMT and a PBSMT model. Translations were evaluated using automatic metrics as well as by means of a qualitative evaluation, whose final goal was to test whether training an NMT system with access to the product images improved the output quality for translations from English into German.

[1] http://iconictranslation.com/

MT Systems - Three different systems were compared in this experiment (1) a *PBSMT* baseline model built with the Moses SMT Toolkit (Koehn et al., 2007), (2) a text-only NMT model (NMT_t), and (3) a multi-modal NMT model (NMT_m), described in more detail in Calixto et al. (2017b), which expands upon the text-only attention-based model and introduces a *visual component* to incorporate *local* visual features.

The data set consists of product listings and images with 23,697 training tuples, each containing *(i)* a product listing in English, *(ii)* a product listing in German, and *(iii)* a product image. Validation and test sets have 480 and 444 tuples, respectively. One point to consider is that the translation of user-generated product listings poses particular challenges, for instance because they are often ungrammatical and can be difficult to interpret even by a native speaker of the language. In particular, the listings in both languages have many scattered keywords and/or phrases glued together, as well as a few typos. These are all complications that make the multi-modal MT of product listings a challenging task, as there are multiple difficulties associated with processing listings and images.

Evaluation - For the qualitative human evaluation, bilingual native German speakers were asked to (1) *assess the multi-modal adequacy* of translations (number of participants N=18); and (2) *rank* translations generated by different models from best to worst (number of participants N = 18). For the *multi-modal adequacy assessment*, participants were presented with an English product listing, a product image and a translation generated by one of the models, without knowing which model. They were then asked how much of the meaning of the source was also expressed in the translation, while taking the product image into consideration, using a 4-point Likert scale (where *4 = None of it* and *1 = All of it*). For the *ranking* assessment, participants were presented with a product image and three translations obtained from different models for a particular English product listing (without identifying the models) and were asked to rank translations from best to worst.

The automatic evaluation was performed with four widely adopted automatic MT metrics: BLEU4, METEOR (Denkowski and Lavie, 2014), TER (Snover et al., 2006), and chrF3.

Results - Table 1 contrasts some automatic metrics with human assessments of the adequacy of translations obtained with two text-only baselines, PBSMT and NMT_t, and one multi-modal model NMT_m.

Model	BLEU4↑	METEOR↑	TER↓	chrF3↑	Adequacy↓
NMT_t	22.5	40.0	58.0	56.7	$2.71 \pm .48$
NMT_m	$25.1^{\dagger}$	$42.6^{\dagger}$	$\mathbf{55.5}^{\dagger}$	58.6	$\mathbf{2.36} \pm .47$
PBSMT	$\mathbf{27.4}^{\dagger\ddagger}$	$\mathbf{45.8}^{\dagger\ddagger}$	$\mathbf{55.4}^{\dagger}$	**61.6**	$\mathbf{2.36} \pm .47$

Table 1. Adequacy of translations and four automatic metrics on product listings and images. For the first three metrics, results are significantly better than those of NMT_t (†) or NMT_m (‡) with $p < 0.01$.

The PBSMT model outperforms both the NMT models according to BLEU, METEOR and chrF3. However, there are no differences between the NMT_m model and the PBSMT according to TER scores.

Additionally, the adequacy scores for both these models, NMT_t and PBSMT, are on average the same according to scores computed over human assessments.

Nonetheless, even though both models are found to produce equally adequate output, translations obtained with PBSMT are ranked best by humans over 56.3% of the time, while translations obtained with the multi-modal model NMT_m are ranked best 24.8% of the time. These results suggest that although NMT models can sometimes reach PBSMT automatic MT scores, they are not preferred by human evaluators according to this use-case.

3.2. NMT for the Patent Domain

The evaluation presented in this section was based on a collaborative project carried out between the MT group at the ADAPT Centre, Dublin City University (Ireland), and Iconic Translation Machines Ltd. (Iconic), a commercial MT provider based in Dublin (Ireland). Iconic develops domain-specific MT engines for its users, frequently addressing language pairs and content types that pose great challenges for MT. One such combination in particular demand is Chinese patent information, for translation into English, with more that 100 million words machine translated in 2016.

The goal of this evaluation was to compare the performance between the mature Chinese to English patent MT engines used in production at Iconic with novel NMT engines developed at the ADAPT Centre on an 'apples to apples' basis, trained on the same available data.

The domain of evaluation was chemical patent titles and abstracts (see Table 2). This content type has particular characteristics that present challenges for MT, including very technical content with specialised terminology, names of chemical components, and alphanumeric and aminoacid sequences. The titles and abstract section of the patent themselves are quite distinct: titles are short, with 8.2 tokens on average, and are written in a formulaic telegraphic style; abstracts typically contain between 2-6 sentences that are quite long, with an average length of 42.5 tokens.

Description	Sentence Pairs	Words (source)
Chemical Abstracts	1,076,894	50,198,888
Chemical Titles	350,840	2,868,121
General Patent	11,931,127	324,222,969
Glossaries	1,575	1,575
Total	**13,358,861**	**377,291,553**

Table 2. Training data use for Iconic and NMT engine building

MT Systems - The Iconic MT engines are based on a proprietary Ensemble Architecture ^{TM} which combines elements of phrase-based, syntactic, and rule-driven MT, along with automatic post-editing. The engines have been highly tuned over a number of years for the patent domain, using multiple different translation and language models, and incorporate content-specific terminology.

The ADAPT/Iconic NMT engines were implemented using attention-based models built with Nematus[2] using various combinations of data (given there are slightly different domains, all data is used, i.e. just in-domain data, and in-domain plus different portions of the more general data chosen using data selection). We also tuned on different development sets for titles and abstracts. The four best development engines w for the evaluation. Both engines were trained using the same data, which included a mix of very content-specific in-domain data, more general patent data (including chemistry sub-domain) and technical glossaries.

Evaluation - Engines were evaluated separately on their performance on titles and abstracts, with two different test sets comprising 1,123 segments each. Standard automatic evaluation was carried out, and BLEU scores are reported in Table 3. Human evaluation was also carried out to compare the performance of the two engines. Two reviewers assessed 100 randomly selected segments from the aforementioned test sets in two ways: a blind ranking of the better translation (given a reference), and an error analysis to identify the main translation error in a given segment. The error taxonomy consisted of punctuation, part of speech, omission, addition, wrong terminology, literal translation, and word form. Segments were randomly selected from the test set, so that 25% of the segments were short sentences (i.e. they contained <10 words), 25% were long sentences (i.e. >40 words), and the remaining 50% were medium-length sentences (i.e. between 11 and 39 words).

Automatic evaluation results show that NMT slightly outperformed SMT on titles, whereas the SMT system outperformed NMT on abstracts. Regarding human evaluation, in general the SMT system was ranked 'best' 54% of the times, against 39% for NMT. When looking into sentence length, the SMT system was ranked 'best' 84% of the times for short sentences, against only 8% for the NMT system; and ranked best 58% of the times for long sentences (>40 tokens), against 33% for NMT. The NMT system was ranked 'best' more times than the SMT system only for medium-length sentences (>10<40 words), with 57% of preferences against 36% for SMT.

Results - Error types found in the NMT output were high for omission (37% of errors found in the segments against 8% for the SMT system), whereas for SMT the errors consisted of sentence structure (35% of the segments against 10% for the NMT system).

For segments free of errors, 25% of segments from the SMT system were found not to contain any errors, against only 2% of segments from the NMT system. These results indicate again that the NMT system surpasses the SMT one regarding automatic metrics (for the Titles), but human evaluation still prefers the SMT system.

System	**Titles** (BLEU)	**Abstracts** (BLEU)
Iconic MT	31.99	***28.32***
Neural MT	***37.52***	13.39

Table 3. Automatic MT evaluation results for chemical patent titles and abstracts.

[2] https://github.com/rsennrich/nematus

3.3. NMT for the MOOC domain

The evaluation presented in this section was conducted as part of the EU-funded TraMOOC (Translation for Massive Open Online Courses) project[3], which is a Horizon 2020 collaborative project aiming at providing reliable MT for MOOCs. A PB-SMT and an NMT system were compared across four translation directions (i.e. from English (EN) into German (DE), Greek (EL), Portuguese (PT), and Russian (RU) in a series of extensive assessment tasks. The goal of this comparison was to decide which system would provide better quality translations for the project domain.

MT Systems - The phrase-based SMT used was Moses, and the NMT systems were attentional encoder-decoder networks, which were trained with Nematus. The MT engines were trained on large amounts of training data from various sources: WMT training data[4] and OPUS[5], TED from WIT3[6], QCRI Educational Domain Corpus (QED)[7], a corpus of Coursera MOOCs, and the project's own collection of educational data. The amount of training data used is shown in Table 4.

As this evaluation was intended to identify the best-performing MT system for the translation of MOOCs, test sets were extracted from real MOOC data (one thousand English segments - for the ranking task, just one hundred segments were used). These data included explanatory texts, subtitles from video lectures, or user-generated content (UGC) from student forums or the comment sections of e-learning resources.

The UGC data was often poorly formulated and contained frequent grammatical errors. The other texts presented more standard grammar and syntax, but contained specialized terminology and non-contextual variables and formulae.

Target Language	DE	EL	PT	RU
Out-of-domain	23.78	30.73	31.97	21.30
In-domain	0.27	0.14	0.58	2.31

Table 4. Training data size for training MT engines for EN→ translation direction (number of sentence pairs, in millions).*

Evaluation - For the evaluation, automatic metrics were used (BLEU, METEOR and HTER (Snover et al., 2006)), and human evaluation was also performed. The human evaluation was performed by professional translators (three for EL, PT and RU, and two for DE) and consisted of: i) post-editing (PE) of the MT output to achieve publishable quality in the final revised text, ii) rating of fluency and adequacy (i.e.

[3] http://tramooc.eu/

[4] http://www.statmt.org/wmt16/

[5] http://opus.lingfil.uu.se/

[6] http://www.clg.ox.ac.uk/tedcorpus

[7] http://alt.qcri.org/resources/qedcorpus/

the extent to which a target segment reflects the meaning of the source segment) on a 4-point Likert scale for each segment, and iii) performing error annotation using a simple taxonomy (which included: inflectional morphology, word order, omission, addition, and mistranlsation).

Results - The automatic evaluation (see Table 3) showed that NMT outperformed SMT in terms of BLEU and METEOR scores for German, Greek and Russian (statistically significant in a one-way ANOVA pairwise comparison ($p<.05$)).

For Portuguese, only moderate improvements can be observed. The HTER scores show that more PE was required when using the output from the SMT system for all target languages (not statistically significant). These results indicate that when human intervention was considered (post-editing), the gain with NMT was less consistent.

Lang.	System	BLEU	METEOR	HTER	Fluency	Adequacy
DE	SMT	41.5	33.6	49.0	2.60	2.85
	NMT	61.2 †	42.7 †	32.2	2.95	2.79
EL	SMT	47.0	35.8	45.1	2.86	3.44
	NMT	56.6 †	40.1 †	38.0	3.08	3.46
PT	SMT	57.0	41.6	33.4	3.15	3.73
	NMT	59.9	43.4	31.6	3.22	3.79
RU	SMT	41.9	33.7	44.6	2.70	2.98
	NMT	57.3 †	40.65 †	33.9	3.08	3.12

Table 5. Automatic Evaluation Results (statistically significant results marked with †), Fluency and Adequacy

Human Evaluation - Regarding the human assessment of *fluency*, although no statistically significant differences were found, NMT was rated as more fluent than SMT for all language pairs (Table 5). Results for *adequacy* were less consistent, with higher mean scores for German SMT. These results show that as NMT gains in fluency, however, when assessing how much of the meaning expressed in the source appears in the translation, SMT is slightly better than or equal to NMT.

Regarding the *error annotation* task, the total number of issues identified in the output was greater for SMT than NMT for all language pairs.

Moreover, the number of segments without errors was greater for NMT across all language pairs. NMT output was also found to contain fewer word order errors and fewer inflectional morphology errors in all the target languages. However, SMT output contained fewer errors of omission, addition, or mistranslation for EN-EL than the NMT output; it also showed fewer omissions than the NMT system for EN-PT, while EN-RU SMT showed fewer mistranslations than the NMT system. Interestingly, for German, inflectional morphology errors make up 49% of all

Lang.	System	Technical Effort	Temporal Effort	WPS
DE	SMT	5.8	74.8	0.21
	NMT	3.9	72.8	0.22
EL	SMT	13.9	77.7	0.22
	NMT	12.5	70.4	0.24
PT	SMT	3.8	57.7	0.29
	NMT	3.6	55.19	0.30
RU	SMT	7.5	104.6	0.14
	NMT	7.2	105.6	0.14

Table 6. Technical (keystrokes/segment) and Temporal Post-Editing Effort (secs/segment) and words per second (WPS)

the errors found in NMT output, a higher proportion than for SMT (where inflectional morphology accounts for 43% of the errors). With respect to the *post-editing* tasks, results show that fewer NMT segments were considered by participants to require editing (but with statistical significance only for German ($p<.05$, where $M=.06$, $SE=.04$)). Average throughput or temporal effort (Table 6) was only marginally improved for German, Greek and Portuguese post-editing with NMT, while temporal effort for English-Russian was lower for SMT at the segment level. These results are also replicated in words per second (WPS).

Technical post-editing effort was reduced for NMT in all language pairs using measures of actual keystrokes (Table 6) or the minimum number of edits required to go from pre- to post-edited text (HTER in Table 5). Feedback from the participants indicated that they found NMT errors more difficult to identify, whereas word order errors and disfluencies requiring revision were detected faster in SMT output.

Finally, regarding the *ranking* task, the participants in the evaluation preferred NMT output across all language pairs, with a particularly marked preference for English-German. There was a 53% preference for NMT for short segments (20 tokens or fewer), and a 61% preference for NMT for long segments (over 20 tokens). In conclusion, for the language pairs under consideration (EN-DE, EN-EL, EN-PT and EN-RU) and for the specific MOOC domain, fluency was improved and word order errors decreased when using NMT. Fewer segments required post-editing when using NMT, especially due to the lower number of morphological errors. There was, however, no clear improvement with regard to omission and mistranslation errors when comparing SMT and NMT. There was also no great decrease in post-editing effort, suggesting that NMT for production may not as yet offer more than an incremental improvement in temporal post-editing effort.

4. Discussion and Conclusion

NMT has generated great hype, especially as the translation industry is eager for improved MT quality in order to minimise costs (Moorkens, 2017). Although promising results are being reported when comparing NMT with other MT paradigms using automatic metrics, when human evaluation is added to the comparison, the results are not yet so clear-cut. We have attempted to exemplify this statement with three use-cases comparing NMT against SMT systems where the evaluation was also performed by humans.

The results presented in Section 3.1 for translations of product listings show that NMT models are indeed very promising, especially considering that the state-of-the-art PBMST system has been deployed for quite some time, whereas the NMT models – especially the multimodal NMT system – have been developed over a shorter period of time. However, the PBSMT system still produces better translation when assessed both via automatic and human evaluation metrics. The same outcome can be observed in Section 3.2, with NMT models fast approaching SMT automatic scores

within a few months of deployment for the patent domain. It is important to notice that for both use cases 3.1 and 3.2, the training data is the same training data that is used in their everyday work, which makes it real-world results.

Finally, the extensive human evaluation described in Section 3.3 for the MOOC domain shows that NMT performs well in terms of automatic metrics (apart from Portuguese, where the improvement is only marginal), but is inconsistent for adequacy and post-editing effort. Even though the neural model demonstrates gains in fluency, it also shows a greater number of errors of omission, addition and mistranslation. The decision to move to the NMT model as the MT system of choice for the TraMOOC project reaffirms that neural models are very promising even though little time is put into their development when compared to long-standing PBSMT systems.

While automatic evaluation results published for NMT are undeniably exciting, so far it would appear that NMT has not fully reached the quality of SMT, based on human evaluation. We believe that the hype created in the MT field with the rise of the neural models must be treated cautiously. Overselling a technology that is still in need of more research may cause negativity about MT, as already seen before with SMT systems (especially with the release of the freely-available Moses toolkit in 2006, which made it easier for everyone to train their own MT system), when it was claimed that MT was producing 'near human quality' translations and that MT would 'steal translators' jobs', making translators 'merely post-editors of MT'. The hype that came with this euphoric presentation of SMT systems created a wave of discontent and suspicion among translators, that resulted in an 'us *versus* them' type of confrontation.

NMT no doubt represents a step forward for the MT field. However, there are also limitations for the neural models that cannot be overlooked and still need to be addressed. In our view, at this stage, researchers and industry need to be cautious not to promise too much, and allow for more research to address the limitations of NMT and more extensive human evaluations to be performed, addressing as many text types, domains and language pairs as possible.

Acknowledgements

The TraMOOC project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement № 644333. The ADAPT Centre for Digital Content Technology at Dublin City University is funded under the Science Foundation Ireland Research Centres Programme (Grant 13/RC/2106) and is co-funded under the European Regional Development Fund.

Bibliography

Bahdanau, Dzmitry, Kyunghyun Cho, and Yoshua Bengio. Neural Machine Translation by Jointly Learning to Align and Translate. In *International Conference on Learning Representations, ICLR 2015*, San Diego, California., 2015.

Bentivogli, Luisa, Arianna Bisazza, Mauro Cettolo, and Marcello Federico. Neural versus Phrase-Based Machine Translation Quality: a Case Study. *CoRR*, abs/1608.04631, 2016. URL http://arxiv.org/abs/1608.04631.

Bojar, Ondřej, Rajen Chatterjee, Christian Federmann, Yvette Graham, Barry Haddow, Matthias Huck, Antonio Jimeno Yepes, Philipp Koehn, Varvara Logacheva, Christof Monz, Matteo Negri, Aurelie Neveol, Mariana Neves, Martin Popel, Matt Post, Raphael Rubino, Carolina Scarton, Lucia Specia, Marco Turchi, Karin Verspoor, and Marcos Zampieri. Findings of the 2016 Conference on Machine Translation. In *Proceedings of the First Conference on Machine Translation*, pages 131–198, Berlin, Germany, August 2016. Association for Computational Linguistics.

Calixto, Iacer, Daniel Stein, Evgeny Matusov, Pintu Lohar, Sheila Castilho, and Andy Way. Using Images to Improve Machine-Translating E-Commerce Product Listings. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 637–643, Valencia, Spain, 2017a. URL http://www.aclweb.org/anthology/E17-2101.

Calixto, Iacer, Qun Liu, and Nick Campbell. Doubly-Attentive Decoder for Multi-modal Neural Machine Translation. In *Proceedings of the 55th Annual Meeting on Association for Computational Linguistics - Volume 1*, Vancouver, Canada (Paper Accepted), 2017b. URL https://arxiv.org/abs/1702.01287.

Cho, Kyunghyun, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734, Doha, Qatar, 2014. URL http://www.aclweb.org/anthology/D14-1179.

Denkowski, Michael and Alon Lavie. Meteor Universal: Language Specific Translation Evaluation for Any Target Language. In *Proceedings of the Ninth Workshop on Statistical Machine Translation*, pages 376–380, Baltimore, Maryland, USA, 2014. URL http://www.aclweb.org/anthology/W14-3348.

Jean, Sébastien, Kyunghyun Cho, Roland Memisevic, and Yoshua Bengio. On Using Very Large Target Vocabulary for Neural Machine Translation. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1–10, Beijing, China, 2015. URL http://www.aclweb.org/anthology/P15-1001.

Kalchbrenner, Nal and Phil Blunsom. Recurrent Continuous Translation Models. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, EMNLP 2013*, pages 1700–1709, Seattle, October 2013.

Koehn, Philipp, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondřej Bojar, Alexandra Constantin, and Evan Herbst. Moses: Open Source Toolkit for Statistical Machine Translation. In *Proceedings of the ACL-2007 Demo and Poster Sessions*, pages 177–180, Prague, Czech Republic, 2007. Association for Computational Linguistics.

Luong, Thang, Hieu Pham, and Christopher D. Manning. Effective Approaches to Attention-based Neural Machine Translation. In *Proceedings of the 2015 Conference on Empirical Methods*

in Natural Language Processing (EMNLP), pages 1412–1421, Lisbon, Portugal, 2015. ISBN 978-1-941643-32-7.

Moorkens, Joss. Under pressure: translation in times of austerity. *Perspectives*, 25(3):1–14, 2017. doi: 10.1080/0907676X.2017.1285331. URL http://dx.doi.org/10.1080/0907676X.2017.1285331.

Popović, Maja. chrF: character n-gram F-score for automatic MT evaluation. In *Proceedings of the Tenth Workshop on Statistical Machine Translation*, pages 392–395, Lisbon, Portugal, September 2015.

Sennrich, Rico and Barry Haddow. Linguistic Input Features Improve Neural Machine Translation. In *Proceedings of the First Conference on Machine Translation*, pages 83–91, Berlin, Germany, August 2016.

Sennrich, Rico, Barry Haddow, and Alexandra Birch. Edinburgh Neural Machine Translation Systems for WMT 16. In *Proceedings of the First Conference on Machine Translation (WMT16)*, Berlin, Germany, 2016.

Snover, Matthew, Bonnie Dorr, Richard Schwartz, Linnea Micciulla, and John Makhoul. A study of translation edit rate with targeted human annotation. In *Proceedings of association for machine translation in the Americas*, volume 200(6), 2006.

Sutskever, Ilya, Oriol Vinyals, and Quoc V Le. Sequence to Sequence Learning with Neural Networks. In *Advances in Neural Information Processing Systems*, pages 3104–3112, Montréal, Canada, 2014.

Toral, Antonio and Víctor M. Sánchez-Cartagena. A Multifaceted Evaluation of Neural versus Phrase-Based Machine Translation for 9 Language Directions. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 1063–1073, Valencia, Spain, April 2017. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/E17-1100.

Wu, Yonghui, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Lukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation. *CoRR*, abs/1609.08144, 2016.

Address for correspondence:
Sheila Castilho
sheila.castilho@adaptcentre.ie
ADAPT Centre, Dublin City University

PBML

The Prague Bulletin of Mathematical Linguistics
NUMBER 108 JUNE 2017 121–132

Fine-Grained Human Evaluation of Neural Versus Phrase-Based Machine Translation

Filip Klubička,[a] Antonio Toral,[b] Víctor M. Sánchez-Cartagena[c]

[a] University of Zagreb
[b] University of Groningen
[c] Prompsit Language Engineering

Abstract

We compare three approaches to statistical machine translation (pure phrase-based, factored phrase-based and neural) by performing a fine-grained manual evaluation via error annotation of the systems' outputs. The error types in our annotation are compliant with the multidimensional quality metrics (MQM), and the annotation is performed by two annotators. Inter-annotator agreement is high for such a task, and results show that the best performing system (neural) reduces the errors produced by the worst system (phrase-based) by 54%.

1. Introduction

A paradigm to machine translation (MT) based on deep neural networks and usually referred to as neural MT (NMT) has emerged in the past few years. This has disrupted the MT field as NMT, despite its infancy, has already surpassed the performance of phrase-based MT (PBMT), the mainstream approach to date.

We have witnessed the potential of NMT in terms of overall performance scores, be those automatic (e.g. BLEU) or human (e.g. system rankings); for example, in last year's news translation shared task at WMT.[1] There, out of 9 language directions where NMT systems were submitted, they significantly outperformed PBMT in 8, according to the human evaluation. In the remaining language direction (Russian-to-

[1] http://www.statmt.org/wmt16/translation-task.html

Corresponding author: fklubick@ffzg.hr
Cite as: Filip Klubička, Antonio Toral, Víctor M. Sánchez-Cartagena. Fine-Grained Human Evaluation of Neural Versus Phrase-Based Machine Translation. The Prague Bulletin of Mathematical Linguistics No. 108, 2017, pp. 121–132. doi: 10.1515/pralin-2017-0014.

English), the best PBMT submission was ranked higher than the best NMT system, but the difference was found not to be significant.

Given the impressive overall performance of NMT, some researchers have attempted in the past year to analyse the potential of NMT in a more detailed manner. The motivation comes from the fact that while overall scores give an indication of the general performance of a system, they do not provide any additional information. Hence, in order to delve further and try to shed light on the strengths and weaknesses of this new paradigm to MT, two recent papers have looked at conducting multifaceted evaluations.

- Bentivogli et al. (2016) conducted a detailed analysis for the English-to-German language direction where they compared state-of-the-art PBMT and NMT systems on transcribed speeches. They found out that NMT (i) decreases post-editing effort, (ii) degrades faster than PBMT with sentence length and (iii) improves notably on reordering and inflection.
- Toral and Sánchez-Cartagena (2017) carried out a series of analyses and evaluations for NMT and PBMT systems on the domain of news for 9 language pairs. They corroborated the findings of Bentivogli et al. (2016) with respect to NMT outstanding performance on reordering and inflection and its degradation with sentence length. They also contributed additional findings: NMT systems (i) exhibit higher inter-system variability, (ii) lead to more fluent outputs and (iii) perform more reordering than PBMT but less than hierarchical PBMT.

A limitation of these analyses lies in the fact that all of them were performed automatically. E.g. reordering and inflection errors were detected based on automatic evaluation metrics. Hence, one could argue that their outcomes are somewhat affected as automatic tools are, of course, never perfect.

In this paper we conduct a detailed human analysis of the outputs produced by NMT and PBMT systems. Namely, we annotate manually the errors found according to a detailed error taxonomy, that is compliant with the hierarchical listing of issue types defined as part of the Multidimensional Quality Metrics (MQM) (Lommel et al., 2014a). Specifically, we carry out this analysis for the news domain in the English-to-Croatian language direction. First, we define an error taxonomy that is relevant to the problematic linguistic phenomena of this language pair. Subsequently, we annotate the errors produced by 3 state-of-the-art translation systems that belong to the following paradigms: PBMT, factored PBMT and NMT. Finally, we analyse the annotations.

The main contributions of this paper can then be summarised as follows:

1. We conduct, to the best of our knowledge, the first human fine-grained error analysis of NMT in the literature.
2. We analyse NMT in comparison not only to pure PBMT and hierarchical PBMT, as in previous works, but also with respect to factored models.
3. We develop an MQM-compliant error taxonomy for Slavic languages.
4. We develop a novel approach to statistically analyzing and interpreting the results of MQM error annotation.

The rest of the paper is organised as follows. Section 2 describes the MT systems and the datasets used in our experiments. Section 3 covers the analysis, including the definition of the error taxonomy, the annotation setup and guidelines and finally the results obtained and their discussion. Finally, Section 4 outlines the conclusions and lines of future work.

2. MT Systems

This section describes the MT systems and the datasets used in our experiments. We built PBMT, factored PBMT and NMT systems.

The 3 systems were trained on the same parallel data. We considered a set of publicly available English–Croatian parallel corpora, comprising the DGT Translation Memory [2], HrEnWaC[3], JRC Acquis [4], OpenSubtitles 2013, SETimes and Ted talks. We concatenated all these corpora and performed cross-entropy based data selection (Moore and Lewis, 2010) using the development set. Once the data is ranked we keep the highest ranked 25% sentence pairs (4,786,516).

PBMT systems used also monolingual data for language modelling. To this end we used the concatenation of the hrWaC corpus (Ljubešić and Klubička, 2014) and the target side of the aforementioned parallel corpora.

As development set we used the first 1,000 sentences of the English test set used at the WMT12 news translation task[5], translated by a professional translator into Croatian. Similarly, our test set is made of the first 1,000 sentences of the English test set of the WMT13 translation task[6], again manually translated into Croatian.

The PBMT system was built with Moses v3.0[7]. In addition to the default models we also used hierarchical reordering (Galley and Manning, 2008), an operation sequence model (Durrani et al., 2011) and a bilingual neural language model (Devlin et al., 2014).

The factored PBMT system maps one factor in the source language (surface form) to two factors in the target (surface form and morphosyntactic description). This system is described in detail by Sánchez-Cartagena et al. (2016).

The NMT system is based on the sequence-to-sequence architecture with attention and we applied sub-word segmentation with byte pair encoding (Sennrich et al., 2015) jointly on the source and target languages. We performed 85 000 join operations. Training was run for 10 days and a model was saved every 4.5 hours. We decoded

[2] https://ec.europa.eu/jrc/en/language-technologies/dgt-translation-memory

[3] https://www.clarin.si/repository/xmlui/handle/11356/1058

[4] http://tinyurl.com/CroatianAcquis

[5] http://www.statmt.org/wmt12/translation-task.html

[6] http://www.statmt.org/wmt13/translation-task.html

[7] https://github.com/moses-smt/mosesdecoder/tree/RELEASE-3.0

the test set using an ensemble of 4 models. These were the 4 models with the highest BLEU scores on the development set.

2.1. Evaluation

We report the scores obtained in terms of the BLEU and TER automatic evaluation metrics for the 3 systems described in the previous section. Table 1 shows the results.

As the table shows, the use of factored models leads to a substantial improvement upon pure PBMT (6% relative in terms of BLEU). NMT, on its turn, allows us to obtain a further notable improvement; 14% relative in terms of BLEU compared to the factored PBMT system and 21% compared to the initial PBMT system.

System	BLEU	TER
PBMT	0.2544	0.6081
Factored PBMT	0.2700	0.5963
NMT	0.3085	0.5552

Table 1. Automatic evaluation (BLEU and TER scores) of the 3 MT systems

3. Error analysis

In this section we report on the motivation for conducting the manual error analysis, describe the framework and overall annotation process, and present the results.

The fact that Croatian is rich in inflection, has rather free word order and other similar phenomena that English does not, gives rise to specific translation issues. For example, grammatical categories that do not exist in English, like gender and case, may be particularly hard to generate reliably in a Croatian translation. We built our factored PBMT system aiming to directly address such issues. Similarly motivated, we wished to see how an NMT system would grapple with the same issues.

Indeed, as shown in Section 2, automatic evaluation shows significant improvement for both systems, compared to the pure PBMT system. However, as is the nature of automatic metrics, the automatic scoring methods do not indicate whether any of the linguistic problems mentioned earlier have been addressed by the systems. The question of whether the linguistic quality, or rather, grammaticality of the output is improved has not been answered by automatic evaluation. Are cases and gender handled better? Is there better agreement? Is the fluency of the translation higher?

In order to provide answers to these research questions, we decide to thoroughly compare these systems by systematically analyzing their outputs via manual error analysis. In this way we can obtain a more complete picture of what is happening in the translation, which can provide pointers on where to act to obtain further improvements in the future.

3.1. Multidimensional Quality Metrics and the Slavic tagset

After looking into different ways of performing the task of manual evaluation via error analysis, we decided to make use of the MQM framework, developed in the QT-Launchpad project[8]. This is a framework for describing and defining custom translation quality metrics. It provides a flexible vocabulary of quality issue types and a mechanism for applying them to generate quality scores. It does not impose a single metric for all uses, but rather provides a comprehensive catalog of quality issue types, with standardized names and definitions, that can be used to describe particular metrics for specific tasks.

The main reason we chose the MQM framework was the flexibility of the issue types and their granularity — it gave us a reliable methodology for quality assessment, that still allowed us to pick and choose which error tags we wish to use.

The MQM guidelines propose a great variety of tags on several annotation layers[9]. However, the full tagset is too comprehensive to be viable for any annotation task, so the process begins with choosing the tags to use in accordance to our research questions. Initially we started off with the core tagset, a default set of evaluation metrics (i.e. error categories) proposed by the MQM guidelines, as seen in Figure 1.

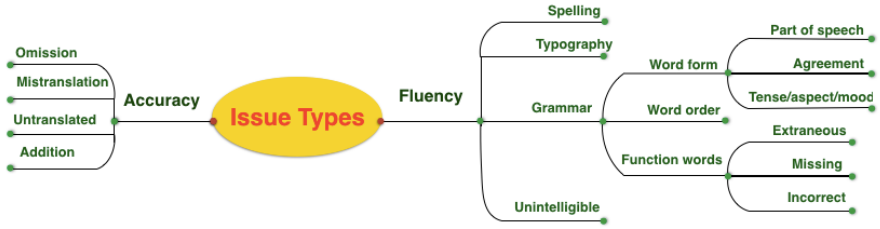


Figure 1. The core error categories proposed by the MQM guidelines

However, given the morphological complexity of Croatian and the level at which we made interventions in the system, we found that these core categories were not detailed enough, or rather, did not allow for an analysis of the specific phenomena we were interested in. Some categories that were of interest to us, like specific *Agreement* types, were not present in the tagset, while some errors, like *Typography*, were irrelevant to us. So we created our own set of tags by modifying the core set, rearranging the hierarchy, adding new tags and removing those that are of little relevance. We call this new tagset the Slavic tagset, as its expansion allows for the identification of

[8] http://www.qt21.eu/mqm-definition/definition-2015-06-16.html

[9] http://www.qt21.eu/mqm-definition/issues-list-2015-12-30.html

grammatical errors which are commonly shared by Slavic languages. This tagset is outlined in Figure 2.

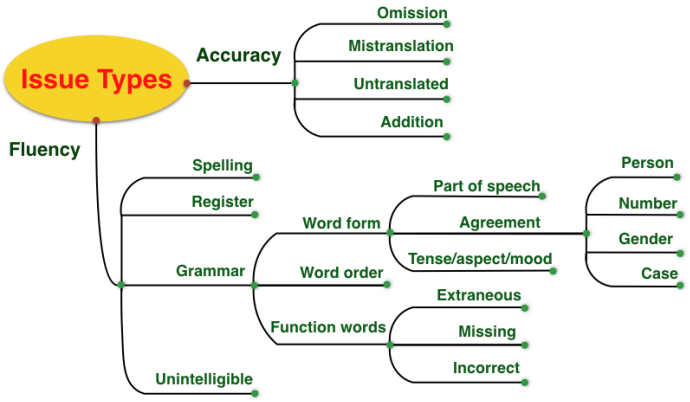


Figure 2. The Slavic tagset, a modified version of the MQM core tagset

3.2. Annotation setup

In order to carry out the annotations we used `translate5`[10], a web-based tool that implements annotations of MT outputs using hierarchical taxonomies, as is the case of MQM.

We had two annotators at our disposal, who both had prior experience with MQM as well as the same background - an MA in English linguistics and information science. They were thoroughly familiarized with the official annotation guidelines and the decision process[11] prior to annotation.

The annotators annotated 100 random sentences from the test set introduced in Section 2. These sentences were translated by all three MT systems, and the annotators were presented with the source text, a reference translation and the unannotated system outputs at the same time. All three translations were then annotated by both our annotators (i.e. each system translated the same 100 sentences, each annotator annotated the 300 translated sentences, making a total of 600 annotated sentences). Once the sentences were annotated, the annotation data was extracted, we calculated inter-annotator agreement and analyzed the output to see what the number of error tags can tell us about the performance of each system.

[10] `http://www.translate5.net/`

[11] `http://www.qt21.eu/downloads/annotatorsGuidelines-2014-06-11.pdf`

3.3. Inter-Annotator Agreement

Though carefully thought out and developed, the MQM metrics, and manual MT evaluation in general, are notorious for resulting in low inter-annotator agreement scores. This is attested by the body of work that has addressed this issue, most notably Lommel et al. (2014b), who worked specifically on MQM, and (Callison-Burch et al., 2007), who investigated several tasks. This is why it is important that we check how well our annotators agree on the task at hand, and whether this is consistent with other work done with MQM so far.

Once the data was annotated, agreement was observed at the sentence level, and inter-annotator agreement was calculated using the Cohen's Kappa (κ) metric (Cohen, 1960). Agreement was calculated on the annotations of every system separately, as well as on a concatenation of annotations, in order to both see whether there are differences in agreement across systems, as well as to gain insight into the overall agreement between annotators. Additionally, Coehn's κ was also calculated for every error type separately. Detailed results can be found in Table 2.

Generally, one can see that our annotators agree best on evaluations of the PBMT system, less so on evaluations of the Factored SMT system, and least in evaluations of the NMT system. Overall agreement scores are relatively low - the average total κ is approximately 0.51. Furthermore, the κ scores are relatively consistent across all error types, mostly ranging between 0.35 and 0.55. According to Cohen, such scores constitute moderate agreement. However, as already stated, this is to be expected, given the complexity of the problem and annotation schema. In fact, this is a notably higher score than what has been reported in similar work, e.g. Lommel et al. (2014b), who achieve κ scores ranging between 0.25 and 0.34. However, this comparison should be taken with a grain of salt, as our calculations are just an approximation compared to Lommel et al.'s, given that in our setup we looked only at sentence level agreement, while they calculated agreement on the token level.

3.4. Results of annotation

Directly extracting raw annotation data from the `translate5` system provides a sum of error tags annotated for each error type by each annotator and system. The total values are presented in Table 3.

Looking at the aggregate data alone, one can easily detect that both annotators have judged that the PBMT system contains the most errors, and that the NMT system contains the smallest number of errors. This trend is consistent across most fine-grained error categories as well.

However, even though simply counting the errors can provide insight into which system performs better, we thought that this approach does not adequately represent our findings, as it does not allow a proper quantification of the quality of the outputs. Certainly, based on data from Table 3 we can claim, for example, that the NMT system

Error type	PBMT	Factored	NMT	Concatenated
Accuracy				
Mistranslation	0.51	0.48	**0.58**	0.53
Omission	0.34	0.39	0.37	0.37
Addition	0.5	0.54	0.33	0.47
Untranslated	**0.86**	**0.86**	-0.02	**0.72**
Fluency				
Unintelligible	0.39	0.32	0	0.35
Register	0.37	0.2	0.22	0.27
Word order	0.56	0.33	0.21	0.4
Function words				
Extraneous	0.56	0.32	0.49	0.46
Incorrect	0.37	0.18	0.34	0.29
Missing	0	0.49	0	0.33
Tense...	0.44	0.36	0.15	0.38
Agreement	0.24	0.41	0	0.33
Number	0.53	0.55	0.52	0.54
Gender	0.46	0.59	0.48	0.53
Case	0.53	0.49	0.52	0.56
All errors	**0.56**	**0.49**	**0.44**	**0.51**

Table 2. Inter-annotator agreement (Cohen's κ values) for the MQM evaluation task. The highest score for any individual system and the concatenation, as well as the overall score, are shown in bold.

	Annotator 1			Annotator 2		
System	**PBMT**	**Factored**	**NMT**	**PBMT**	**Factored**	**NMT**
Total errors	317	276	178	264	199	132

Table 3. Total errors per system per annotator

produces less errors in general, or less errors of a specific type, but given that the outputs are different, as is the number of tokens in each translation, we decided to normalize the data.

To the best of our knowledge there is no related work on how to approach this, as previous work simply counts the number of MQM tags and stops there. After some consideration, we decided to normalize at the token level. I.e. instead of counting just error tags produced by each annotator, we count the tokens that these errors are assigned to – tokens that do and tokens that do not have an error annotation. Once

these numbers are divided by the total number of tokens in the system's output, they provide a concrete idea of the ratio of tokens with and without errors.

The results of such analysis again show that the PBMT system has the largest error ratio, while the NMT system has the smallest one. This is further backed up by a pairwise chi-squared (χ^2) statistical significance test; we calculate statistical significance from 2x2 contingency tables for every system pair (PBMTxFactored, PBMTxNMT and FactoredxNMT). The results show that the differences in the total number of tokens with errors are statistically significant for all three system pairs, with the p value being lower than 0.0001 in each case.

Furthermore, we also wanted to see which error types are the ones making a significant impact on this result. So we repeated these same measurements, but instead of performing them on all error types combined, they were performed separately for each specific error category. The combined results of the calculations and transformations are presented in Table 4.

We can derive several findings from this table. Firstly, when looking simply at the grand total of tokens with and without errors, the difference between the systems is statistically significant by a wide margin. When looking at PBMT and factored PBMT, the factored system has significantly less errors than the pure PBMT system. The overall error rate is in this case reduced by 20%. A separate analysis of specific error types that contribute to this score reveals that only some of the error categories are significantly different between the two systems. In the table, those categories are filled in with green. One can see that, when it comes to agreement, the only agreement type that produces significantly less errors is agreement in case.

However, taking a look at NMT shows that, not only does it result in a 42% overall error reduction compared to the factored system, and 54% with respect to pure PBMT, but it produces even less agreement errors – overall, as well as at the level of number, gender and case – while not using any kind of linguistic information at all. This might in part be due to the use of sub-word segmentation, as inflections in Croatian are relatively regular. In addition to improving in the Agreement category, NMT also produces significantly less errors in many more categories than the factored model does. Interestingly, it produces more Omission errors than either of the other two systems. It seems that it tends to sacrifice completeness of translation in order to increase overall fluency. Indeed, extrapolating from the data in Table 4, shows that, though differences are very small, NMT does have the lowest token per sentence ratio (PBMT 18.99, Factored PBMT 18.89, NMT 18.36).

4. Conclusion

The fine-grained manual evaluation performed for the purpose of this research has provided answers to several questions, one of which was the main drive behind our developing the factored system: is there a way to handle better agreement when

	PBMT		Factored		NMT	
Error type	**No error**	**Error**	**No error**	**Error**	**No error**	**Error**
Accuracy	3467	369	3525	*291	3402	266
Mistranslation	3547	289	3586	*230	3471	197
Omission	3801	35	3793	23	3619	*49
Addition	3814	22	3797	19	3655	13
Untranslated	3813	23	3797	19	3662	*6
Fluency	3195	641	3298	*518	3465	**188
Unintelligible	3790	46	3769	47	3668	**0
Register	3810	26	3794	22	3646	22
Spelling	3833	3	3812	4	3659	9
Grammar	3270	566	3371	**445	3497	**156
Word order	3752	84	3752	64	3646	**22
Function words	3801	35	3780	36	3650	*18
Extraneous	3829	7	3810	6	3664	4
Incorrect	3810	26	3790	26	3655	*13
Missing	3834	2	3812	4	3667	1
Word form	3389	447	3471	*345	3538	**102
Part of speech	3822	14	3800	16	3663	*5
Tense...	3775	61	3765	51	3648	*20
Agreement	3466	370	3540	*276	3566	**102
Number	3778	58	3772	44	3646	*22
Gender	3788	48	3756	60	3644	*24
Case	3614	222	3694	*122	3622	**46
Person	3836	0	3816	0	3664	4
Total errors	2826	1010	3007	**809	3199	**469

*Table 4. Processed annotation data from both annotators concatenated: each system's total number of tokens with and without errors. Statistical significance for a system, when compared to the system on its left, is marked with * where* p*-value is <0.05 and ** where* p*-value is <0.0001. Cells with a green background indicate that the system has less errors than the one on its left, while those in red indicate that it has more.*

translating to Croatian? We can now confidently claim that factored models result in significantly less agreement errors overall compared to pure PBMT.

We can also confidently claim that NMT handles all types of agreement better than both pure PBMT and factored PBMT, which corroborates the findings of other researchers' NMT evaluations. Our system produces sentences with far less errors, and a language that is more fluent and more grammatical, which should be of help when it comes to the task of post-editing.

Furthermore, the error taxonomy that was developed for this research, while only used for the English-to-Croatian language direction, should be applicable for the analysis of errors for any translation direction towards a Slavic language, as it takes into account grammatical properties specific to these languages.

Among other possible lines of future work, including the application of our methodology to another language pair (e.g. English-Czech), performing more controlled IAA analysis or IAA adjudication, as well as comparing to an NMT model without sub-word segmentation, another one is adapting the tagset further. In its current version, it has proved to be informative when comparing PBMT to factored PBMT. However, NMT has shown itself to produce language that is so fluent that the fine-grained hierarchy in the *Fluency* branch is of little use. Meanwhile, the most common error type in the NMT output is *Mistranslation*, which, according to the MQM guidelines, covers both lexical selection and, less intuitively, translation of grammatical properties (e.g. if 'cats[pl.]' is translated as 'mačka[sg.]', this is to be tagged as *Mistranslation*, in spite of correct lexical choice). This makes it quite a vague category, so if one would wish to perform an even more nuanced linguistic error analysis for NMT, adding additional layers to the *Accuracy* branch would seem a promising direction to follow.

Acknowledgements

We would like to extend our thanks to Maja Popović, who provided valuable advice on how to approach the annotation and evaluation, and Denis Kranjčić, who participated in the annotation task. The research leading to these results has received funding from the European Union Seventh Framework Programme FP7/2007-2013 under grant agreement PIAP-GA-2012-324414 (Abu-MaTran) and the Swiss National Science Foundation grant 74Z0_160501 (ReLDI).

Bibliography

Bentivogli, Luisa, Arianna Bisazza, Mauro Cettolo, and Marcello Federico. Neural versus Phrase-Based Machine Translation Quality: a Case Study. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 257–267. Association for Computational Linguistics, 2016.

Callison-Burch, Chris, Cameron Fordyce, Philipp Koehn, Christof Monz, and Josh Schroeder. (Meta-) evaluation of machine translation. In *Proceedings of the Second Workshop on Statistical Machine Translation*, pages 136–158. Association for Computational Linguistics, 2007.

Cohen, Jacob. A Coefficient of Agreement for Nominal Scales. *Educational and Psychological Measurement*, 20(1):37–46, 1960. doi: 10.1177/001316446002000104.

Devlin, Jacob, Rabih Zbib, Zhongqiang Huang, Thomas Lamar, Richard Schwartz, and John Makhoul. Fast and Robust Neural Network Joint Models for Statistical Machine Translation. In *Proceedings of ACL*, pages 1370–1380, 2014.

Durrani, Nadir, Helmut Schmid, and Alexander Fraser. A joint sequence translation model with integrated reordering. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 1045–1054. Association for Computational Linguistics, 2011.

Galley, Michel and Christopher D Manning. A simple and effective hierarchical phrase reordering model. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 848–856. Association for Computational Linguistics, 2008.

Ljubešić, Nikola and Filip Klubička. {bs,hr,sr}WaC – Web corpora of Bosnian, Croatian and Serbian. In *Proceedings of the 9th Web as Corpus Workshop (WaC-9)*, pages 29–35, Gothenburg, Sweden, 2014. Association for Computational Linguistics.

Lommel, Arle Richard, Aljoscha Burchardt, and Hans Uszkoreit. Multidimensional Quality Metrics (MQM): A Framework for Declaring and Describing Translation Quality Metrics. *Tradumàtica: tecnologies de la traducció*, 0(12):455–463, 12 2014a.

Lommel, Arle Richard, Maja Popovic, and Aljoscha Burchardt. Assessing inter-annotator agreement for translation error annotation. In *MTE: Workshop on Automatic and Manual Metrics for Operational Translation Evaluation*, 2014b.

Moore, Robert C. and William Lewis. Intelligent Selection of Language Model Training Data. In *Proceedings of the ACL 2010 Conference Short Papers*, ACLShort '10, pages 220–224, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics.

Sánchez-Cartagena, Victor M., Nikola Ljubešić, and Filip Klubička. Dealing with Data Sparseness in SMT with Factured Models and Morphological Expansion: a Case Study on Croatian. *Baltic Journal of Modern Computing*, 4(2):354–360, 2016.

Sennrich, Rico, Barry Haddow, and Alexandra Birch. Neural machine translation of rare words with subword units. *arXiv preprint arXiv:1508.07909*, 2015.

Toral, Antonio and Víctor M. Sánchez-Cartagena. A Multifaceted Evaluation of Neural versus Phrase-Based Machine Translation for 9 Language Directions. *CoRR*, abs/1701.02901, 2017.

Address for correspondence:
Filip Klubička
fklubick@ffzg.hr
Faculty of Humanities and Social Sciences
3 Ivan Lučić Street, Zagreb, PA 10000, Republic of Croatia

PBML

The Prague Bulletin of Mathematical Linguistics
NUMBER 108 JUNE 2017 133–145

A Neural Network Architecture for Detecting Grammatical Errors in Statistical Machine Translation

Arda Tezcan, Véronique Hoste, Lieve Macken

LT³, Language and Translation Technology Team
Department of Translation, Interpreting and Communication
Ghent University

Abstract

In this paper we present a Neural Network (NN) architecture for detecting grammatical errors in Statistical Machine Translation (SMT) using monolingual morpho-syntactic word representations in combination with surface and syntactic context windows. We test our approach on two language pairs and two tasks, namely detecting grammatical errors and predicting overall post-editing effort. Our results show that this approach is not only able to accurately detect grammatical errors but it also performs well as a quality estimation system for predicting overall post-editing effort, which is characterised by all types of MT errors. Furthermore, we show that this approach is portable to other languages.

1. Introduction

Despite the recent improvements in machine translation (MT), the task of producing grammatically correct sentences remains challenging for MT systems and post-editing is still necessary to obtain high quality translations. Furthermore, Statistical Machine Translation (SMT) systems seem to suffer more from grammatical errors than Neural Machine Translation (NMT) systems (Bentivogli et al., 2016), which pushed ahead the state of the art and challenged the dominance of SMT systems in recent (Bojar et al., 2016). The accurate detection of grammatical errors at the word level can be used as a major component for estimating the quality and post-editing effort in machine-translated texts. Moreover, such systems can assist post-editors by high-

Corresponding author: arda.tezcan@ugent.be
Cite as: Arda Tezcan, Véronique Hoste, Lieve Macken. A Neural Network Architecture for Detecting Grammatical Errors in Statistical Machine Translation. The Prague Bulletin of Mathematical Linguistics No. 108, 2017, pp. 133–145. doi: 10.1515/pralin-2017-0015.

lighting errors, can inform MT developers about the strengths and/or weaknesses of MT systems and can further be developed as Automatic Post-Editing (APE) systems.

In this paper we present a Recurrent Neural Network (RNN) architecture for word-level detection of grammatical errors in SMT output by using word vectors that represent the *PoS*, *morphology* and *dependency relation* of words within surface and syntactic context windows. We test this approach for the English-Dutch (EN-NL) language pair and show that it can be used to detect grammatical errors with high accuracy, even when a relatively small data set is provided. Our results also indicate that, to detect grammatical errors in MT output, morpho-syntactic word representations are more informative than word embeddings, which capture precise syntactic and semantic word relationships (Mikolov et al., 2013). Furthermore, we apply this approach to predict overall post-editing effort for the EN-NL and English-German (EN-DE) language pairs by only relying on monolingual morpho-syntactic word representations, which do not provide any information about the semantic properties of words.

2. Related Work

Quality Estimation (QE) is the task of providing a quality indicator for machine-translated text without relying on reference translations (Gandrabur and Foster, 2003). Word-level QE, which can identify and locate problematic text fragments within a given MT output, has gained more attention in recent years (Bojar et al., 2016).

The detection of grammatical errors in MT output, without relying on reference translations, can be considered as a QE task that caters for a particular MT error type. Stymne and Ahrenberg (2010) used a rule-based grammar checker to assess and post-edit grammatical errors of their English-Swedish SMT system. Ma and McKeown (2012) decomposed parse trees of Chinese-English MT output into elementary trees and reconstructed the original parse trees using attribute value matrices that define the syntactic usage of each node in each tree. They considered reconstruction failures as indicators of grammatical errors. Recently, Tezcan et al. (2016) obtained dependency parse trees on English-Dutch MT output and queried the sub-trees of each parse tree against a treebank of correct sentences in the target language. The number of matching constructions were then used to mark words as grammatically incorrect.

Neural Networks (NNs) have been applied to many tasks in the Natural Language Processing (NLP) community, with language modelling (Bengio et al., 2003) and MT (Bahdanau et al., 2014) being two examples. In recent years, NNs have also shown promising results for sentence and word-level QE in different languages and domains (Kreutzer et al., 2015; Patel and Sasikumar, 2016). Moreover, focusing on the detection of grammatical errors from a different perspective, Liu and Liu (2016) proposed to derive positive and negative samples from unlabelled data, by generating grammatical errors artificially, and showed that RNNs outperform Support Vector Machines (SVMs) in judging the grammaticality of each word. All these NN-based systems uti-

lized distributed word embeddings within context windows that preserve the original word sequence of given texts.

3. Morpho-syntactic Word Representations

Our assumption is that syntactic, morphological and dependency-related information about words provides useful information for detecting grammatical errors made by MT systems. Therefore, we have transformed each word in a given MT output into a feature vector using *multi-hot encoding*, which represents three types of information at the same time: *PoS*, *morphology* and *dependency relation*. These binary vectors are the same length as the size of the total vocabulary of all three types of information. In each word vector, all elements are assigned the value of 0, except the elements representing the linguistic features of each word, which are assigned 1. As a result, in this representation, each word is accurately represented with respect to its morpho-syntactic features, while avoiding the data sparsity issue, given the small vocabulary sizes of *PoS*, *morphology* and *dependency* labels.

Another approach to word representations is learning a distributed representation (word embeddings) for each word, which is dense and real-valued. Each dimension in distributed word representations, which are called word embeddings, represent a latent feature of a word, hopefully capturing useful syntactic and semantic properties (Turian et al., 2010). Unlike word embeddings, the morpho-syntactic representation strips out semantic features from words, which can be considered as unnecessary information for the task of detecting grammatical errors in MT output. Figure 1 shows an example source sentence (EN), its machine-translated version (NL) and the morpho-syntactic representation for the word *'zijn (are)'*. The MT output in this figure contains a grammatical error in the form of subject-verb agreement in number between the words *'zijn (are)'* (plural) and *'kans (chance)'* (singular). We obtain the morpho-syntactic features for Dutch using the Alpino parser (Van Noord, 2006).

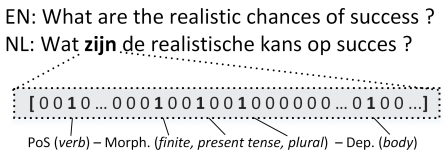


Figure 1. Binary vector for 'zijn (are)' consisting of 1s for its PoS, morphology and dependency features and 0s for the remaining items in the vocabulary.

4. Surface and Syntactic N-Grams

Surface n-grams are sequences of words as they appear in texts, with n corresponding to the number of words in the sequence. While surface n-grams have been used effectively in various types of NLP tasks, they primarily rely on local context and are not informative on a syntactic level. Dependency trees, on the other hand, represent words in a sentence as nodes and grammatical relations between the words as edges. Unlike the surface n-grams, syntactic n-grams, which can be constructed by using paths in dependency trees, offer context windows based on syntactic neighbours of words and are able to capture long-distance dependencies.

Given each target word in the MT output, we consider four different fixed-sized context windows, which are based on the following surface and syntactic n-grams:

Surface n-gram (n): Sequence of words as they appear in MT output, centered around the target word (n=5)

Syntactic n-grams (sn):

- **Parents (sn_p)**: Vertical sequence of parent nodes in a given dependency tree for a given target node (n=3)
- **Siblings (sn_s)**: Horizontal sequence of sibling nodes sharing the same parent in a given dependency tree, centered around the target node (n=5)
- **Children (sn_c)**: Sequence of children nodes for a given target node (depth 1), containing the target node in the centre (n=5)

We include additional placeholder tokens in the vocabulary of the morpho-syntactic features to indicate boundaries (namely '<s>' to indicate a sentence boundary, '[ROOT]' to indicate the root of the dependency tree and '[NA]' to indicate horizontal boundaries in the sub-trees). Moreover, we preserve the original word order in each syntactic n-gram, with the aim of capturing word ordering errors in machine-translated texts. The n values for the four n-gram types are chosen as the best values between 3 up to 5, which maximized the estimation performance when all n-gram types are used together[1]. The four different context windows extracted for the word '*zijn (are)*' are illustrated in Figure 2.

One difficulty of using dependency parsers on MT output is that the syntactic relationships between words can only be accurately captured provided that a correct dependency parse tree is obtained to start with. This can be illustrated in the example in Figures 1 and 2. In this example, the surface 5-gram context window is unable to capture the dependency relation between the two words generating the grammatical error: '*zijn (are)*'[2] (plural) and '*kans (chance)*' (singular). While the syntactic n-gram (children) is able to capture it, the disagreement cannot be directly observed

[1]The 99-percentile for the number of parents (up to the root), siblings and children over all tokens in the dependency trees generated for the EN-NL data set are observed as 10, 4 and 4, respectively.

[2]In the example given in Figures 1 and 2, the dependency label *body* refers to the body of a verbal projection within a WH-phrase, headed by the word 'wat', which is marked as *ROOT*. Detailed information (in Dutch) about the syntactic annotations used by the Alpino parser can be found at `http://www.let.rug.nl/vannoord/Lassy/sa-man_lassy.pdf`

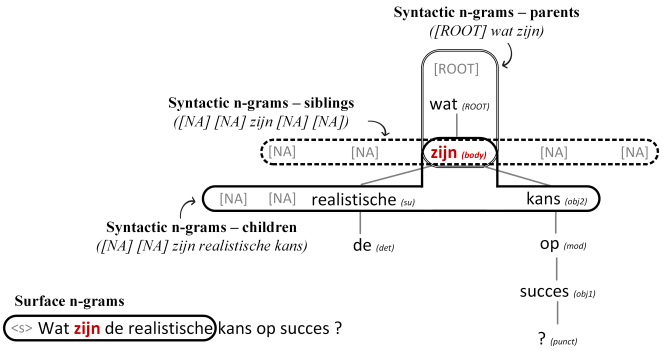


Figure 2. A machine-translated sentence (lower left), its dependency parse tree (upper right) and the four different context windows used for the target word 'zijn (are)'.

in the parse tree since the parser (incorrectly) labels '*realistische (realistic)*' as the subject of the sentence. Tezcan et al. (2016) show that dependency parsers can nevertheless be useful to detect grammatical errors due to the unusual dependency structures they produce on MT output that contains errors. Similarly, our motivation for using syntactic n-grams is to learn such unusual structures by exploiting morpho-syntactic word representations in combination with dependency structures.

5. Neural Network Architecture

We propose a neural network architecture that uses Gated Recurrent Units (GRUs) (Cho et al., 2014). Similar to Long Short Term-Memory (LSTM)(Hochreiter and Schmidhuber, 1997), GRU is a variant of RNNs that are well suited to learn from history to process time series. Despite their similarities, LSTMs and GRUs have been shown to outperform each other in particular NLP tasks. LSTMs, for example, seem to be a better approach for language modelling (Irie et al., 2016). GRUs, on the other hand, have been shown to perform better in the task of word-level quality estimation of machine translation (Patel and Sasikumar, 2016).

We provide four different context vectors (as described in Section 4) as inputs to four GRU layers, which are concatenated before they are connected to the output layer, which consists of two units. The softmax over the activation of these two units is taken as a score for the two classes OK and BAD, which represent the correct and erroneous words, respectively. To reduce overfitting, we apply dropout (Srivastava et al., 2014) within the GRU layers (for the input gates and the recurrent connections) and after the concatenated hidden units. Figure 3 illustrates the proposed NN architecture.

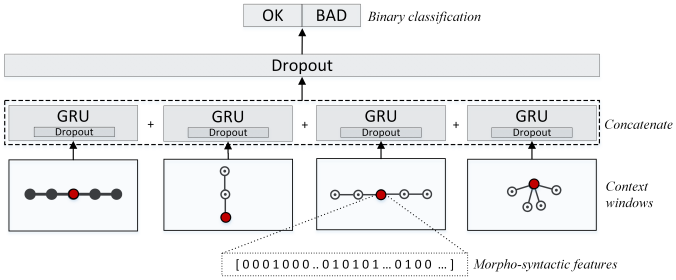


Figure 3. The proposed neural network architecture.

In all of our experiments, we have used binary cross-entropy as loss function and RMSProp (Tieleman and Hinton, 2012) as optimiser. We fixed the mini-batch size to 50 and trained each model for 50 epochs[3]. We implemented all models using the TensorFlow framework (Abadi et al., 2016). We adjusted the sizes of the GRU layers according to the sizes of the two data sets we used in our experiments, which are detailed in Section 6.

6. Experiments

We evaluated the proposed method on two tasks: detecting grammatical errors and predicting post-editing effort in SMT. In the first experiment we evaluate the performance of the proposed method on detecting grammatical errors for EN-NL. The second experiment aimed to find out if the same method could successfully be applied to a different language pair (EN-DE) and whether it could be used to predict overall post-editing effort. In both experiments, we considered F1_MULTI as the primary evaluation metric, which is the multiplication of F1 scores for the OK and BAD classes.

6.1. Detecting Grammatical Errors

To evaluate the performance of the proposed method in the task of detecting grammatical errors, we used the SCATE corpus of SMT errors (Tezcan et al., in press), which consists of 2967 sentence pairs. The source sentences in this data set were extracted from three different text types of the Dutch Parallel Corpus (Macken et al., 2011). The translations in this data set were obtained from Google Translate[4]. This corpus contains manual error annotations, which are classified based on the distinction between

[3]The mini-batch size of 50 has been selected as the best value after training the system with different sizes from 25 to 'full batch'. In all out experiments, each network converged to a point of minimal error after 40 epochs.

[4]http://translate.google.com (June, 2014)

fluency and accuracy by referring to the type of information that is needed to detect them. According to this taxonomy, fluency errors are detected on the target text alone (monolingual level), while to detect both the source and target text need to be analyzed (bilingual level). The fluency errors are further divided into the following sub-categories: *grammar*, *lexicon*, *orthography*, *multiple errors* and *other fluency errors*. To evaluate the proposed method we used the annotations of *grammar* and *multiple errors*. The label *multiple errors* was used when different fluency errors occurred at the same time, e.g. a word order combined with a wrong lexical choice. It is safe to say that most of the words labelled as *multiple errors* contain grammatical problems. As a result, the data set that we used in this experiment consisted of 58002 words, with an OK to BAD ratio of approx. 3.4:1. All systems in this experiment were evaluated using the average 10-fold cross-validation scores. To handle the issue of skewed distribution of labels, during training, we assigned class weights that are inversely proportional to their frequency in each training fold. For the EN-NL experiments, we trained the NN systems with GRU layer sizes of 50. We used the Alpino parser to extract the morpho-syntactic features for each word in a given MT output. The resulting word vectors consist of 128 features.

In the first part of this experiment we compared the proposed NN architecture (NN-MS) and the impact of using different morpho-syntactic features in this architecture to a baseline system proposed by Tezcan et al. (2016). The baseline system is based on querying subtrees of the dependency trees obtained on the MT output against a treebank of dependency trees built from correct sentences[5]. Considering their ability to capture syntactic and semantic properties of words, we also compare the effectiveness of word embeddings (NN-*word2vec*) to morpho-syntactic features as alternative word representations. For this purpose, we pre-trained 200-dimensional *word2vec* word embeddings (Mikolov et al., 2013) using 328M words from the SoNaR corpus (Oostdijk et al., 2008)[6]. In this experiment, we evaluated all NN systems using the surface 5-gram context windows (n).

The results in Table 1 clearly show that the NN architectures perform better than a simple frequency-based method (Baseline). We see that using only PoS features in the NN architecture is enough to beat this baseline system. Moreover, introducing additional morpho-syntactic features further improves the system. The positive effect of using dependency labels supports our hypothesis that they provide useful information for learning grammatical errors, even though the parser makes mistakes (as shown in Figure 2). Finally, we see that the performance of this NN architecture drastically improves when all three morpho-syntactic features are used instead

[5]Even though this system is evaluated on a subset of the data set used in this paper, it can safely be compared to the proposed method, given that it uses the same annotation set (*grammar* and *multiple errors*) and the assumption that it would achieve similar results on a larger test set because it is not based on machine-learning methods.

[6]We replace singleton words in the training data with <*unk*> to handle unknown words and apply zero padding to the n-grams containing sentence boundaries

	F1_BAD	F1_OK	F1_MULTI
Baseline (Tezcan et al., 2016)	0.3811	0.6789	0.2587
NN-MS - PoS (n)	0.4343	0.7493	0.3253
NN-MS - PoS+Morph (n)	0.4561	0.7951	0.3626
NN-MS - PoS+Morph+Dep (n)	**0.4729**	**0.8138**	**0.3848**
NN-*word2vec* (n)	0.4110	0.7779	0.3204

Table 1. Performance of the baseline system and the NN systems using different word representations.

of word embeddings. This observation suggests that the semantic and syntactic relationships captured by word embeddings are not as informative as the proposed morpho-syntactic features for this task.

In the second part of this experiment, we analyzed the predictive power of the surface and syntactic n-grams as context windows. Table 2 provides an overview of the performance of the different systems using the same three morpho-syntactic features with different combinations of context windows.

	F1_BAD	F1_OK	F1_MULTI
NN-MS (n)	0.4729	0.8138	0.3848
NN-MS ($\mathrm{sn_p}$)	0.4053	0.7806	0.3162
NN-MS ($\mathrm{sn_s}$)	0.4079	0.7861	0.3255
NN-MS ($\mathrm{sn_c}$)	0.4077	0.7865	0.3203
NN-MS ($\mathrm{n} + \mathrm{sn_p} + \mathrm{sn_s} + \mathrm{sn_c}$)	**0.4799**	**0.8338**	**0.3998**
NN-MS ($\mathrm{sn_p} + \mathrm{sn_s} + \mathrm{sn_c}$)	0.4383	0.8135	0.3565

Table 2. Performances of the NN systems using the three morpho-syntactic features with different combinations of context windows.

As is evident from the results of the four different context windows in isolation, the surface n-gram context window provides the most useful information when used alone. The syntactic n-grams seem to contain extra useful information and maximize the performance of the system when they are used in combination with the surface n-gram windows. Furthermore, removing a specific type of context window from the combined set reduces the performance in all cases. The largest drop in performance occurs when the surface n-grams are removed, which confirms the usefulness of the information provided by this context window.

6.2. Predicting Post-editing Effort

Applying the proposed method to a different language requires the use of a different set of language-specific NLP tools and/or models. To compare the performance over different languages, we applied the proposed method to predict post-editing ef-

fort to two different language pairs, namely to EN-NL and EN-DE. For EN-DE we tested this method on the WMT'16 data set, which has been used in the shared task on word-level QE. This data set consists of 15K source-target sentence pairs (279976 words in the target language) in the IT-domain with target sentences being the machine-translated version of the source sentences by a phrase-based SMT system. The data was partitioned into 12K, 1K and 2K sentence pairs as training, tuning and test sets, respectively. All words in this data set have been automatically annotated for errors with binary word-labels (OK and BAD) using the alignments between the MT output and its post-edited version provided by the TER tool[7] (Snover et al., 2006). In all three data sets, the OK to BAD ratio is approx. 4:1. Prior to training the NN, we obtained PoS, morphology and dependency labels for each German word in the MT output (in CoNLL-U format), using the Mate tools (Bohnet and Nivre, 2012). The resulting word vectors consist of 127 features. During training, we assigned class weights that are inversely proportional to their frequencies in the training set. For the EN-DE experiments, we trained the NN system with GRU layer sizes of 100 (instead of 50) and increased the complexity of the NN, given the relatively larger data set compared to the EN-NL language pair. For the EN-NL language pair, we used the same NN architecture and the data set as detailed in Section 6.1 with one difference: instead of using the gold-standard error annotations for grammatical errors, for this experiment, we automatically annotated the words for errors using the same procedure in the shared task of QE (WMT'16), by using the TER tool. For this purpose, we used the post-edited version of the MT output from a Master student in translation studies.

We evaluated the EN-NL system with regard to the average cross-validation results. The evaluation of the EN-DE system, on the other hand, was conducted on the test set made available by the organizers. This approach allows us to additionally compare the performance of the EN-DE system with the competing systems in the shared task. We trained both systems using the morpho-syntactic features consisting of *PoS*, *morphology* and *dependency* features and the four context windows consisting of surface and syntactic n-grams. Table 3 provides an overview of the performance of the proposed method for the two language pairs.

	F1_BAD	**F1_OK**	**F1_MULTI**
EN-NL (SCATE) - avg. cross val.	0.4335	0.8649	0.3749
EN-DE (WMT'16) - held out test set	0.4224	0.8319	0.3514

Table 3. Performance of the NN systems for predicting post-editing effort.

[7]The settings used are: tokenized, case insensitive, exact matching only. *Deletions* are not annotated as they cannot be associated with any word and *shifts* are disabled, but rather annotated as edits in the form of *deletions* and *insertions*.

From Table 3, we can see that, despite the difference between the data sizes and the tools we used, both systems obtained similar results. Furthermore, by comparing the results obtained for the EN-NL system on the two tasks, we can see that the proposed method performs better on detecting grammatical errors (F1_MULTI = 0.3998, as provided in Figure 2) than predicting overall post-editing effort (F1_MULTI = 0.3749), which represents all types of MT errors. We can gain a better picture of the performance of the proposed method on predicting post-editing effort when we compare the EN-DE system with the systems that participated in the shared task of word level QE in WMT'16 (Bojar et al., 2016). Three of these systems (out of 14) are provided in Table 4.

	Rank	**F1_BAD**	**F1_OK**	**F1_MULTI**
UNBABEL/ensemble	1	0.5599	0.8845	0.4952
CDACM/RNN	8	0.4192	0.8421	0.3531
EN-DE (WMT'16)	-	0.4224	0.8319	0.3514
BASELINE	11	0.3682	0.8800	0.3240

Table 4. Performances of the proposed NN architecture in comparison to three competing systems (and the ranks they achieved) in WMT'16 shared task on QE.

The proposed system outperforms the baseline system used in this shared task, consisting of 22 features representing monolingual and bilingual properties of each translated text. Moreover, it performs slightly worse than another GRU-based NN system (CDACM/RNN), which uses *word2vec* word embeddings within monolingual context windows of surface n-grams (Patel and Sasikumar, 2016). This observation shows that the morpho-syntactic features can provide almost as useful information as word embeddings for learning overall post-editing effort.

7. Conclusion

We have proposed an RNN architecture for word-level detection of grammatical errors in SMT that utilizes monolingual features in context windows of surface and syntactic n-grams. Our approach relies on PoS, morphological and dependency information of the MT output and uses multi-hot encoding to represent the morpho-syntactic properties of words as word vectors. We showed that this approach achieves high performance on EN-NL SMT output, even when a relatively small training set is available. Moreover, our results suggest that word embeddings, despite their informativeness on syntactic and semantic properties of words, should not be considered as a one-size-fits-all approach in the QE task. For detecting grammatical errors in SMT output, we achieved a marked improvement in performance by using accurate morpho-syntactic features over word embeddings. By applying the proposed

approach on the task of predicting post-editing effort, we demonstrated its ability to learn all MT error types on two language pairs, EN-NL and EN-DE. This observation shows the applicability of the proposed method across languages and reveals the amount of valuable monolingual information that can be employed for estimating overall quality in machine-translated texts.

Building separate error-detection systems that are trained on different types of MT errors can be considered as an alternative approach to existing QE systems, which try to make a direct estimation of overall quality. By combining such specialized systems, we would like to build a single QE system that does not only achieve high performance on the QE task, but can be informative about the reasons of the estimated quality and the types and the location of errors MT systems make. We would also like to adapt this approach with a view to detecting common errors in NMT systems, which seem to make fewer grammatical errors compared to SMT systems.

Acknowledgements

This research has been carried out in the framework of the SCATE project funded by the Flemish government agency IWT (IWT-SBO 130041).

Bibliography

Proceedings of the Tenth Workshop on Statistical Machine Translation, WMT@EMNLP 2015, 17-18 September 2015, Lisbon, Portugal, 2015. The Association for Computer Linguistics. ISBN 978-1-941643-32-7. URL `http://aclweb.org/anthology/W/W15/`.

Abadi, Martín, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Gregory S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, et al. TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems. *CoRR*, abs/1603.04467, 2016.

Bahdanau, Dzmitry, Kyunghyun Cho, and Yoshua Bengio. Neural Machine Translation by Jointly Learning to Align and Translate. *CoRR*, abs/1409.0473, 2014.

Bengio, Yoshua, Réjean Ducharme, Pascal Vincent, and Christian Janvin. A Neural Probabilistic Language Model. *J. Mach. Learn. Res.*, 3:1137–1155, 2003. ISSN 1532-4435.

Bentivogli, Luisa, Arianna Bisazza, Mauro Cettolo, and Marcello Federico. Neural versus Phrase-Based Machine Translation Quality: a Case Study. *CoRR*, abs/1608.04631, 2016.

Bohnet, Bernd and Joakim Nivre. A Transition-based System for Joint Part-of-speech Tagging and Labeled Non-projective Dependency Parsing. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, EMNLP-CoNLL '12, pages 1455–1465. Association for Computational Linguistics, 2012.

Bojar, Ondrej, Rajen Chatterjee, Christian Federmann, Yvette Graham, Barry Haddow, Matthias Huck, et al. Findings of the 2016 Conference on Machine Translation. In *Proceedings of the First Conference on Machine Translation, WMT 2016, colocated with ACL 2016, August 11-12, Berlin, Germany*, pages 131–198, 2016.

Cho, Kyunghyun, Bart van Merriënboer, Çağlar Gülçehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734. Association for Computational Linguistics, 2014.

Gandrabur, Simona and George Foster. Confidence Estimation for Translation Prediction. In *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003 - Volume 4*, CONLL '03, pages 95–102. Association for Computational Linguistics, 2003.

Hochreiter, Sepp and Jürgen Schmidhuber. Long Short-Term Memory. *Neural Computation*, 9 (8):1735–1780, 1997.

Irie, Kazuki, Zoltán Tüske, Tamer Alkhouli, Ralf Schlüter, and Hermann Ney. LSTM, GRU, Highway and a Bit of Attention: An Empirical Overview for Language Modeling in Speech Recognition. In Morgan (2016), pages 3519–3523. doi: 10.21437/Interspeech.2016. URL `http://dx.doi.org/10.21437/Interspeech.2016`.

Kreutzer, Julia, Shigehiko Schamoni, and Stefan Riezler. QUality Estimation from ScraTCH (QUETCH): Deep Learning for Word-level Translation Quality Estimation. In *Proceedings of the Tenth Workshop on Statistical Machine Translation, WMT@EMNLP 2015, 17-18 September 2015, Lisbon, Portugal* DBL (2015), pages 316–322. ISBN 978-1-941643-32-7. URL `http://aclweb.org/anthology/W/W15/`.

Liu, Zhuoran and Yang Liu. Exploiting Unlabeled Data for Neural Grammatical Error Detection. *CoRR*, abs/1611.08987, 2016.

Ma, Wei-Yun and Kathleen McKeown. Detecting and Correcting Syntactic Errors in Machine Translation Using Feature-Based Lexicalized Tree Adjoining Grammars. *IJCLCLP*, 17(4), 2012.

Macken, Lieve, Orphée De Clercq, and Hans Paulussen. Dutch parallel corpus: a balanced copyright-cleared parallel corpus. *Meta: Journal des traducteursMeta:/Translators' Journal*, 56 (2):374–390, 2011.

Mikolov, Tomas, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed Representations of Words and Phrases and their Compositionality. In *Advances in Neural Information Processing Systems 26*, pages 3111–3119. Curran Associates, Inc., 2013.

Morgan, Nelson, editor. *Interspeech 2016, 17th Annual Conference of the International Speech Communication Association, San Francisco, CA, USA, September 8-12, 2016*, 2016. ISCA. doi: 10.21437/Interspeech.2016. URL `http://dx.doi.org/10.21437/Interspeech.2016`.

Oostdijk, N., M. Reynaert, P. Monachesi, G. Van Noord, R. Ordelman, and I. Schuurman. From DCoi to SoNaR: a reference corpus for Dutch. In *In Proceedings of the Sixth International Conference on Language Resources and Evaluation*, 2008.

Patel, Raj Nath and M. Sasikumar. Translation Quality Estimation using Recurrent Neural Network. *CoRR*, abs/1610.04841, 2016.

Snover, Matthew, Bonnie Dorr, Richard Schwartz, Linnea Micciulla, and John Makhoul. A study of translation edit rate with targeted human annotation. In *In Proceedings of Association for Machine Translation in the Americas*, pages 223–231, 2006.

Srivastava, Nitish, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *J. Mach. Learn. Res.*, 15(1):1929–1958, 2014.

Stymne, Sara and Lars Ahrenberg. Using a Grammar Checker for Evaluation and Postprocessing of Statistical Machine Translation. In *Proceedings of the Seventh conference on International Language Resources and Evaluation (LREC'10)*. European Language Resources Association (ELRA), 2010. ISBN 2-9517408-6-7.

Tezcan, Arda, Véronique Hoste, and Lieve Macken. Detecting grammatical errors in machine translation output using dependency parsing and treebank querying. *Baltic Journal of Modern Computing*, 4(2):203–217, 2016.

Tezcan, Arda, Véronique Hoste, and Lieve Macken. SCATE Taxonomy and Corpus of Machine Translation Errors. In *Trends in e-tools and resources for translators and interpreters*. Brill, in press.

Tieleman, T. and G. Hinton. Lecture 6.5—RmsProp: Divide the gradient by a running average of its recent magnitude. COURSERA: Neural Networks for Machine Learning, 2012.

Turian, Joseph, Lev Ratinov, and Yoshua Bengio. Word Representations: A Simple and General Method for Semi-supervised Learning. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, ACL '10, pages 384–394. Association for Computational Linguistics, 2010.

Van Noord, Gertjan. At last parsing is now operational. In *TALN06. Verbum Ex Machina. Actes de la 13e conference sur le traitement automatique des langues naturelles*, pages 20–42, 2006.

Address for correspondence:
Arda Tezcan
arda.tezcan@ugent.be
LT³ Language and Translation Technology Team
Department of Translation, Interpreting and Communication
Faculty of Arts and Philosophy
Ghent University
Groot-Brittanniëlaan 45, B-9000 Ghent, Belgium

PBML

The Prague Bulletin of Mathematical Linguistics
NUMBER 108 JUNE 2017 147–158

Evaluating the Usability of a Controlled Language Authoring Assistant

Rei Miyata,[a] Anthony Hartley,[b] Kyo Kageura,[c] Cécile Paris[d]

[a] Nagoya University
[b] Rikkyo University
[c] The University of Tokyo
[d] Data61, CSIRO

Abstract

This paper presents experimental results of a usability evaluation of a controlled language (CL) authoring assistant designed to help non-professional writers create machine translatable source texts. As the author drafts the text, the system detects CL rule violations and proscribed terms. It also incorporates several support functions to facilitate rephrasing of the source. In order to assess the usability of the system, we conducted a rewriting experiment, in which we compared two groups of participants, one with the aid of the system and the other without it. The results revealed that our system helped reduce the number of CL violations by about 9% and the time to correct violations by more than 30%. The CL-applied source text resulted in higher fluency and adequacy of MT outputs. Questionnaire and interview results also implied the improved satisfaction with the task completion of those participants who used the system.

1. Introduction

In recent years, machine translation (MT) has been increasingly adopted not only for company documentation but also by public services. A number of local governments in Japan have started using MT on their websites to provide local residents with multilingual information. However, the resultant translation is often confusing given that MT between distant languages such as Japanese and English is generally difficult (Isahara, 2015), and the necessary post-editing into multiple languages is too costly. To make better use of MT in the field, one viable solution is to constrain the source into a form amenable to MT by making use of a controlled language (CL), including properly controlled terminology.

 Corresponding author: miyata@nuee.nagoya-u.ac.jp
Cite as: Rei Miyata, Anthony Hartley, Kyo Kageura, Cécile Paris. Evaluating the Usability of a Controlled Language Authoring Assistant. The Prague Bulletin of Mathematical Linguistics No. 108, 2017, pp. 147–158.
doi: 10.1515/pralin-2017-0016.

Although the effectiveness of a CL itself is evidenced by improvements in not only machine translatability but also human readability (e.g., Bernth and Gdaniec, 2001; Aikawa et al., 2007; Miyata et al., 2015), writing in accordance with a particular CL is a hard task, especially for non-professional writers, such as those who create municipality websites and documents, since it requires a command of controlled writing. Thus, in practice it is essential to support authors in checking conformity to CL guidelines and terminology, and in editing the source text appropriately. In this research project, which focuses on Japanese-to-English translation of municipal documents, we developed an interactive authoring assistant designed to help non-professional writers create machine translatable source text (ST). The key feature of our system is that it supports users' decision-making at each step in validating the source.

While much effort has been devoted to the conventional product evaluation of MT (Bojar et al., 2015), few attempts have been made to assess the *usability* of an authoring support system to maximise MT use. In this study, we conducted a usability evaluation based on the ISO standard for human-computer interaction (ISO, 2010) and related studies (e.g., Doherty and O'Brien, 2013; Sauro and Lewis, 2012). To the best of our knowledge, this is the first attempt to evaluate the usability of a CL authoring assistant intended for improving MT performance.

We discuss related work in Section 2. In Section 3, we describe our CL guidelines compiled for this study. Section 4 explains the CL authoring assistant and the implementation of the guidelines. We elaborate on our experimental set up in Section 5 and present our results accompanied by discussion in Section 6. Section 7 presents conclusions and future directions.

2. Related Work

A number of CL rule sets have been proposed with a view to improving machine translatability as well as facilitating human comprehension (Kittredge, 2003; Kuhn, 2014). Evidence of improved machine translatability and post-editing productivity has also been provided (Pym, 1990; Bernth and Gdaniec, 2001; Aikawa et al., 2007; O'Brien and Roturier, 2007). Miyata et al. (2015) revealed in an evaluation experiment comparing four MT systems that compiling optimal rules for particular MT systems yields a great improvement in MT quality, a case also mentioned by O'Brien (2003).

Terminology management also plays a central role in improving both ST consistency and MT quality. In an experiment translating technical documentation from English to French, Thicke (2011) demonstrated that simply customising an MT engine with terminology boosted post-editing productivity. She also concluded that the combination of controlling the ST via general writing guidelines and customising the MT engine with terminology further increased translation productivity, making it four times faster than human translation from scratch.

However, writing source texts in accordance with a CL and pre-defined terminology is not an easy task. Providing writing support tools is essential, particularly for non-professional authors. A leading example of CL writing support in combina-

tion with MT is the KANTOO Controlled Language Checker (Mitamura et al., 2003; Nyberg et al., 2003), which incorporates functions to detect problems in the ST and provides diagnostic messages for interactive rewriting. Although some commercial source checking tools for Japanese have recently become available,[1] to date few practical implementations or evaluation results for Japanese CL tools have been provided.

While the performance of CL checkers has been benchmarked in terms of precision and recall of their violation detection components (Mitamura et al., 2003; Rascu, 2006; Miyata et al., 2016), to develop a workable system usability assessment is also crucially important. The relevant ISO standard defines usability as the 'extent to which a system, product or service can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use' (ISO, 2010). It describes these three measures as follows:

effectiveness: accuracy and completeness with which users achieve specified goals
efficiency: resources expended in relation to the accuracy and completeness with which users achieve goals
satisfaction: freedom from discomfort and positive attitudes towards the use of the product

Compared to the number of conventional product evaluations of MT by human subjective judgement or automated metrics (Bojar et al., 2015), the MT research community has published relatively few usability evaluations. Exceptions are, for example, Castilho et al. (2014) and Doherty and O'Brien (2012, 2013), who employed the three measures above to evaluate MT outputs, and Alabau et al. (2012), who conducted a user evaluation of their interactive MT systems. How usable a CL authoring system is for the end user remains an open question which needs to be addressed to enable the adoption of CL and MT in the workplace.

3. Controlled Language Guidelines

As just mentioned, CL rules become particularly effective when rule sets are tailored to specific MT systems. In our scenario, we focused on two MT systems: TransGateway,[2] a commercial rule-based MT (RBMT) system widely used in Japanese municipalities, and TexTra,[3] a freely available state-of-the-art statistical MT (SMT) system. We previously created a total of 60 Japanese CL rules and assessed the effectiveness of each rule with different MT systems (Miyata et al., 2015). Based on the evaluation results, we selected effective rules for each system and compiled two CL guidelines, henceforth **CL-R** and **CL-S**.[4]

[1] For example, Acrolinx supports several languages including Japanese. http://www.acrolinx.com/

[2] Kodensha CO., http://www.kodensha.jp

[3] NICT, https://mt-auto-minhon-mlt.ucri.jgn-x.jp

[4] R and S stand for RBMT and SMT, respectively.

No	Rule	CL-R	CL-S	Implement	
1	Try to write sentences of no more than 50 characters.	✓	✓	✓	(10)
2	Do not interrupt a sentence with a bulleted list.	✓	✓		
3	Ensure the relationship between the modifier and the modified is clear.	✓	✓		
4	Use the particle *Ga* only to mean 'but'.	✓	✓	✓	(9)
5	Do not use the preposition *Tame* to mean 'because'.	✓		✓	(10)
6	Avoid using multiple negative forms in a sentence.	✓		✓	(10)
7	Do not use *Reru/Rareru* to express the potential mood or honorifics.	✓	✓	✓	(4)
8	Avoid using words that can be interpreted in multiple ways.	✓	✓		
9	Avoid using the expression *To-iu*.	✓		✓	(8)
10	Avoid using the expression *Omowa-reru* and *Kangae-rareru*.	✓		✓	(10)
11	Avoid the single use of the form *Tari*.		✓	✓	(10)
12	Use words from a general Japanese-English dictionary.	✓	✓		
13	Avoid using compound Sahen-nouns.[5]		✓	✓	(10)
14	Ensure there are no typos or missing characters.	✓	✓		
15	Do not omit subject.	✓	✓	✓	(5)
16	Do not omit object.	✓	✓		
17	Do not use comma for connecting noun phrase enumeration.	✓	✓	✓	(7)
18	Avoid using particle *Ga* for object.	✓	✓	✓	(8)
19	Avoid using *Te-kuru/Te-iku*.		✓	✓	(10)
20	Avoid inserted adverbial clause.		✓		
21	Do not end clause with noun.	✓	✓		
22	Avoid using Sahen-noun + auxiliary verb *Desu*.	✓	✓	✓	(10)
23	Avoid using attributive use of *Shika-Nai*.	✓	✓	✓	(10)
24	Avoid using verb + *You-ni*.		✓	✓	(10)
25	Avoid using particle *Nado*.		✓	✓	(10)
26	Avoid using giving and receiving verb.	✓	✓	✓	(10)
27	Avoid using verbose word.	✓	✓	✓	(10)
28	Avoid using compound word.	✓	✓	✓	(9)
29	Do not omit parts of words in enumeration.	✓	✓	✓	(4)
30	Do not omit expression to mean 'per A'.	✓	✓	✓	(10)
31	Avoid using conjunctive particle *Te*.	✓	✓	✓	(10)
32	Avoid using particle *To* to mean 'if'.	✓		✓	(10)
33	Use Chinese Kanji characters for verb as much as possible instead of Japanese Kana characters.	✓	✓	✓	(4)
34	Avoid leaving bullet mark in texts.	✓	✓	✓	(10)
35	Avoid using machine dependent characters.	✓	✓	✓	(10)
36	Avoid using square bracket for emphasis.	✓	✓	✓	(10)
Term	Use term properly	✓	✓	✓	(10)

Table 1. CL rules and implementation (with precision scores)

Guideline **CL-R** comprises 30 rules while **CL-S** comprises 31 rules. The total number of distinct rules is 36, with 25 rules belonging to the both guidelines (Table 1).

For each CL rule, we provided a description and example rewrites, to enable authors to fully understand and apply the rule while drafting or revising.

[5] A Sahen-noun is a noun which can be connected to the verb *Suru* and act as a verb.

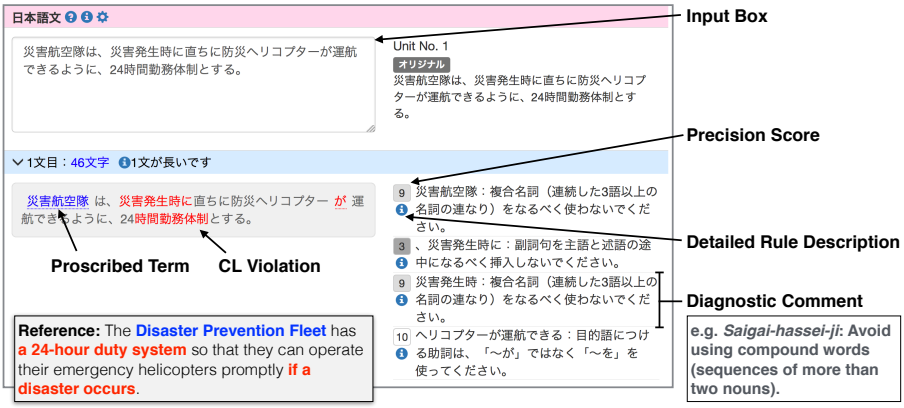


Figure 1. User interface

4. Controlled Language Authoring Assistant

4.1. Concept

The aim of our system is to help users create controlled STs. We designed a real-time, interactive system to check texts for conformity to CL rules and terminology during drafting or revision. Whenever a user enters input violating any of the operative CL rules or a term registered in a proscribed term list, the system alerts and supports the user in amending it.

Given that our target users—non-professional writers—tend to be unaccustomed to the principle of controlled writing and unfamiliar with writing tools, we need to provide support explanations and instructions. We therefore implemented several functions to assist the author's decision-making at each step of (re)writing, i.e., *detection*, *suggestion* and *correction*. Hitherto, CL 'checkers' have been deployed in two settings: post-hoc revision or rewriting (of legacy documents, for example) and assistance with 'drafting-from-scratch' (the more productive workflow). Our tool is designed to fit both scenarios.

4.2. Interface and Function

Figure 1 shows the system interface. The use scenario is as follows.

1. Users enter Japanese text in the **input box**.
2. The system automatically analyses each sentence and displays any detected **CL violations** in red and **proscribed terms** in blue (*detection*).
3. Users modify the problematic segments based on the **diagnostic comments**, referring to **detailed rule descriptions**, if needed.
4. For particular highlighted segments, the function offers alternative expressions displayed by clicking the segments (*suggestion*).
5. If the author clicks a suggestion, the segment in the input box is automatically replaced (*correction*).

4.3. Rule Implementation

To implement the CL violation detection function, we created surface part-of-speech pattern matching rules using the Japanese morphological analyser MeCab.[6] We then conducted a benchmark evaluation to calculate the precision and recall of the detection performance of each rule based on a previous study (Miyata et al., 2016). If the precision was below 0.4, we chose not to implement the rule. If the precision was above 0.4, we mapped it to a 10-point scale (**precision score**), which informs users how reliable the detection presented by the system is (shown in Figure 1). The rightmost column in Table 1 shows the 28 implemented rules with their precision scores. The terminology check function can similarly be implemented by simple string matching rules and integrated into the system. What is needed is to create a list of synsets of preferred and proscribed terms (Warburton, 2014).

5. Experimental Setup

Based on the ISO definition of usability introduced in Section 2, our questions for the system evaluation are: (1) Does the system help reduce CL violations and improve MT quality? (**effectiveness**); (2) Does the system help reduce time spent on controlled writing? (**efficiency**); (3) Is the system easy for non-professional writers to use and favourably accepted? (**satisfaction**)

To assess these three aspects, we designed a rewriting task in which two groups of participants were asked to amend Japanese source sentences violating CL rules and terminology, respectively with and without the aid of the system. Thus, we emulate the post-hoc revision setting. We (1-a) counted the number of corrected violations, (1-b) evaluated the MT quality, (2) measured the time taken to correct violations, and (3) gauged subjective satisfaction.

5.1. Task Design

Data: To count the number of corrected violations (**1-a. effectiveness**), we prepared a manually annotated dataset. We used sentence data extracted from Japanese municipal websites and selected 30 sentences to ensure that the dataset contained at least one violation of each of the 36 rules. Additionally, we artificially modified two proper nouns from the municipal domain into proscribed forms. The final dataset consisted of 67 violations of **CL-R** and 76 violations of **CL-S**, including two terminology violations of each.

Condition: For each of the two CL guidelines, **CL-R** and **CL-S**, one group of participants rewrites sentences with the sole aid of a print copy of the guideline and a term list[7] without access to the system's support functions (**control**), while the other

[6] MeCab: Yet Another Part-of-Speech and Morphological Analyzer, http://taku910.github.io/mecab/

[7] All participants were given the same term list, which enumerates 100 Japanese municipal terms including some proscribed forms. It was artificially created by the authors for the purpose of the evaluation.

1	Overall, I am satisfied with the ease of completing the tasks in this scenario.
2	Overall, I am satisfied with the amount of time it took to complete the tasks in this scenario.
3	Overall, I am satisfied with the support information (online-line help, messages, documentation) when completing the tasks.

Table 2. After-Scenario Questionnaire (ASQ)

1	I think that I would like to use this system frequently.
2	I found the system unnecessarily complex.
3	I thought the system was easy to use.
4	I think that I would need the support of a technical person to be able to use this system.
5	I found the various functions in this system were well integrated.
6	I thought there was too much inconsistency in this system.
7	I would imagine that most people would learn to use this system very quickly.
8	I found the system very cumbersome to use.
9	I felt very confident using the system.
10	I needed to learn a lot of things before I could get going with this system.

Table 3. System Usability Scale (SUS)

group can use the full assistance of the system (**treatment**). Thus, four conditions were prepared: (1) **C**ontrol group with CL-**R** (**CR**); (2) **C**ontrol group with CL-**S** (**CS**); (3) **T**reatment group with CL-**R** (**TR**); (4) **T**reatment group with CL-**S** (**TS**).

Procedure: Each participant is presented with a sentence in the input box of the system (see Figure 1) and is asked to amend any segments that violate CL rules or terminology, while maintaining the meaning of the source. All functions of the authoring assistant are disabled for the control group. As soon as the correction is completed, the resulting sentence is automatically saved and the participant proceeds to the next sentence. The system also records the elapsed time of the task for each sentence (**2. efficiency**).

Post-task questionnaire: To investigate **3. satisfaction** with the task and the system, we employed two standardised questionnaires widely used in usability studies: After-Scenario Questionnaire (ASQ) (Sauro and Lewis, 2012) and System Usability Scale (SUS) (Brooke, 1996). To evaluate how satisfied users were with the task, we used an ASQ with three questions on a seven-point Likert scale from '1: strongly disagree' to '7: strongly agree' (Table 2).[8] To evaluate the usability of the system itself, we used a SUS with ten questions on a five-point Likert scale from '1: strongly disagree' to '5: strongly agree' (Table 3).[9] Odd-numbered questions are worded positively, while even-numbered questions are worded negatively.

MT evaluation: To evaluate the resultant MT outputs (**1-b. effectiveness**), we conducted the traditional human evaluation. An evaluator judges each MT output in

[8]Since the questionnaire is originally in English, we translated it into Japanese. We also changed 'online help, messages, documentation' to 'documentation' in the third question for the control group as we did not provide them with any online help or messages.

[9]We also translated this into Japanese.

	Treatment (with system)			Control (without system)		
	TR	TS	Mean	CR	CS	Mean
Corrected violation (num.)	55.0	62.7	58.8	49.7	55.7	52.7
Missed violation (num.)	11.0	13.3	12.2	16.3	20.3	18.3
Correction rate (%)	83.3	82.5	82.9	75.3	73.3	74.3

Table 4. Effectiveness for each condition

terms of *fluency* from '5: Flawless English' to '1: Incomprehensible' and *adequacy* from '5: All' (of the meaning correctly correctly expressed) to '1: None'. The rewritten versions of the ST were translated by the intended MT systems, TransGateway (RBMT) or TexTra (SMT). As baseline and oracle outputs, we also translated the original ST and two sets of fully CL-compliant STs that we rewrote according to **CL-R** and **CL-S**.

5.2. Implementation

We recruited 12 university students, all of them native speakers of Japanese and regularly writing Japanese texts on computers, but none engaged in professional writing activity, such as technical writing or translation. They can thus be regarded as typical of our target end-users, i.e., non-professional writers. Three participants were randomly placed in each of the four conditions.

We first gave participants brief instructions for the rewriting task, then asked them to read through the CL guideline and the term list. In a preliminary session, each participant rewrote five example sentences to get used to the task and the system. In the task proper, each participant rewrote all 30 sentences, the order of which was randomised. Since this task imposes a heavy cognitive load on participants, we divided the 30 sentences into three sets, each of 10 sentences, and let participants take a short rest between the sets. After the main task, we asked them to answer ASQ and SUS,[10] and conducted a follow-up interview based on the responses.

For the MT evaluation task, we employed three native English speakers, who are engaged in Japanese-to-English translation, to evaluate all versions of the MT outputs.

6. Results and Discussions

6.1. Effectiveness

Table 4 shows the result of the effectiveness measures. Correction rate indicates the percentage of violations correctly amended throughout the task. On average, the treatment group achieved about a 9% higher correction rate than the control group, which an independent t-test found to be a significant difference ($t = -2.878, df = 10, p = .016$).

Detailed analysis of the results revealed that the correction rate for four rules—12, 14, 16 and 29 in Table 1—of the treatment group is lower than that of control group. We also noted that three of these four rules—12, 14 and 16—are not yet implemented. This implies that users tend to rely on the system and overlook any violations the

[10]Participants assigned to the control group answered only ASQ.

	RBMT				SMT			
	TR	CR	Original	Oracle	TS	CS	Original	Oracle
Fluency	2.74	2.72	2.36	2.87	2.58	2.57	2.32	2.70
Adequacy	3.20	3.19	2.80	3.46	2.89	2.80	2.62	3.34

Table 5. Result of MT quality evaluation

	Treatment (with system)			Control (without system)		
	TR	TS	Mean	CR	CS	Mean
Total time (sec.)	2405	2206	2306	3744	2844	3294
Time per sentence (sec.)	80.2	73.5	76.9	124.8	94.8	109.8
Time per correction (sec.)	43.7	35.2	39.5	75.3	51.1	63.2

Table 6. Efficiency for each condition

system does not detect. It is worthwhile pointing out that rules 12 and 14 can be implemented by utilising existing dictionaries and spell checkers, while rule 16 can be implemented by integrating deeper language tools such as parsers and chunkers, a task for future work.

Table 5 summarises the human evaluation results of fluency and adequacy by the MT systems. Comparing the control and treatment groups, we can see the fluency and adequacy for the MT outputs by the treatment group, **TR** and **TS**, are almost equal to or slightly higher than those by the control group, **CR** and **CS**. More notable is that the rewritten versions of ST, regardless of the help by the system, showed much higher MT quality than the original ST, which demonstrated our selected CL rule sets were indeed effective in improving machine translatability.

The oracle STs in which CL violations were corrected as much as possible, not surprisingly, exhibited the best MT quality. The adequacy scores achieved 3.46 for RBMT and 3.34 for SMT, well surpassing the score of '3: Much of the meaning correctly expressed'. The oracle scores can be regarded as the upper bound of the MT quality when our CL is properly applied. To achieve this point, further support for writers is needed.

6.2. Efficiency

Table 6 shows the results for the efficiency measures. Time per correction indicates the average time taken to correct one violation. We can observe that the treatment group corrected violations 30% faster than the control group. An independent t-test also found a significant difference in scores between the two groups ($t = 2.826, df = 10, p = .018$). This result demonstrates that our system greatly enhanced the efficiency of checking for and correcting violations.

6.3. Satisfaction

Finally, we look at the results of the two usability questionnaires and the follow-up interviews.

ASQ (satisfaction with the task) revealed no statistically significant difference between the control group and the treatment group, nonetheless we can see that for all

	Treatment (with system)							Control (without system)						
Q.	TR1	TR2	TR3	TS1	TS2	TS3	Mean	CR1	CR2	CR3	CS1	CS2	CS3	Mean
1	5	3	6	4	3	6	4.5	4	2	3	5	6	3	3.8
2	5	4	6	5	6	4	5.0	3	2	3	5	6	5	4.0
3	6	5	7	4	5	5	5.3	6	6	4	5	5	3	4.8

Table 7. Result of Questionnaire ASQ (satisfaction with the task)

Q.	TR1	TR2	TR3	TS1	TS2	TS3	Mean
1	4	3	5	3	5	4	4.0
2	1	2	1	2	2	2	1.7
3	5	3	5	2	4	4	3.8
4	3	4	1	4	4	3	3.2
5	2	4	5	2	5	4	3.7
6	4	2	1	2	1	2	2.0
7	4	4	5	2	4	4	3.8
8	1	2	1	3	2	2	1.8
9	3	3	5	3	5	4	3.8
10	4	3	1	4	4	2	3.0
Score	70	68	100	54	80	78	75.0

Table 8. Result of Questionnaire SUS (satisfaction with the system)

three questions the mean scores of the treatment group are higher than those of the control group. It is also evident that, while there are only two negative answers (i.e., Likert scale of 1–3) from the treatment group, from the control group there are seven. This suggests that participants assisted by the system were generally satisfied with the task completion.

The SUS results (satisfaction with the system) pertain only to the treatment group. To calculate an overall SUS score ranging from 0 to 100 in 2-point increments, we inverted the scale of even-numbered questions from 1–5 to 5–1, and then doubled the sum of all the scores (see the bottom row of Table 8). The higher the score, the more usable the system was judged to be. The mean score is 75.0, which is reasonably high.

It is important to note that most participants agreed with Question 4 ('I think that I would need the support of a technical person to be able to use this system') and Question 10 ('I needed to learn a lot of things before I could get going with this system') with scores of 3–5. Both questions relate to the 'learnability' of the system. The follow-up interview results also revealed that some participants were unable to use the various support functions, such as suggestions of alternative expressions. Moreover, we found that some participants failed to correct proscribed terms highlighted in blue, even though they recognised them, simply because they forgot what the blue highlighting indicated. To make the system more effective, we need to provide more detailed user instructions and further simplify the interface.

7. Conclusions and Future Work

We have presented an experiment to assess the usability of a CL authoring assistant developed to support non-professional writers in checking conformity to CL

rules and terminology. Based on the ISO definition of usability, we assessed three aspects: effectiveness, efficiency and user satisfaction. Comparing two groups of participants—respectively, with and without the help of the system—we reached the following conclusions:

- The system helped reduce rule violations by about 9% (**effectiveness**).
- The system helped reduce the time taken to correct violations by more than 30% (**efficiency**).
- Participants were generally satisfied with the system, although some did not find the functions and interface easy to learn (**satisfaction**).

Our system now implements optimal CL rule sets individually tailored to two MT systems and the STs written in accordance with the rule sets proved to greatly improve machine translatability. The usability evaluation demonstrates that the system significantly enhances the efficiency of CL authoring by non-professional writers. This opens the promising prospect of practical joint deployment of CL and MT in real world scenarios.

The MT evaluation results of the oracle ST suggested that there is still room for improvement in MT quality. In future research, we plan to utilise existing language resources and tools to implement the remaining CL rules and so further assist authors in eliminating CL violations. We will also improve the interface and user documentation so that users take effective advantage of the full range of available functions.

Acknowledgements

This work was supported by JSPS KAKENHI Grant Number 16J11185 and by the Research Grant Program of KDDI Foundation, Japan.

Bibliography

Aikawa, Takako, Lee Schwartz, Ronit King, Monica Corston-Oliver, and Carmen Lozano. Impact of Controlled Language on Translation Quality and Post-Editing in a Statistical Machine Translation Environment. In *Proc. of MT Summit*, pages 1–7, 2007.

Alabau, Vicent, Luis A. Leiva, Daniel Ortiz-Martínez, and Francisco Casacuberta. User Evaluation of Interactive Machine Translation Systems. In *Proc. of EAMT*, pages 20–23, 2012.

Bernth, Arendse and Claudia Gdaniec. MTranslatability. *Machine Translation*, 16(3):175–218, 2001.

Bojar, Ondřej, Rajen Chatterjee, Christian Federmann, Barry Haddow, Matthias Huck, Chris Hokamp, Philipp Koehn, Varvara Logacheva, Christof Monz, Matteo Negri, Matt Post, Carolina Scarton, Lucia Specia, and Marco Turchi. Findings of the 2015 Workshop on Statistical Machine Translation. In *Proc. of WMT*, pages 1–46, 2015.

Brooke, John. SUS: A Quick and Dirty Usability Scale. In Jordan, P. W., B. Thomas, B. A. Weerdmeester, and A. L. McClelland, editors, *Usability Evaluation in Industry*, pages 189–194. Taylor and Francis, London, 1996.

Castilho, Sheila, Sharon O'Brien, Fabio Alves, and Morgan O'Brien. Does Post-editing Increase Usability? A Study with Brazilian Portuguese as Target Language. In *Proc. of EAMT*, pages 183–190, 2014.

Doherty, Stephen and Sharon O'Brien. A User-Based Usability Assessment of Raw Machine Translated Technical Instructions. In *Proc. of AMTA*, 2012.

Doherty, Stephen and Sharon O'Brien. Assessing the Usability of Raw Machine Translated Output: A User-Centered Study Using Eye Tracking. *International Journal of Human Computer Interaction*, 30(1):40–51, 2013.

Isahara, Hitoshi. Translation Technology in Japan. In Chan, Sin-Wai, editor, *Routledge Encyclopedia of Translation Technology*, pages 315–326. Routledge, New York, 2015.

ISO. ISO 9241-210:2010 Ergonomics of Human-System Interaction—Part 210: Human-Centred Design for Interactive Systems, 2010.

Kittredge, Richard. Sublanguages and Controlled Languages. In Mitkov, Ruslan, editor, *Oxford Handbook of Computational Linguistics*, pages 430–437. Oxford University Press, Oxford, 2003.

Kuhn, Tobias. A Survey and Classification of Controlled Natural Languages. *Computational Linguistics*, 40(1):121–170, 2014.

Mitamura, Teruko, Kathryn Baker, Eric Nyberg, and David Svoboda. Diagnostics for Interactive Controlled Language Checking. In *Proc. of EAMT/CLAW*, pages 237–244, 2003.

Miyata, Rei, Anthony Hartley, Cécile Paris, Midori Tatsumi, and Kyo Kageura. Japanese Controlled Language Rules to Improve Machine Translatability of Municipal Documents. In *Proc. of MT Summit*, pages 90–103, 2015.

Miyata, Rei, Anthony Hartley, Cécile Paris, and Kyo Kageura. Evaluating and Implementing a Controlled Language Checker. In *Proc. of CLAW*, pages 30–35, 2016.

Nyberg, Eric, Teruko Mitamura, and Willem-Olaf Huijsen. Controlled Language for Authoring and Translation. In Somers, Harold, editor, *Computers and Translation: A Translator's Guide*, pages 245–281. John Benjamins, Amsterdam, 2003.

O'Brien, Sharon. Controlling Controlled English: An Analysis of Several Controlled Language Rule Sets. In *Proc. of EAMT/CLAW*, pages 105–114, 2003.

O'Brien, Sharon and Johann Roturier. How Portable are Controlled Language Rules? In *Proc. of MT Summit*, pages 345–352, 2007.

Pym, Peter. Pre-editing and the Use of Simplified Writing for MT. In Mayorcas, Pamela, editor, *Translating and the Computer 10*, pages 80–95. Aslib, London, 1990.

Rascu, Ecaterina. A Controlled Language Approach to Text Optimization in Technical Documentation. In *Proc. of KONVENS*, pages 107–114, 2006.

Sauro, Jeff and James R. Lewis. *Quantifying the User Experience: Practical Statistics for User Research*. Morgan Kaufmann, Burlington, 2012.

Thicke, Lori. Improving MT results: A study. *Multilingual*, January/February:37–40, 2011.

Warburton, Kara. Developing Lexical Resources for Controlled Authoring Purposes. In *Proc. of LREC Workshop: Controlled Natural Language Simplifying Language Use*, pages 90–103, 2014.

Address for correspondence:
Rei Miyata
miyata@nuee.nagoya-u.ac.jp
Graduate School of Engineering, Nagoya University
Furo-cho, Chikusa-ku, Nagoya, 464-8601, Japan

PBML

The Prague Bulletin of Mathematical Linguistics
NUMBER 108 JUNE 2017 159–170

A Linguistic Evaluation of Rule-Based, Phrase-Based, and Neural MT Engines

Aljoscha Burchardt,[a] Vivien Macketanz,[a] Jon Dehdari,[a] Georg Heigold,[a] Jan-Thorsten Peter,[b] Philip Williams[c]

[a] German Research Center for Artificial Intelligence (DFKI)
[b] RWTH Aachen University
[c] University of Edinburgh

Abstract

In this paper, we report an analysis of the strengths and weaknesses of several Machine Translation (MT) engines implementing the three most widely used paradigms. The analysis is based on a manually built test suite that comprises a large range of linguistic phenomena. Two main observations are on the one hand the striking improvement of an commercial online system when turning from a phrase-based to a neural engine and on the other hand that the successful translations of neural MT systems sometimes bear resemblance with the translations of a rule-based MT system.

1. Introduction

Test suites are a familiar tool in NLP in areas such as grammar checking, where one may wish to ensure that a parser is able to analyse certain sentences correctly or test the parser after changes to see if it still behaves in the expected way. In contrast to a "real-life" corpus the input in a test suite may well be made-up or edited to isolate and illustrate issues.

Apart from several singular attempts (King and Falkedal, 1990; Isahara, 1995; Koh et al., 2001, etc.) broadly-defined test suites have not generally been used in MT research. One of the reasons for this might be the fear that the performance of statistical MT systems depends so much on the particular input data, parameter settings, etc., that final conclusions about the errors they make, particularly about the different

 Corresponding author: aljoscha.burchardt@dfki.de
Cite as: Aljoscha Burchardt, Vivien Macketanz, Jon Dehdari, Georg Heigold, Jan-Thorsten Peter, Philip Williams. A Linguistic Evaluation of Rule-Based, Phrase-Based, and Neural MT Engines. The Prague Bulletin of Mathematical Linguistics No. 108, 2017, pp. 159–170. doi: 10.1515/pralin-2017-0017.

reasons (e.g., length of n-grams, missing training examples), are difficult to obtain. A related concern is that statistical MT systems are designed to maximise scores on test corpora that are comparable to the training/tuning corpora and that it is therefore unreliable to test these systems in different settings. While these concerns may hold for systems trained on very narrowly-defined domains, genres, and topics (such as biomedical patent abstracts), in fact many systems are trained on large amounts of data covering mixed sources and are expected to generalize to some degree.

A last reason might be that "correct" MT output cannot be specified in the same way as the output of other language processing tasks like parsing or fact extraction where the expected results can be more or less clearly defined. Due to the variation of language, ambiguity, etc., checking and evaluating MT output can be almost as difficult as the translation itself. Still, people have tried to automatically classify errors comparing MT output to reference translations or post-edited MT output using tools like Hjerson (Popovic, 2011).

In narrow domains there seems to be interest in detecting differences between systems and within the development of one system, e.g., in terms of verb-particle constructions (Schottmüller and Nivre, 2014) or pronouns (Guillou and Hardmeier, 2016). Bentivogli et al. (2016) performed a comparison of neural- with phrase-based MT systems on IWSLT data using a coarse-grained error typology. Neural systems have been found to make fewer morphological, lexical and word-order errors.

Below, we present a pioneering effort to address translation barriers in a systematic fashion. We are convinced that testing of system performance on error classes leads to insights that can guide future research and improvements of systems. By using test suites, MT developers will be able to see how their systems perform compared to scenarios that are likely to lead to failure and can take corrective action.

This paper is structured as follows: After the general introduction (Section 1), Section 2 will briefly introduce the test suite we have used in the experiments reported in Section 3. Section 4 concludes the paper.

2. The Test Suite

The experiments reported below are based on a test suite for MT Quality we are currently building for the language pair English – German in the QT21 project. The test suite itself will be described in more detail in a future publication. In brief, it contains segments selected from various parallel corpora and drawn from other sources such as grammatical resources, e.g., the TSNLP Grammar Test Suite (Lehmann et al., 1996) and online lists of typical translation errors.

Each test sentence is annotated with the phenomenon category and the phenomenon it represents. An example showing these fields can be seen in Table 1 with the first column containing the source segment and the second and third column containing the phenomenon category and the phenomenon, respectively. The fourth column shows the translation given by the old Google Translate system and the last column

contains a post-edit of the MT output that is created by making as few changes as possible. In our latest version of the test suite, we have a collection of about 5,000 segments per language direction that are classified in about 15 categories (most of them similar in both language directions) and about 120 phenomena (many of them similar but also some differing, as they are language-specific). Depending on the nature of the phenomenon, each is represented by at least 20 test segments in order to guarantee for a balanced test set. The categories cover a wide range of different grammatical aspects that might or might not lead to translation difficulties for a MT system. Currently, we are still in the process of optimising our test segments and working on an automatic solution for the evaluation.

Source	**Phenomenon Category**	**Phenomenon**	**Target (raw)**	**Target (edited)**
Lena machte sich früh vom Acker.	MWE	Idiom	Lena [left the field early].	Lena left early.
Lisa hat Lasagne gemacht, sie ist schon im Ofen.	Non-verbal agreement	Coreference	Lisa has made lasagne, [she] is already in the oven.	Lisa has made lasagna, it is already in the oven.
Ich habe der Frau das Buch gegeben.	Verb tense/ aspect/ mood	Ditransitive - perfect	I [have] the woman of the Book.	I have given the woman the book.

Table 1. Example test suite entries German→English (simplified for display purposes).

For the experiments presented here, we have used a preliminary version of our test suite (ca. 800 items per language direction, to a large extent verb paradigms) to include the changes of Google Translate which has recently been switched from a phrase-based to neural approach according to the companies' publications. There are more than 100 different linguistic phenomena that we investigated in this version of the test suite in each language direction. In this preliminary version, the number of instances reported in the experiments below strongly varies among the categories (as well as between the languages).

3. Evaluating PBMT, NMT, and RBMT Engines and an Online System

3.1. System Description

We have evaluated several engines from leading machine translation research groups and a commercial rule-based system on the basis of the very same test suite version to be able to compare performance with the leading online system that has recently switched to a neural model. We included a number of different NMT systems with different properties and levels of sophistication to shed light on how these

new types of systems perform on the different kinds of phenomena. Below, we will briefly describe the systems.

O-PBMT Old version of Google Translate (web interface, Feb. 2016).

O-NMT New version of Google Translate (web interface, Nov. 2016).

OS-PBMT Open-source phrase-based system that primarily uses a default configuration to serve as a baseline. This includes a 5-gram modified Kneser-Ney language model, `mkcls` and MGiza for alignment, GDFA phrase extraction with a maximum phrase length of five, `msd-bidi-fe` lexical reordering, and the Moses decoder (Koehn et al., 2007). The WMT'16 data was Moses-tokenized and normalized, truecased, and deduplicated.

DFKI-NMT Barebone neural system from DFKI. The MT engine is based on the encoder-decoder neural architecture with attention. The model was trained on the respective parallel WMT'16 data.

ED-NMT Neural system from U Edinburgh. This MT engine is the top-ranked system that was submitted to the WMT '16 news translation task (Sennrich et al., 2016). The system was built using the Nematus toolkit.[1] Among other features, it uses byte-pair encoding (BPE) to split the vocabulary into subword units, uses additional parallel data generated by back-translation, uses an ensemble of four epochs (of the same training run), and uses a reversed right-to-left model to rescore n-best output.

RWTH-NMT NMT-system from RWTH (only used for German – English experiments). This system is equal to the ensemble out of 8 NMT systems optimized on TEDX used in the (Peter et al., 2016) campaign. The eight networks used make use of subwords units and are finetuned to perform well on the IWSLT 2016 MSLT German to English task.

RBMT Commercial rule-based system Lucy (Alonso and Thurmair, 2003).

3.2. Evaluation Procedure

In order to evaluate a system's performance on the categories in the test suite, we concentrate solely on the phenomenon in the respective sentence and disregard other errors. This means that we have to determine whether a translation error is linked to the phenomenon under examination or if it is independent from the phenomenon. If the former is the case, the segment will be validated as incorrect. If, however, the error in the translation can not be traced back to the phenomenon, the segment will be counted as correct.

Currently, the system outputs are being automatically compared to a "reference translation" which is, in fact, a post-edit of the O-PBMT output as those were the very first translations to be generated and evaluated when we started building the test suite (see description of the test suite in Section 2 and Table 1). In a second step,

[1] `https://github.com/rsennrich/nematus`

	#	O-PBMT	O-NMT	RBMT	OS-PBMT	DFKI-NMT	RWTH-NMT	ED-NMT
Ambiguity	17	12%	35%	**42%**	24%	35%	12%	35%
Composition	11	27%	**73%**	55%	27%	45%	45%	**73%**
Function words	19	5%	**68%**	21%	11%	26%	**68%**	42%
LDD & interrogative	66	12%	**79%**	62%	21%	36%	55%	52%
MWE	42	14%	**36%**	7%	21%	10%	12%	19%
NE & terminology	25	48%	48%	40%	**52%**	40%	48%	40%
Subordination	36	22%	**58%**	50%	31%	47%	42%	31%
Verb tense/aspect/mood	529	59%	80%	**91%**	52%	53%	74%	63%
Verb valency	32	16%	**50%**	44%	13%	47%	38%	**50%**
Sum	777	358	567	583	337	367	490	435
Average		46%	73%	**75%**	43%	47%	63%	56%

Table 2. Results of German – English translations. Boldface indicates best system(s) on each category (row).

all the translations that do not match the "reference" are manually evaluated by a professional linguist since the translations might be very different from the O-PBMT post-edit but nevertheless correct. As this is a very time-consuming process, we are currently working on automating this evaluation process by providing regular expressions for various possible translation outputs – naturally, only focusing on the phenomenon under investigation.

We refrain from creating an independent reference as we think that generating the regular expressions that focus solely on the phenomena instead is the more sophisticated solution in this context. As a consequence, we cannot compute automatic scores like BLEU. We do not see this as a disadvantage as with the test suite we want to focus rather on gaining insights about the nature of translations than on how well translations match a certain reference.

3.3. Results German – English

Table 2 shows the results for the translations from German to English from the different systems on the categories. The second column in the table ("#") contains the number of instances per category. As the distribution of examples per category in this old version of our test suite was very unbalanced with some categories having only very few examples, some more categories we tested were excluded from the analysis we present here.

Before we discuss the results, we want to point out that the selection of phenomena and the number of instances used here is not representative of their occurrence in

corpora. Consequently, it can not be our goal to find out which of the systems is the globally "best" or winning system. Our goal is to check and illustrate the strengths and weaknesses of system (types) with respect to the range of phenomena we cover with this version of the test suite. Using this evaluation approach, researchers and system developers ideally can form hypotheses about the reasons why certain errors happen (systematically) and can come up with a prioritised strategy for improving the systems. Our ultimate goal is to represent all phenomena relevant for translation in our test suite.

Coming to the analysis, it is first of all striking how much better the neural version of Google Translate (O-NMT) is as compared to its previous phrase-based version (O-PBMT). Interestingly, the O-NMT and the RBMT – two very different approaches – are the best-performing systems on average, achieving almost the same amount of correct translations on average, i.e., 73%, resp. 75%, but looking at the scores of the categories reveals that the performance of the two systems regarding the categories is in fact very diverse. While the O-NMT system is the most-frequent best-performing system per phenomenon, as it is best on composition, function words, long distance dependency (LDD) & interrogative, multi-word expressions (MWE), subordination and verb valency, the RBMT is only the best system on ambiguity[2] and verb tense/aspect/mood. The high number of instances of the latter category leads to the high average score of the RBMT system, as verb paradigms are part of the linguistic information RBMT systems are based on.

The OS-PMBT reaches the lowest average score, but it is nevertheless the best-performing system on named entities (NE) & terminology. The DFKI-NMT system reaches a higher average score than the PBMT system (four percentage points more). The RWTH-NMT is (along with the O-NMT) the best-performing system on function words. On average it reaches 63% of correct translations. The ED-NMT outrules (also along with the O-NMT) the other systems on composition and verb valency and reaches 56% correct translations on average.

In order to see if we find some interesting correlations that might serve as a preview for more extensive analyses with a more solid and balanced amount of test segments in the future, we have calculated Pearson's coefficient over the phenomenon counts (being aware that we are dealing with very small numbers here). As the correlations for the direction English – German were higher and for space reasons, we will show the numbers only for the other direction in the following Subsection to give an indication about possible future work.

One general impression that will also be supported by the examples below is that NMT seems to learn some capabilities that the RBMT system has. It may lead to the speculation that NMT indeed learns something like the rules of the language. This, however, needs more intensive investigation. Another interesting observation is that

[2]The good performance of RBMT on ambiguity can be explained by the very small number of items and it is more or less accidental that the preferred readings were the ones the RBMT has coded in its lexicon.

the RWTH-NMT system has a lower overall correlation with the other NMT systems. This might be because it has also been trained and optimised on transcripts of spoken language as opposed to the other systems trained solely on written language.

The following examples depict interesting findings from the analysis and comparison of the different systems. When a system created a correct output (on the respective category), the system's name is marked in boldface.

(1)	**Source**:	Warum hörte Herr Muschler mit dem Streichen auf?
	Reference:	Why did Mr. Muschler stop painting?
	O-PBMT:	Why heard Mr Muschler on with the strike?
	O-NMT:	Why did Mr. Muschler stop the strike?
	RBMT:	Why did Mr Muschler stop with the strike?
	OS-PBMT:	Why was Mr Muschler by scrapping on?
	DFKI-NMT:	Why did Mr Muschler listen to the rich?
	RWTH-NMT:	Why did Mr. Muschler listen to the stroke?
	ED-NMT:	Why did Mr. Muschler stop with the stump?

Example (1) contains a phrasal verb and belongs to the category composition. German phrasal verbs have the characteristics that their prefix might be separated from the verb and move to the end of the sentence in certain constructions, as it has happened in example (1) with the prefix *auf* being separated from the rest of the verb *hören*. The verb *aufhören* means *to stop*, but the verb *hören* without the prefix simply means *to listen*. Thus, phrasal verbs might pose translations barriers in MT when the system translates the verb separately not taking into account the prefix at the end of the sentence. The output of the O-PBMT, DFKI-NMT and RWTH-NMT indicates that this might have happened. The O-NMT, RBMT and the ED-NMT correctly translate the verb which could mean that more context (and thus, including the prefix *auf* at the end of the sentences) was taken into account for the generation of the output.

(2)	**Source**:	Warum macht der Tourist drei Fotos?
	Reference:	Why does the tourist take three fotos?
	O-PBMT:	Why does the tourist three fotos?
	O-NMT:	Why does the tourist make three fotos?
	RBMT:	Why does the tourist make three fotos?
	OS-PBMT:	Why does the tourist three fotos?
	DFKI-NMT:	Why does the tourist make three fotos?
	RWTH-NMT:	Why is the tourist taking three fotos?
	ED-NMT:	Why does the tourist make three fotos?

One of the phenomena in the category LDD & interrogative is wh-movement. It is for example involved in wh-questions, like in the sentence in (2). A wh-question in English is usually built with an auxiliary verb and a full verb, e.g., wh-word + *to*

have/to be/to do + full verb. In German on the other hand, an auxiliary verb is not necessarily needed. This fact might lead to translation difficulties, as can be seen in (2), where the O-PBMT and the OS-PBMT treat the verb *does* as a full verb instead of an auxiliary verb. All the other systems translate the question with two verbs, however, except for the RWTH-NMT, they all mistranslate *ein Foto machen* as *to make a foto* (literal translation) instead of *to take a foto*. Nevertheless, these translations count as correct, since they do contain an auxiliary verb + a full verb.

(3)
Source:	Die Arbeiter müssten in den sauren Apfel beißen.
Reference:	The workers would have to bite the bullet.
O-PBMT:	The workers would have to bite the bullet.
O-NMT:	The workers would have to bite into the acid apple.
RBMT:	The workers would have to bite in the acid apple.
OS-PBMT:	The workers would have to bite the bullet.
DFKI-NMT:	Workers would have to bite in the acid apple.
RWTH-NMT:	The workers would have to bite into the clean apple.
ED-NMT:	The workers would have to bite in the acidic apple.

Idioms are an interesting phenomenon within the category MWE. The meaning of an idiom in one language can not be transferred to another language by simply translating the separate words, as the meaning of these multi-word units goes beyond the meaning of the separate words. As a consequence, idioms have to be transferred to another language as a whole. For German <> English it is often the case that an idiom in one language can be transferred to another idiom in the other language. This is also the case in example (3). The German idiom *in den sauren Apfel beißen* can be translated as *to bite the bullet*. Only the two PBMT system correctly translate this idiom, the other systems all give a literal translation - with the RWTH-NMT translating *sauren* as *clean* instead of *acid(ic)* like the other systems, probably not knowing the word *sauren* and instead translating the similar word *sauberen*. This is one example where a phrase-based approach has a real advantage (if the phrase was in the training data).

(4)
Source:	Wie kann ich die Farbe, mit der ich arbeite, ändern?
Reference:	How can I change the color I am working with?
O-PBMT:	How can I change the color with which I work to change?
O-NMT:	How can I change the color with which I work?
RBMT:	How can I change the color with which I work?
OS-PBMT:	How can I change the colour, with whom i work, change?
DFKI-NMT:	How can I change the color I work with?
RWTH-NMT:	How can I change the color I work with?
ED-NMT:	How can I change the color I work with?

The sentence in (4) contains a relative clause which belongs to the category subordination. Relative clauses in English can, but do not have to contain a relative pronoun. The outputs in (4) show both properties. The O-PBMT and the OS-PBMT double the verb *change*, the remaining systems correctly translate the relative clause.

(5)
Source:	Ich hätte nicht lesen gedurft.
Reference:	I would not have been allowed to read.
O-PBMT:	I would not have been allowed to read.
O-NMT:	I should not have read.
RBMT:	I would not have been allowed to read.
OS-PBMT:	I would not have read gedurft.
DFKI-NMT:	I would not have been able to read.
RWTH-NMT:	I wouldn't have read.
ED-NMT:	I wouldn't have read.

Verb paradigms (verb tense/aspect/mood) make up about one third of the whole test suite. Example (5) shows a sentence with a negated modal verb, in the tense pluperfect subjunctive II. This is a quite complex construction, thus it is not surprising that only few systems correctly translate the sentence. As might be expected, one of them is the RBMT system. The second one is the O-PBMT. The neural version of this system on the other hand does not correctly produce the output.

3.4. Results English – German

The results for the English – German translations can be found in Table 3. For this language direction, only five systems were available instead of seven like for the other direction. As in the analysis for the other language direction, we excluded the categories that had too few instances from the table. Nevertheless, similarities between the categories of both language directions can be found.

As in the German – English translations, the RBMT system performs best of all systems on average, reaching 83%. It performs best of all systems on verb tense/aspect/mood and verb valency. The second-best system is – just like in the other language direction but with a greater distance (seven percentage points less on average, namely 76%) – the O-NMT. The O-NMT shows quite contrasting results on the different categories, compared to RBMT: it outrules (most of) the other systems on the remaining categories, i.e., on coordination & ellipsis, LDD & interrogative, MWE, NE & terminology, special verb types and subordination.

The third-best system on average is the ED-NMT system. It reaches an average of 61% correct translations. The other remaining NMT system, the barebone DFKI-NMT system, reaches 11 percentage points less on average than the ED-NMT, for it reaches 50%. But it outrules the other systems on subordination along with O-NMT. The system with the lowest average score is the previous version of Google Translate,

	#	O-PBMT	O-NMT	RBMT	DFKI-NMT	ED-NMT
Coordination & ellipsis	17	6%	**47%**	29%	24%	35%
LDD & interrogative	70	19%	**61%**	54%	41%	40%
MWE	42	21%	**29%**	19%	21%	26%
NE & terminology	20	25%	**80%**	40%	45%	65%
Special verb types	14	14%	**86%**	79%	29%	64%
Subordination	35	11%	**71%**	54%	**71%**	69%
Verb tense/aspect/mood	600	41%	82%	**96%**	53%	66%
Verb valency	22	36%	59%	**68%**	64%	59%
Sum	820	287	622	679	410	499
Average		35%	76%	**83%**	50%	61%

Table 3. Results of English – German translations. Boldface indicates best system(s) on each category (row).

Correlations	O-PBMT	O-NMT	RBMT	DFKI-NMT	ED-NMT
O-PBMT	1.00				
O-NMT	0.34	1.00			
RBMT	0.39	0.55	1.00		
DFKI-NMT	0.28	0.29	0.36	1.00	
ED-NMT	0.30	0.33	0.43	0.55	1.00

Table 4. Overall correlation of English – German systems

namely the O-PBMT. With 35% on average, it reaches less than half of the score of the O-NMT.

The results of the calculation of the Pearson's coefficient can be found in Table 4. Only categories with more than 25 observations had their correlation analysed. For the interpretation, we used a rule-of-thumb mentioned in the literature[3].

In the overall correlation, RBMT has a moderate correlation with O-NMT, which might be traced back to the fact that these are the two systems that correctly translate most of the test segments, compared to the other systems. The two neural systems, DFKI-NMT and ED-NMT, also have moderate correlations. All the other systems have weak correlation with each other.

Again, for the small and unbalanced numbers of samples, we do not want to put too much emphasis on the observations regarding correlations. This type of analysis might, however, become more informative in future work.

[3] http://www.dummies.com/education/math/statistics/how-to-interpret-a-correlation-coefficient-r

4. Conclusions and Outlook

While the selection of test items/categories and even more the selection of examples we discussed provides a selective view on the performance of the system, we are convinced that this type of quantitative and qualitative evaluation provides valuable insights and ideas for improvement of the systems, e.g., by adding linguistic knowledge in one way or another. Two main observations we want to repeat here is the striking improvement of the commercial online system when turning from a phrase-based to a neural engine. A second observation is that the successful translations of some NMT systems often bear resemblance with the translations of the RBMT system. Hybrid combinations or pipelines where RBMT systems generate training material for NMT systems seem a promising future research direction to us.

While the extracted examples above give very interesting insights on the systems' performances on the categories, these are only more or less random spot tests. However, taking a close look at the separate phenomena at a larger scale and in more detail will lead to more general, systematic observations. This is what we aim to do with our current version of the test suite which is therefore much more extensive and systematic and therefore also allows for more general observations and more quantitative statements in future experiments.

Our ultimate goal is to automate the test suite testing. To this end, we are currently working on a method that is using regular expressions for automatically checking the output of engines on the test suite. The idea is to manually provide positive and negative tokens for each test item that can range from expected words in case of disambiguation up to, verbs and their prefixes with wild cards in between up to complete sentences in the case of verb paradigms.

Acknowledgements

This research is supported by the EC's Horizon 2020 research and innovation programme under grant agreements no. 645452 (QT21).

Bibliography

Alonso, Juan A and Gregor Thurmair. The Comprendium Translator system. In *Proceedings of the Ninth Machine Translation Summit*. International Association for Machine Translation (IAMT), 2003.

Bentivogli, Luisa, Arianna Bisazza, Mauro Cettolo, and Marcello Federico. Neural versus Phrase-Based Machine Translation Quality: a Case Study. *CoRR*, abs/1608.04631, 2016.

Guillou, Liane and Christian Hardmeier. PROTEST: A Test Suite for Evaluating Pronouns in Machine Translation. In Chair), Nicoletta Calzolari (Conference, Khalid Choukri, Thierry Declerck, Sara Goggi, Marko Grobelnik, Bente Maegaard, Joseph Mariani, Helene Mazo, Asuncion Moreno, Jan Odijk, and Stelios Piperidis, editors, *Proceedings of the Tenth Interna-*

tional Conference on Language Resources and Evaluation (LREC 2016), Paris, France, may 2016. European Language Resources Association (ELRA). ISBN 978-2-9517408-9-1.

Isahara, Hitoshi. JEIDA's test-sets for quality evaluation of MT systems: Technical evaluation from the developer's point of view. In *Proceedings of the MT Summit V. Luxembourg*, 1995.

King, Margaret and Kirsten Falkedal. Using Test Suites in Evaluation of Machine Translation Systems. In *Proceedings of the 13th Conference on Computational Linguistics - Volume 2*, COLING '90, pages 211–216, Stroudsburg, PA, USA, 1990. Association for Computational Linguistics.

Koehn, Philipp, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondřej Bojar, Alexandra Constantin, and Evan Herbst. Moses: Open Source Toolkit for Statistical Machine Translation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions*, pages 177–180, Prague, Czech Republic, June 2007. Association for Computational Linguistics.

Koh, Sungryong, Jinee Maeng, Ji-Young Lee, Young-Sook Chae, and Key-Sun Choi. A test suite for evaluation of English-to-Korean machine translation systems. In *Proceedings of the MT Summit VIII. Santiago de Compostela, Spain*, 2001.

Lehmann, Sabine, Stephan Oepen, Sylvie Regnier-Prost, Klaus Netter, Veronika Lux, Judith Klein, Kirsten Falkedal, Frederik Fouvry, Dominique Estival, Eva Dauphin, Hervé Compagnion, Judith Baur, Lorna Balkan, and Doug Arnold. TSNLP - Test Suites for Natural Language Processing. In *Proceedings of the 16th International Conference on Computational Linguistics*, pages 711–716, 1996.

Peter, Jan-Thorsten, Andreas Guta, Nick Rossenbach, Miguel Graça, and Hermann Ney. The RWTH Aachen Machine Translation System for IWSLT 2016. In *International Workshop on Spoken Language Translation*, Seattle, USA, Dec. 2016.

Popovic, Maja. Hjerson: An Open Source Tool for Automatic Error Classification of Machine Translation Output. *The Prague Bulletin of Mathematical Linguistics*, 96:59–68, 10 2011.

Schottmüller, Nina and Joakim Nivre. Issues in Translating Verb-Particle Constructions from German to English. In *Proc. of the 10th Workshop on Multiword Expressions (MWE)*, pages 124–131, Gothenburg, Sweden, April 2014. Association for Computational Linguistics.

Sennrich, Rico, Barry Haddow, and Alexandra Birch. Edinburgh Neural Machine Translation Systems for WMT 16. *CoRR*, abs/1606.02891, 2016.

Address for correspondence:
Aljoscha Burchardt
aljoscha.burchardt@dfki.de
German Research Center for Artificial Intelligence (DFKI)
Language Technology Lab, Alt-Moabit 91c, 10559 Berlin, Germany

PBML

The Prague Bulletin of Mathematical Linguistics
NUMBER 108 JUNE 2017 171–182

Pre-Reordering for Neural Machine Translation: Helpful or Harmful?

Jinhua Du, Andy Way

ADAPT Centre, School of Computing, Dublin City University

Abstract

Pre-reordering, a preprocessing to make the source-side word orders close to those of the target side, has been proven very helpful for statistical machine translation (SMT) in improving translation quality. However, is it the case in neural machine translation (NMT)? In this paper, we firstly investigate the impact of pre-reordered source-side data on NMT, and then propose to incorporate features for the pre-reordering model in SMT as input factors into NMT (factored NMT). The features, namely parts-of-speech (POS), word class and reordered index, are encoded as feature vectors and concatenated to the word embeddings to provide extra knowledge for NMT. Pre-reordering experiments conducted on Japanese↔English and Chinese↔English show that pre-reordering the source-side data for NMT is redundant and NMT models trained on pre-reordered data deteriorate translation performance. However, factored NMT using SMT-based pre-reordering features on Japanese→English and Chinese→English is beneficial and can further improve by 4.48 and 5.89 relative BLEU points, respectively, compared to the baseline NMT system.

1. Introduction

In recent years, NMT has achieved impressive progress (Kalchbrenner and Blunsom, 2013; Cho et al., 2014; Sutskever et al., 2014; Bahdanau et al., 2015). The state-of-the-art NMT model employs an encoder–decoder architecture with an attention mechanism, in which the encoder summarizes the source sentence into a vector representation, and the decoder produces the target string word by word from vector representations, and the attention mechanism learns the soft alignment of a target word against source words (Bahdanau et al., 2015). NMT systems have outperformed the state-of-the-art SMT model on various language pairs in terms of translation qual-

 Corresponding author: andy.way@adaptcentre.ie
Cite as: Jinhua Du, Andy Way. Pre-Reordering for Neural Machine Translation: Helpful or Harmful?. The Prague Bulletin of Mathematical Linguistics No. 108, 2017, pp. 171–182. doi: 10.1515/pralin-2017-0018.

ity (Luong et al., 2015; Bentivogli et al., 2016; Junczys-Dowmunt et al., 2016; Wu et al., 2016; Toral and Sánchez-Cartagena, 2017). However, due to some deficiencies of NMT systems such as the limited vocabulary size, low adequacy for some translations, much research work has involved incorporating extra knowledge such as SMT features or linguistic features into NMT to improve translation performance (He et al., 2016; Sennrich and Haddow, 2016; Nadejde et al., 2017; Wang et al., 2017).

Pre-reordering, a preprocessing step in SMT, modifies the word order of a source-side sentence to be more similar to the word order in a target language, and has proven very helpful in improving translation quality for SMT systems (Xia and McCord, 2004; Collins et al., 2005; Neubig et al., 2012; Miceli-Barone and Attardi, 2013; Nakagawa, 2015).[1] NMT has a strong capability to learn word orders or word alignment from sequential lexical information using the soft alignment (attention) mechanism, and NMT systems introduce more changes in word order than pure phrase-based SMT (PB-SMT) systems. Furthermore, NMT's reorderings are closer to the reorderings in the reference than those of PB-SMT (Toral and Sánchez-Cartagena, 2017). Thus, in this paper, we ask the question whether pre-reordering is necessary and helpful for NMT.

The intuition behind pre-reordering for NMT is contradictory: on the one hand, if the word order of a source-side sentence is close to that of the target language, then the attention mechanism can easily learn a diagonal alignment, so pre-reordering might be helpful to the learning process; on the other hand, compared to the weak global reordering capability of PB-SMT, the attention mechanism in NMT can globally learn the word alignment, so pre-reordering might be redundant.

Zhu (2015) firstly reported the observation that performing pre-reordering on NMT hurts the model performance. In his experiment, the pre-reordered NMT system using long-short term memory (LSTM) degrades by 1.22 BLEU (Papineni et al., 2002) points compared to the baseline NMT system. However, he only empirically performed experiments on English→Japanese, and did not have a general verification on other language pairs and analyse the reason behind the result.

In this paper we investigate the impact and generality of pre-reordering on NMT, and verify whether pre-reordering is redundant for NMT by comprehensively experimenting on two language pairs, four translation directions in total, and then propose an indirect method of utilizing the pre-reordering features as factors in NMT to enhance the attention model to learn more accurate word alignments. The main contributions of this work include:

- We examine the effect of pre-reordered training data on NMT models on a number of translation directions, which shows that pre-reordering is not helpful to the current NMT architecture. The pre-reordering operation is like a hard constraint which deteriorates the learning capability of neural networks from the natural word order.

[1] A huge of amount of work has been done on this topic. Here we only list some example papers.

- We propose a new feature and incorporate it with SMT-based pre-reordering features as factors to NMT to verify their impact on translation quality.
- We provide a qualitative analysis on the translation results.

2. Related Work

To the best of our knowledge, there is limited work published on the issue of pre-reordering for NMT. Zhu (2015) is the first work to report that the NMT system trained on the pre-reordered data hurts translation quality compared to the NMT system trained on the naturally ordered data. In his experiments on English→Japanese task, the pre-reordered NMT system decreases by 1.22 BLEU points compared to the normal LSTM NMT system. However, he did not examine the reasons behind the result and verify on other language pairs.

Niehues et al. (2016) proposed a pre-translation strategy to combine SMT and NMT, in which the SMT system is used to pre-translate the input and then an NMT system generates the final hypothesis using the pre-translation. In this framework, they only use the pre-reordered data to train SMT systems rather than NMT systems. In their experiments, the pre-translation system using the pre-reordered SMT system can improve translation quality compared to that trained on naturally ordered data.

Toral and Sánchez-Cartagena (2017) carried out a multifaceted evaluation of NMT versus PB-SMT for 9 language directions. One evaluation is the reordering. However, their work is not to perform reordering in the source-side sentences to train the NMT systems, but to measure the amount of reordering performed by NMT and PB-SMT systems, i.e. whether NMT systems produce more changes in the word order of a sentence than the PB-SMT systems, and whether NMT systems make the word order of the translation closer to that of the reference.

A number of works on integrating extra knowledge or different features into NMT have been carried out recently. He et al. (2016) incorporate SMT features, such as a translation model and an n-gram language model, with the NMT model under the log-linear framework. Their experiments show that the proposed method significantly improves translation quality of the baseline NMT system on Chinese→English translation tasks.

Wang et al. (2017) propose to incorporate an SMT model into the NMT framework in which at each decoding step, SMT offers additional recommendations of generated words based on the decoding information from NMT, and then an auxiliary classifier is employed to score the SMT recommendations and a gating function is used to combine the SMT recommendations with NMT generations, both of which are jointly trained within the NMT architecture in an end-to-end manner. Experimental results on Chinese–English translation show that the proposed approach achieves significant and consistent improvements over state-of-the-art NMT and SMT systems.

Different from the above work, Sennrich and Haddow (2016) integrate linguistic features such as morphological features, POS tags, and syntactic dependency labels

as input features to NMT system by generalising the embedding layer of the encoder. In experiments on WMT16 training and test sets, linguistic input features improve model quality. García-Martínez et al. (2016) propose the concept of factored NMT, and they use the linguistic decomposition of the words in the output side rather than in the input.

Similar to the work in Sennrich and Haddow (2016), we propose to incorporate features such as SMT-based pre-reordering features and a new reordered index feature as inputs to NMT to verify their effectiveness in improving translation quality.

3. Neural Machine Translation

The basic principle of an NMT system is that it can map a source-side sentence $\mathbf{x} = (x_1, \ldots, x_m)$ to a target sentence $\mathbf{y} = (y_1, \ldots, y_n)$ in a continuous vector space, where all sentences are assumed to terminate with a special "end-of-sentence" token $< eos >$. Conceptually, an NMT system employs neural networks to solve the conditional distributions as in (1):

$$p(\mathbf{y}|\mathbf{x}) = \prod_{i=1}^{n} p(y_i|y_{<i}, x_{\leq m}) \tag{1}$$

We utilise the NMT architecture in Bahdanau et al. (2015), which is implemented as an attentional encoder-decoder network with recurrent neural networks (RNN).

In this framework, the encoder is a bidirectional neural network (Sutskever et al., 2014) with gated recurrent units (Cho et al., 2014) where a source-side sequence $\mathbf{x}$ is converted to a one-hot vector and fed in as the input, and then a forward sequence of hidden states $(\overrightarrow{h}_1, \ldots, \overrightarrow{h}_m)$ and a backward sequence of hidden states $(\overleftarrow{h}_1, \ldots, \overleftarrow{h}_m)$ are calculated and concatenated to form the annotation vector h_j. The decoder is also an RNN that predicts a target sequence $\mathbf{y}$ word by word where each word y_i is generated conditioned on the decoder hidden state s_i, the previous target word y_{i-1}, and the source-side context vector c_i as in (2):

$$p(y_i|y_{<i}, \mathbf{x}) = g(y_{i-1}, s_i, c_i) \tag{2}$$

where g is the activation function that outputs the probability of y_i, and c_i is calculated as a weighted sum of the annotations h_j. The weight α_{ij} is computed as in (3):

$$\alpha_{ij} = \frac{exp(e_{ij})}{\sum_{k=1}^{m} exp(e_{ik})} \tag{3}$$

where

$$e_{ij} = a(s_{i-1}, h_j)$$

is an alignment model which models the probability that the inputs around position j are aligned to the output at position i. The alignment model is a single-layer feed-forward neural network that is learned jointly through backpropagation.

4. Top-Down BTG-based Pre-reordering

In PB-SMT, the difference in word order between source and target languages is one of the major problems. Pre-reordering source-side word order closes to that of the target language is one of many approaches to deal with this issue. In this paper, we investigate a pre-reordering method based on Bracketing Transduction Grammar (BTG) (Neubig et al., 2012) for NMT systems.[2]

The BTG-based pre-reordering method reorders source sentences by handling sentence structures as latent variables. Nakagawa (2015) proposed an incremental top-down parsing method to improve the computational efficiency of the original BTG-based pre-reordering where model parameters can be learned using latent variable Perceptron with the early update technique. His experiments show that pre-ordering using the top-down parsing algorithm was faster and achieved higher BLEU scores than the original BTG-based pre-ordering method.

The advantage of the top-down BTG-based pre-reordering method is that it can be easily applied to any languages using only parallel text. Given a word x_i in a source-side sentence $\mathbf{x}$, three features are used to pre-reorder $\mathbf{x}$, namely the word surface form x_i^w, POS tag x_i^p and word class x_i^c. To train the pre-ordering model, the word alignment links between words in the source and target sentences of the parallel training data are also provided. The trained pre-reordering model is then employed to prereorder the training data and test data annotated by the above three features.

5. Factored NMT Using Pre-reordering Features

Factored NMT, introduced in Sennrich and Haddow (2016), represents the encoder input as a combination of features as in (4):

$$\overrightarrow{h}_j = g(\overrightarrow{W}(\Big\|_{k=1}^{|F|} E_k x_{jk}) + \overrightarrow{U}\overrightarrow{h}_{j-1}) \tag{4}$$

where $\|$ is the vector concatenation, $E_k \in \mathbb{R}^{m_k \times K_k}$ are the feature embedding matrices, with $\sum_{k=1}^{|F|} m_k = m$, and K_k is the vocabulary size of the k_{th} feature, and $|F|$ is the number of features in the feature set F (Sennrich and Haddow, 2016).

In factored NMT, the features can be any form of knowledge which might be useful to NMT systems, such as POS tags, lemmas, morphological features and dependency labels used in Sennrich and Haddow (2016). In our work, besides the the pre-reordering features, namely the POS tag and word class, we propose another feature

[2] In future work, we will examine the impact of different pre-reordering methods on NMT.

to verify how these features affect the performance of NMT systems. The new feature is defined as "Reordered Index" which is illustrated in Table 1.

Source:	Aozhou shi yu Beihan you bangjiao de shaoshu guojia zhiyi .
Original Index:	0 1 2 3 4 5 6 7 8 9 10
Reference:	Australia is one of the few countries that have diplomatic relations with North Korea .
Pre-reordered:	Aozhou shi zhiyi shaoshu guojia de you bangjiao yu Beihan .
Absolute Reordered Index:	0 1 9 7 8 6 4 5 2 3 10
Source:	Aozhou shi yu **Beihan** you bangjiao de shaoshu guojia zhiyi .
Relative Reordered Index:	0 0 6 **6** 2 2 -1 -4 -4 -7 0

Table 1. An example of reordered index as an input feature for NMT

In Table 1, the source language is Chinese (shown as Chinese Pinyin) and the reference is English. "Pre-reordered" indicates the reordered Chinese sentence by the BTG-based pre-reordering model. "Original Index" is the sequence of word position in the original source-side sentence, and "Absolute Reordered Index" is the reordered sequence of word positions where the number represents the word position in the original source-side sequence.

In order to reduce data sparseness, we convert the absolute word positions in "Absolute Reordered Index" to relative word positions in "Relative Reordered Index", which is calculated as in (5):

$$\mathrm{relative\_p} = \mathrm{p\_in\_reordered\_sequence} - \mathrm{p\_in\_original\_sequence} \quad (5)$$

For example, the word "**Beihan**" in Table 1 has the absolute position "3" in the original source sentence, while it moves to position "9" in the pre-reordered source sentence. Then we have $\{\mathrm{relative\_p} = 9 - 3 = \mathbf{6}\}$ as shown in the last row of Table 1.

6. Experiments

As Japanese and Chinese languages differ drastically from English in terms of word order and grammatical structure, we select Japanese–English and Chinese–English translations[3] to verify the impact of pre-reordering on NMT.

Two sets of experiments are set up as follows:

- Pre-reordering for NMT: four translation directions (JP↔EN and ZH↔EN) are evaluated on non-prereordered and pre-reordered data for NMT.

[3]In the rest of the paper, we use JP, ZH and EN to denote Japanese, Chinese and English, respectively.

- Factored NMT: SMT-based pre-reordering features are encoded as input factors for NMT systems.

In the following sections, we will report our experimental setup and results in terms of these two experiments.

6.1. Experimental Settings

For JP–EN translation tasks, the training data is the first part (train-1) of the JP–EN Scientific Paper Abstract Corpus (ASPEC-JE) that contains 1M sentence pairs, the development/validation set contains 1,790 sentence pairs, and the test set contains 1,812 sentence pairs (Nakazawa et al., 2016). There is only one reference for each source-side sentence in the validation and test sets.

For ZH–EN tasks, we use 1.4M sentence pairs extracted from LDC ZH–EN corpora as the training data, and NIST 2004 current set as the development/validation set that contains 1,597 sentences, and NIST 2005 current set as the test set that contains 1,082 sentences. There are four references for each Chinese sentence and there is only one reference for each English sentence in the validation and test sets. For EN→ZH, we use the first reference out of four references for Chinese as the input (English).

The pre-reordering factors, namely the POS tag, word class and reordered index are obtained by:

- POS tag: the Japanese data are segmented and tagged using KyTea (Neubig et al., 2011), and the Chinese data are segmented and tagged using the ICTCLAS toolkit (Zhang et al., 2003).
- Word Class (WoC): the word classes of the training data are obtained using "mkcls" by setting the number of classes to 50. For an Out-of-Vocabulary word in the validation and test sets, we randomly allocate a class between (1, 50) to it.
- Reordered Index (ReIdx): we generate two different kinds of reordered indices, namely the "Absolute Reordered Index" (AbsReIdx) and "Relative Reordered Index" (RelaReIdx) which are described in Section 5.

Chinese and Japanese are not suitable for using the Byte Pair Encoding (BPE) method (Sennrich et al., 2016) to encode words as subword units. Thus, we keep the words as translation units. We use Moses (Koehn et al., 2007) with default settings as the standard PB-SMT system, and use KenLM (Heafield et al., 2013) to train a 5-gram language model with the target side of the parallel data. We use Nematus (Sennrich et al., 2017) as the baseline NMT system, and set minibatches of size 80, a maximum sentence length of 60, word embeddings of size 600, and hidden layers of size 1024. The vocabulary size for input and output is set to 45K. Models are trained with the Adadelta optimizer (Zeiler, 2012), reshuffling the training corpus between epochs. We validate the model every 5,000 minibatches via BLEU scores on the validation set.

As in Sennrich and Haddow (2016), for factored NMT systems, in order to ensure that performance improvements are not simply due to an increase in the number of model parameters, we keep the total size of the embedding layer fixed to 600. Table 2

shows the vocabulary size and embedding size for pre-reordering features and the word as the input for the JP→EN NMT system. The total embedding size is fixed to 600. "Varied" indicates that for each single feature, the word embedding size will be different which is obtained by $[600 - embedding\_size(feature)]$. For example, the word embedding size will be $600 - 10 = 590$ for using POS tags as the input feature. Similar settings and parameters are for Chinese. We add 'UNK' to the vocabulary of each feature.

	Input Voc. Size		Input Voc. Size		Embedding Size	
Feature	JP	Model	ZH	Model	All	Single
POS tags	21	21	37	37	10	10
Word Class	51	51	51	51	15	15
AbsReIdx	61	61	61	61	15	15
RelaReIDX	117	117	117	117	20	20
Word	161,390	45,000	185,029	45,000	540	Varied

Table 2. Vocabulary size, and size of embedding layer of each feature.

In order to verify the impact of pre-reordered data on NMT systems and how pre-reordering features affects NMT systems, we only use the single NMT model rather than an ensemble model. The beam size for NMT decoding is 12. All results are reported by case-insensitive BLEU scores and carried out a bootstrap resampling significance test (Koehn, 2004).

6.2. Results and Analysis

Tables 3 and 4 show our main results for JP↔EN and ZH↔EN with and without pre-reordered data, respectively. The baseline system is a standard PB-SMT system trained on non-reordered and pre-reordered data, respectively.

	JP→EN				EN→JP			
	Non-reordered		Pre-reordered		Non-reordered		Pre-reordered	
SYS	Validation	Test	Validation	Test	Validation	Test	Validation	Test
SMT	18.25	17.64	**21.79***	**21.71***	27.03	26.32	**33.67***	**33.75***
NMT	**24.16***	**24.55***	20.42	21.43	**35.25***	**35.23***	32.75	32.98
Gain	+5.91	+ 6.91	-1.37	-0.31	+ 8.22	+ 8.91	-0.92	-0.77

Table 3. Results on JP–EN pre-reordering experiments. "" indicates translation performance is significantly better.*

	ZH→EN				EN→ZH			
	Non-reordered		Pre-reordered		Non-reordered		Pre-reordered	
SYS	Validation	Test	Validation	Test	Validation	Test	Validation	Test
SMT	33.13	29.24	**34.63***	**30.59***	14.50	12.77	**16.12***	**13.77***
NMT	**35.49***	**31.76***	33.95	30.23	**15.97***	**15.62***	14.14	13.53
Gain	+2.46	+2.52	-0.68	-0.36	+1.47	+2.85	-1.98	-0.22

Table 4. Results on ZH–EN pre-reordering experiments

NMT systems trained on the non-reordered data significantly improve on the validation set by 5.91 (18.25→24.16) and on the test set by 6.91 (17.64→24.55) absolute points for JP→EN, respectively; and by 8.22 (27.03→35.25) absolute points on the validation set and 8.91 (26.32→35.23) absolute points on the test set for EN→JP, respectively, compared to SMT systems.

Non-reordered NMT systems significantly improve on the validation set by 2.46 (33.13→35.49) and on the test set by 2.52 (29.24→31.76) absolute points for ZH→EN, respectively; and by 1.47 (14.50→15.97) on the validation set and 2.85 (12.77→15.62) absolute points on the test set for EN→ZH, respectively, compared to SMT systems.

However, for NMT systems trained on the pre-reordered data, translation performance decreases both on the validation set and test set compared to the SMT systems trained on the pre-reordered data. We also observe that 1) pre-reordered SMT systems achieve significant improvement compared to baseline SMT systems; 2) pre-reordered NMT systems perform worse than the non-reordered NMT systems.

From the results we can see that the pre-reordering has a negative impact on the learning capability of NMT systems. We infer that the pre-reordering is like a hard constraint for NMT and introduces more noise in terms of word order, which appears to make the learning process more difficult.

We also evaluate pre-reordering features as input factors for the NMT system against the baseline NMT system. The results are shown in Table 5.

	JP→EN		ZH→EN	
SYS	Validation	Test	Validation	Test
NMT	24.16	24.55	35.49	31.76
AbsReIdx	24.40	24.61	36.42*	31.90
RelaReIdx	24.52*	24.90*	36.87*	31.96*
POS+WoC	25.08*	25.17*	37.42*	33.15*
POS+WoC+RelaReIdx	**25.26***	**25.65***	**37.83***	**33.63***
Gain	1.1	1.1	2.34	1.87

Table 5. Results on JP→EN and ZH→EN factored NMT Experiments

We observe that the proposed "Reordered Index" features, namely the AbsReIdx and RelaReIdx can improve translation quality, but the former is not significant while the latter is significant, which shows that the relative reordering positions can provide more extra useful information to the words. The features of the pre-reordering model for SMT, namely the POS tags and word class, improve by 0.92 (24.16→25.08) and 1.93 (35.49→37.42) BLEU points on the validation set, respectively, and 0.62 (24.55→25.17) and 1.39 (31.76→33.15) BLEU points on the test set, respectively, compared to the baseline NMT system. In addition, adding the RelaReIdx further improves by 0.48 (25.17→25.65) and 0.48 (33.15→33.63) BLEU points on the test set, respectively. The incremental improvements in Table 5 show that the POS tags, word class and Reordered Index features contribute different information to the learning process of the NMT system to improve translation performance.

7. Conclusion

In this paper we investigate whether pre-reordering is beneficial to NMT and our empirical results show that it is not the case, i.e. pre-reordering the source-side data deteriorates translation performance. Linguistic knowledge has been verified to be useful in improving translation quality by resolving the reordering problem, so we propose to integrate SMT-based pre-reordering features, namely POS tags, word class and reordered index as input factors into the JP–EN and ZH–EN NMT systems. Our experiments show that these pre-reordering features yield improvements over the baseline NMT system, resulting in improvements on the test set of 1.1 and 1.87 BLEU points, respectively, on the test sets.

As to future work, we expect more experiments on different language pairs and different pre-reordering methods to verify the impact of pre-reordering on NMT, and we will explore the inclusion of novel and different reordering features for NMT to improve reordering in translations further.

Acknowledgements

We would like to thank the reviewers for their valuable and constructive comments. Thanks Dr. Jian Zhang for his initial idea and work on pre-reordered SMT. This research is supported under the Science Foundation Ireland Research Centres Programme (Grant 13/RC/2106).

Bibliography

Bahdanau, Dzmitry, Kyunghyun Cho, and Yoshua Bengio. Neural Machine Translation by Jointly Learning to Align and Translate. In *Proc. of the 3rd International Conference on Learning Representations*, pages 1–15, San Diego, USA, 2015.

Bentivogli, Luisa, Arianna Bisazza, Mauro Cettolo, and Marcello Federico. Neural versus phrase-based machine translation quality: a case study. In *Proc. of the EMNLP*, 2016.

Cho, Kyunghyun, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation. In *Proc. of the EMNLP*, 2014.

Collins, Michael, Philipp Koehn, and Ivona Kucerova. Clause Restructuring for Statistical Machine Translation. In *Proc. of the ACL*, pages 531–540, Ann Arbor, Michigan, USA, 2005.

García-Martínez, Mercedes, Loïc Barrault, and Fethi Bougares. Factored Neural Machine Translation. In *arXiv:1609.04621*, 2016.

He, Wei, Zhongjun He, Hua Wu, and Haifeng Wang. Improved Neural Machine Translation with SMT Features. In *Proc. of the Thirtieth AAAI Conference on Artificial Intelligence*, pages 151–157, Phoenix, Arizona, USA, 2016.

Heafield, Kenneth, Ivan Pouzyrevsky, Jonathan H. Clark, and Philipp Koehn. Scalable Modified Kneser-Ney Language Model Estimation. In *Proc. of the ACL*, pages 690–696, Sofia, Bulgaria, 2013.

Junczys-Dowmunt, Marcin, Tomasz Dwojak, and Hieu Hoang. Is Neural Machine Translation Ready for Deployment? A Case Study on 30 Translation Directions. In *Proc. of the IWSLT*, Tokyo, Japan, 2016.

Kalchbrenner, Nal and Phil Blunsom. Recurrent continuous translation models. In *Proc. of the EMNLP*, pages 1700–1709, Seattle, Washington, USA, 2013.

Koehn, Philipp. Statistical Significance Tests for Machine Translation Evaluation. In *Proc. of the EMNLP*, pages 388–395, Barcelona, Spain, 2004.

Koehn, Philipp, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondřej Bojar, Alexandra Constantin, and Evan Herbst. Moses: Open Source Toolkit for Statistical Machine Translation. In *Proc. of the ACL*, pages 177–180, Prague, Czech Republic, 2007.

Luong, Thang, Hieu Pham, and Christopher D. Manning. Effective approaches to attention-based neural machine translation. In *Proc. of the EMNLP*, pages 1412–1421, Lisbon, Portugal, 2015.

Miceli-Barone, Antonio Valerio and Giuseppe Attardi. Pre-Reordering for Machine Translation Using Transition-Based Walks on Dependency Parse Trees. In *Proc. of the Eighth Workshop on Statistical Machine Translation*, pages 164–169, Sofia, Bulgaria, 2013.

Nadejde, Maria, Siva Reddy, Rico Sennrich, Tomasz Dwojak, Marcin Junczys-Dowmunt, Philipp Koehn, and Alexandra Birch. Syntax-aware Neural Machine Translation Using CCG. In *arXiv:1702.01147*, 2017.

Nakagawa, Tetsuji. Efficient Top-Down BTG Parsing for Machine Translation Preordering. In *Proc. of the ACL-IJCNLP*, pages 208–218, Beijing, China, 2015.

Nakazawa, Toshiaki, Manabu Yaguchi, Kiyotaka Uchimoto, Masao Utiyama, Eiichiro Sumita, Sadao Kurohashi, and Hitoshi Isahara. ASPEC: Asian Scientific Paper Excerpt Corpus. In *Proc. of the Tenth LREC*, Portorož, Slovenia, 2016.

Neubig, Graham, Yosuke Nakata, and Shinsuke Mori. Pointwise Prediction for Robust, Adaptable Japanese Morphological Analysis. In *Proc. of the ACL-HLT*, pages 529–533, Portland, Oregon, USA, 2011.

Neubig, Graham, Taro Watanabe, and Shinsuke Mori. Inducing a Discriminative Parser to Optimize Machine Translation Reordering. In *Proc. of the EMNLP-CoNLL*, pages 843–853, Jeju Island, Korea, 2012.

Niehues, Jan, Eunah Cho, Thanh-Le Ha, and Alex Waibel. Pre-Translation for Neural Machine Translation. In *Proc. of the COLING*, pages 1828–1836, Osaka, Japan, 2016.

Papineni, Kishore, Salim Roukos, Todd Ward, and WeiJing Zhu. BLEU: A Method for Automatic Evaluation of Machine Translation. In *Proc. of the ACL*, pages 311–318, 2002.

Sennrich, Rico and Barry Haddow. Linguistic Input Features Improve Neural Machine Translation. In *Proc. of the 1st Conference on Machine Translation*, pages 83–91, Berlin, 2016.

Sennrich, Rico, Barry Haddow, and Alexandra Birch. Neural Machine Translation of Rare Words with Subword Units. In *Proc. of the ACL*, pages 1715–1725, Berlin, Germany, 2016.

Sennrich, Rico, Orhan Firat, Kyunghyun Cho, Alexandra Birch, Barry Haddow, Julian Hitschler, Marcin Junczys-Dowmunt, Samuel Läubli, Antonio Valerio Miceli Barone, Jozef Mokry, and Maria Nădejde. Nematus: a Toolkit for Neural Machine Translation. In *arXiv:1703.04357*, 2017.

Sutskever, Ilya, Oriol Vinyals, , and Quoc V Le. Sequence to sequence learning with neural networks. In *Proc. of the 2014 Neural Information Processing Systems*, pages 3104–3112, Montreal, Canada, 2014.

Toral, Antonio and Víctor M. Sánchez-Cartagena. A Multifaceted Evaluation of Neural versus Phrase-Based Machine Translation for 9 Language Directions. In *Proc. of the EACL*, Valencia, Spain, 2017.

Wang, Xing, Zhengdong Lu, Zhaopeng Tu, Hang Li, Deyi Xiong, and Min Zhang. Neural Machine Translation Advised by Statistical Machine Translation. In *Proc. of the AAAI Conference on Artificial Intelligence*, San Francisco, California, USA, 2017.

Wu, Yonghui, Mike Schuster, Zhifeng Chen, Quoc V. Le, and Mohammad Norouzi et al. Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation. In *arXiv:1609.08144*, 2016.

Xia, Fei and Michael McCord. Improving a Statistical MT System with Automatically Learned Rewrite Patterns. In *Proc. of the COLING*, pages 508–514, IIT Bombay, India, 2004.

Zeiler, Matthew D. ADADELTA: An Adaptive Learning Rate Method. In *CoRR, abs/1212.5701*, 2012.

Zhang, Huaping, Hongkui Yu, Deyi Xiong, and Qun Liu. HHMM-based Chinese Lexical Analyzer ICTCLAS. In *Proc. of the Second SIGHAN Workshop on Chinese Language Processing*, pages 184–187, Sapporo, Japan, 2003.

Zhu, Zhongyuan. Evaluating Neural Machine Translation in English-Japanese Task. In *Proc. of the 2nd Workshop on Asian Translation*, pages 61–68, Kyoto, Japan, 2015.

Address for correspondence:
Andy Way
andy.way@adaptcentre.ie
ADAPT Centre, School of Computing, Dublin City University,
Glasnevin, Dublin 9, Dublin, Ireland

PBML

The Prague Bulletin of Mathematical Linguistics
NUMBER 108 JUNE 2017 183–195

Towards Optimizing MT for Post-Editing Effort: Can BLEU Still Be Useful?

Mikel L. Forcada,[a] Felipe Sánchez-Martínez,[a] Miquel Esplà-Gomis,[a] Lucia Specia[b]

[a] Departament de Llenguatges i Sistemes Informàtics
Universitat d'Alacant, E-03690 Sant Vicent del Raspeig, Spain
[b] Department of Computer Science
University of Sheffield, Regent Court, 211 Portobello, Sheffield, UK

Abstract

We propose a simple, linear-combination automatic evaluation measure (AEM) to approximate post-editing (PE) effort. Effort is measured both as PE time and as the number of PE operations performed. The ultimate goal is to define an AEM that can be used to optimize machine translation (MT) systems to minimize PE effort, but without having to perform unfeasible repeated PE during optimization. As PE effort is expected to be an *extensive* magnitude (i.e., one growing linearly with the sentence length and which may be simply added to represent the effort for a set of sentences), we use a linear combination of extensive and *pseudo-extensive* features. One such pseudo-extensive feature, 1–BLEU times the length of the reference, proves to be almost as good a predictor of PE effort as the best combination of extensive features. Surprisingly, effort predictors computed using independently obtained reference translations perform reasonably close to those using actual post-edited references. In the early stage of this research and given the inherent complexity of carrying out experiments with professional post-editors, we decided to carry out an automatic evaluation of the AEMs proposed rather than a manual evaluation to measure the effort needed to post-edit the output of an MT system tuned on these AEMs. The results obtained seem to support current tuning practice using BLEU, yet pointing at some limitations. Apart from this intrinsic evaluation, an extrinsic evaluation was also carried out in which the AEMs proposed were used to build synthetic training corpora for MT quality estimation, with results comparable to those obtained when training with measured PE efforts.

 Corresponding author: mlf@dlsi.ua.es
Cite as: Mikel L. Forcada, Felipe Sánchez-Martínez, Miquel Esplà-Gomis, Lucia Specia. Towards Optimizing MT for Post-Editing Effort: Can BLEU Still Be Useful?. The Prague Bulletin of Mathematical Linguistics No. 108, 2017, pp. 183–195. doi: 10.1515/pralin-2017-0019.

1. Introduction

Machine translation (MT) applications fall in two main groups: *assimilation* or *gisting*, and *dissemination*. Assimilation takes place when the raw MT output is used to make sense of documents written in a foreign language. Dissemination refers to the use of the MT output as a draft translation that is *post-edited* (corrected) by a professional to generate a publishable translation (Krings and Koby, 2001; O'Brien and Simard, 2014). The requirements of both groups of applications are quite different;[1] however state-of-the-art MT systems are usually optimized to produce translations that resemble existing references in a training or development set, regardless of their application. In statistical MT, this is done by using *automatic evaluation measures* (AEM) such as BLEU (Papineni et al., 2002), the most popular one. In neural MT —usually trained to maximize logarithmic likelihood— AEMs may still be used as a stopping criterion, or even as part of a loss function (Shen et al., 2016).

For dissemination, rather than optimizing the MT system to imitate existing, independently created, reference translations,[2] it would make more sense to optimize it to reduce post-editing (PE) *effort*. PE effort is an *extensive* magnitude, that is, one that grows linearly[3] with the sentence length and which may be simply added to represent the effort for a set of sentences. One straightforward measure of PE effort is PE *time*, since it is directly related to productivity. Additionally, the time devoted to PE is a key metric to budget a translation task.

In addition to PE time, one of the most used metrics for PE effort is human-targeted translation edit rate (HTER) (Snover et al., 2006, 2009; Specia and Farzindar, 2010). This metric computes the translation edit rate (TER) between the raw translation $\mathrm{MT}(s_i)$ produced by an MT system and a given (*human*, hence the *H*) PE of this translation $t_i^{(p)}$, that is, the minimum number of insertions, deletions and substitutions of one word or shifts of blocks of one or more words, divided by the length of the post-edited translation.

One of the main advantages of this metric over time is that it can be computed on any already post-edited translations. However, to use it as an extensive indicator of effort, rather than normalizing it by the length of the reference translation, we need to use the actual number of translation edits (NTE) instead of translation edit rates. The main disadvantage of NTE over PE time is that it disregards the cognitive effort of PE, that is, it does not take into account the time invested by post-editors reading

[1] For instance, a Russian–English translation with no articles (*some*, *a*, *the*), may be just about right for assimilation, but would need significant post-editing for dissemination.

[2] Reference translations that have been produced based on the source text only, and not by post-editing the output of the MT system being evaluated.

[3] The linear growth assumption should be evaluated empirically. For instance the performance of state-of-the-art systems (neural MT systems) seem to degrade with length (Toral and Sánchez-Cartagena, 2017) and could lead to non-linear PE times. However the linear approach seems to be a good starting point, given that in commercial scenarios, the cost of translation is measured based on the length of the text.

the translation and identifying the parts that need to be fixed, the time invested in checking external resources, such as dictionaries or bilingual concordancers, and the time spent revising the final translation. In contrast, PE time can only be measured in a controlled environment, which makes it less practical.

In dissemination applications of MT, it would therefore make sense to use PE effort metrics for model optimization. However, repeatedly collecting PE time or NTE during system optimization is unfeasible. Hundreds of thousands of candidate translations would have to be edited by professionals, a prohibitively expensive and time-consuming process. Datasets with reference translations are, on the other hand, abundant. Therefore, ideally one could optimize MT by using an AEM that, given the MT output and an independent reference translation, predicts the required PE effort.

A number of publicly available corpora provide PE times or raw and post-edited machine translations (see Section 3); however, to the best of our knowledge, while there has been extensive work in predicting PE time or PE rates as a MT quality estimation (QE) task (Specia and Soricut, 2013) (that is, without a reference translation) as part of shared tasks (Bojar et al., 2013, 2014, 2016), no AEM that could be used to optimize MT systems with respect to PE effort has yet been proposed. The only exception is the work of Denkowski (2015), who shows that when an AEM "tuned to post-editing effort is used as an objective function for system optimization, the resulting translations require less effort to edit than those from a BLEU-optimized system".

Denkowski (2015) used unpublished PE data and METEOR, a rather complex AEM relying on resources such as stemmers and paraphrase tables. This paper sets out to define very simple AEMs based on a linear combination of MT system-independent features which aim at predicting PE effort (either time or NTE) as an *extensive* magnitude. It also studies whether sentence-level BLEU computed on independent reference translations could actually be repurposed as a reasonable predictor of PE effort. This work is part of an ongoing research aimed at defining AEMs to be used to optimize MT systems to minimize PE effort.[4]

2. Predicting post-editing effort as an extensive quantity

Since PE effort is expected to be an extensive quantity, we propose using a linear combination of extensive and *pseudo-extensive* features. We will consider time and the number of edits as specific cases of effort (Forcada and Sánchez-Martínez, 2015). The effort of post editing the MT output for segment i in a translation job may be denoted by $T(s_i, \mathrm{MT}(s_i))$, which will be approximated by a tunable AEM of the form

$$\hat{T}(s_i, \mathrm{MT}(s_i), t_i; \vec{\mu}) = \sum_{j=1}^{n_F} \mu_j f_j(s_i, \mathrm{MT}(s_i), t_i), \quad (1)$$

[4]One could imagine this as a linear per-word cost model with a discount proportional to various indicators of closeness to the reference.

where a single reference t_i is assumed, $f_j(s_i, MT(s_i), t_i)$ are the extensive and pseudo-extensive features, and $\vec{\mu}$ is the set of tunable parameters of the AEM. The coefficients μ_j may be obtained by linear regression on a training set.

2.1. Extensive features

The following list of simple extensive features has been preliminarily studied:

- Word-level length of raw MT output $MT(s_i)$ and reference segments t_i and their corresponding character-level counterparts.
- Word- and character-level Levenshtein-edit distances between $MT(s_i)$ and t_i.
- Word- and character-level components of the TER-style distance (Snover et al., 2006) between $MT(s_i)$ and t_i: number of insertions, deletions, substitutions, and block shifts for words and characters.
- $MT(s_i)$ word n-gram mismatches, i.e. number of sub-segments of length n in $MT(s_i)$ that do not appear in t_i, and vice versa, i.e. number of sub-segments of length n in t_i not appearing in $MT(s_i)$.

2.2. Pseudo-extensive features

Pseudo-extensive features may be easily derived from non-extensive AEM by combining them with the length of the reference segment, $len_W(t_i)$. In this paper we have studied the use of $sBLEU_n$, a sentence-level implementation of the well-known AEM $BLEU_n$ where n is the maximum n-gram size used; usually 4. The $sBLEU_n$ indicator takes values in $[0, 1]$ and is expected to be a *reverse* predictor of PE effort —the larger the $sBLEU_n$, the smaller the effort. Consequently, we use a *reversed* version of it so that the feature value is computed as

$$len_W(t_i) \times (1 - sBLEU_4(MT(s_i), t_i)) \quad (2)$$

where $sBLEU_4(\cdot, \cdot)$ is 4-gram sentence-smoothed implementation of BLEU ("Smoothing 3" by Chen and Cherry (2014), implemented in package MultEval as *JBLEU*).[5]

3. Experimental settings

3.1. Data sets

Several experiments were carried out with data sets based on those published for the shared task on MT quality estimation (QE) at the 2013, 2014 and 2016 editions of the Workshop on Statistical Machine Translation (WMT). Each data set consists

[5] As BLEU is unlikely to decrease linearly with effort, we tried a family of suitably transformed versions of Eq. (2), $len_W(t_i) \times (1 - (sBLEU_4(MT(s_i), t_i))^q)^p$, with $p, q > 0$. We found no significant improvement over $p = 1, q = 1$ by doing this in the range $[\frac{1}{3}, 3]$. Eq. (2) has intuitive interpretation: effort (cost) grows linearly with length, but effort is saved (discount) as BLEU gets higher.

	Translation direction	Num. of instances Training	Test
WMT'13	en→es	803	284
WMT'14	en→es	650	208
WMT'16	en→de	13,000	2,000

Table 1. Statistics about the corpora used in the experiments: translation direction, and number of training and test instances.

of: (a) a set of source language segments $\{s_i\}$; (b) the corresponding raw translation produced by an unknown MT system, which may not be the same system in some data sets; (c) an independent reference translation t_i for every source segment s_i, unrelated to the MT system being studied; and (d) the post-edited version $t_i^{(p)}$ of the MT output, together with the corresponding PE time in seconds, $T(s_i, MT(s_i))$. Corpus statistics are provided in Table 1.

Two of the data sets are for translation from English into Spanish (en→es) and were obtained from the data sets distributed as part of WMT'13 (Bojar et al., 2013)[6] and WMT'14 (Bojar et al., 2014),[7] respectively. Independent references were collected from the parallel data distributed for the shared MT task at the 2012 edition of WMT.[8] PE references were provided by the shared-task organizers.[9] The third data set is for English–German (en→de) translation and corresponds to WMT'16 MT QE shared task (Bojar et al., 2016).[10]

In all the experiments, the training–test division is the same performed for the corresponding WMT shared tasks. WMT'16 also provides development data, which was added to the training corpus.

3.2. Training and evaluation

The limited-memory, bound-constrained Broyden–Fletcher–Goldfarb–Shanno (L-BFGS-B) optimization algorithm (Byrd et al., 1995) implemented in the SciPy package (Walt et al., 2011) was used to learn the parameters $\vec{\mu}$ in Eq. (1) by directly mini-

[6] http://www.quest.dcs.shef.ac.uk/wmt13_files/

[7] http://www.quest.dcs.shef.ac.uk/wmt14_files/

[8] Independent references can be downloaded here: https://v.gd/indepref

[9] Post-edited references can be downloaded here: https://v.gd/peref

[10] Training and development data sets are available at: https://lindat.mff.cuni.cz/repository/xmlui/handle/11372/LRT-1646. Test data is available at http://www.quest.dcs.shef.ac.uk/wmt16_files_qe/task1_en-de_test.tar.gz

mizing the mean absolute error (MAE) over the training set

$$\text{MAE} = \frac{1}{n_{\text{train}}} \sum_{i=0}^{n_{\text{train}}} \left| \hat{T}(s_i, \text{MT}(s_i), t_i; \vec{\mu}) - T(s_i, \text{MT}(s_i)) \right|,$$

where n_{train} is the number of training examples. The models trained were then evaluated by computing the Pearson's correlation r between the predicted effort and the actual PE effort, as well as the MAE, this time over the n_{test} examples in the test set. The correlation and the MAE were computed in two situations, namely, using independent references and using the actual post-edited translations, and for: (a) the best combination of extensive features; (b) the pseudo-extensive version of sBLEU proposed in Section 2.2; (c) a combination of (a) and (b); and (d) an example-based baseline (see below).

3.3. A simple baseline

A simple example-based baseline computes, for a test instance $(\text{MT}(s_i), t_i)$, the character-level edit distance $d = \text{ED}_C(\text{MT}(s_i), t_i)$ and then estimates $\hat{T}(s_i, \text{MT}(s_i), t_i)$ as the average PE time of all training-set segments showing a distance d' which is the closest possible to d:

$$\tilde{T}(d) = \frac{1}{|\{s_j : \text{ED}_C(\text{MT}(s_j), t_j) = d'\}|} \sum_{s_j : \text{ED}_C(\text{MT}(s_j), t_j) = d'} T(s_j, \text{MT}(s_j)).$$

4. Results

4.1. Predicting time

Table 2 reports the PE time prediction results obtained with three groups of AEMs: the best-performing combinations of extensive features[11] in Section 2.1, the AEM using the pseudo-extensive feature based on $\text{sBLEU}_4(\text{MT}(s_i), t_i)$ (which will be called the *pseudo-extensive AEM* from now on), an AEM combining both, and the baseline defined in Section 3.3.

As can be seen, both the extensive and pseudo-extensive AEMs significantly outperform our example-based baseline. Note that the pseudo-extensive AEM shows an excellent performance, comparable to the AEM using the best combination of extensive features. This suggests that a simple AEM, $\mu|t_i|(1 - \text{sBLEU}_4(\text{MT}(s_i), t_i))$, with just one coefficient μ, would already be a reasonable estimator of time. The best independently performing extensive features are the number of mismatched n-grams between $\text{MT}(s_i)$ and t_i,[12] where n-gram matching is at the basis of BLEU.

[11] Other combinations were tried but could not be included given the space constraints.

[12] The best results correspond to $n = 2$ and $n = 3$.

		AEM predicting post-editing time							
		Best ext.		Pseudo-ext.		Combined		Baseline	
Corpus	**references**	r	MAE	r	MAE	r	MAE	r	MAE
WMT'13	independent	0.61	49.0 s	0.62	49.1 s	0.62	49.1 s	0.36	64.7 s
	postedited	0.67	45.2 s	0.68	46.0 s	0.68	44.8 s	0.33	72.5 s
WMT'14	independent	0.70	15.9 s	0.69	16.2 s	0.70	15.9 s	0.51	22.4 s
	postedited	0.85	11.8 s	0.81	13.7 s	0.85	11.6 s	0.63	18.3 s
WMT'16	independent	0.46	25.4 s	0.44	26.7 s	0.46	25.4 s	0.24	34.7 s
	postedited	0.55	21.7 s	0.50	24.7 s	0.55	21.7 s	0.36	30.3 s

Table 2. Pearson's correlation r and mean absolute error (MAE) in seconds for four time-predicting AEMs (best extensive, pseudo-extensive (modified BLEU), combination, and example-based baseline) and three different corpora, computed on independent and postedited references.

As expected, all the results included in Table 2 are substantially better for PE references than for independent ones. However, it is worth noting that they are not too distant. These results are encouraging, since they suggest that even when no PE references are available, for instance when optimizing statistical MT systems, the proposed AEMs can be useful.

How good are these results? As mentioned in Section 3, the data sets used in these experiments had previously been used for MT QE. For data set WMT'13, the Pearson correlation r and the MAE are available for the original task (Bojar et al., 2013, Table 18). The results obtained with our (rather simple) linear AEM (having access to a single reference) are around $r = 0.62$ and MAE $= 49$ s while those reported for MT QE (without access to a reference translation) range between $r = 0.42$ and $r = 0.68$ and between MAE $= 48$ s and MAE $= 71$ s. For data set WMT'14, only the MAE is available (Bojar et al., 2014, Table 16); our MAE are around 16 s while the results reported for MT QE range between 16.7 s and 21.5 s. As a contrast, ignoring the quality of MT(s), and using just the length of MT(s) as a single feature, without accessing the reference t_i, the results are slightly worse than our best predictors, but far better than the example-based baseline: $r = 0.57$ and MAE $= 52.0$ s for WMT'13, and MAE $= 18.7$ s for WMT'14. These results would suggest that more elaborate AEMs should be explored to give a better estimate of time; improvements are expected to happen through the introduction of both additional extensive features and additional pseudo-extensive versions of features based on non-extensive indicators.

4.2. Predicting the number of edits

Table 3 is analogous to the experiments in Table 2, but here the reference AEM is the number of translation edits (NTE) instead of PE time. Table 3 contains an addi-

	AEM predicting the number of translation edits									
	Best ext.		Pseudo-ext.		Combined		Baseline		Indep. NTE	
Corpus	r	MAE	r	MAE	r	MAE	r	MAE	r	MAE
WMT'13	0.82	3.4	0.81	3.5	0.82	3.4	0.65	4.7	0.81	3.9
WMT'14	0.69	2.8	0.69	2.9	0.70	2.8	0.41	4.3	0.70	5.0
WMT'16	0.75	2.3	0.58	2.5	0.75	2.3	0.42	3.1	0.73	3.7

Table 3. Pearson's correlation r and mean absolute error (MAE) in number of edit operations for four NTE-predicting AEMs (best extensive, pseudo-extensive (modified BLEU), combination, example-based baseline, and using simply the independent-reference NTE as a predictor) and three different corpora, computed on independent references.

tional column that contains the results obtained by using the NTE needed to convert the MT output into an independent reference to predict the actual NTE performed to convert the MT output into its post-edited version (HNTE or *human* NTE). This is used as a second baseline that allows to measure the difficulty of the task of predicting the actual number of edits done when post-editing. As can be seen, the independent value of NTE strongly correlates with the HNTE. It clearly outperforms the example-based baseline used in the previous experiment. However, as regards MAE, the best-extensive MAE and the pseudo-extensive MAE obtain clearly better results, especially for the case of the WMT'14 data. As in the previous experiments, the impact of combining extensive and pseudo-extensive features is almost negligible.

In general, one can see that the approaches in Table 3 correlate much better with HNTE than those in Table 2 with PE time. To explain this, note that HNTE cannot take cognitive (thinking, documentation) effort into account, while PE time naturally includes it. Since none of the features used in this work is capable of directly representing cognitive effort, it would seem logical that using them leads to AEMs showing a better correlation with HNTE than with PE time. It is worth mentioning that when predicting HNTE, the results are also more stable across corpora, ranging between 2 and 5 edit operations.

4.3. Extrinsic evaluation in a quality estimation task

AEMs were also evaluated extrinsically by using them to build synthetic corpora for training MT QE systems that predict time. Synthetic corpora were built as follows: 25% of the training data in each data set in Table 1 was used to train simple pseudo-extensive time-predicting AEMs of the form given in Eq. (2), in view of their performance. Each AEM was then used to predict from independent references the PE time of the remaining 75% of the corresponding corpus, which was then used as the training corpus for a linear regressor built on the baseline features used for Task 1.3 at WMT'13 (Bojar et al., 2013) and WMT'14 (Bojar et al., 2014). The MAE and

Corpus	Training set	PE time		NTE	
		r	MAE	r	MAE
WMT'13	original	0.61	52.8 s	0.75	4.0
	synthetic	0.60	52.1 s	0.71	4.1
WMT'14	original	0.61	18.8 s	0.61	3.3
	synthetic	0.58	17.9 s	0.51	3.4
WMT'16	original	0.40	29.4 s	0.66	2.6
	synthetic	0.39	28.1 s	0.54	2.8

Table 4. Pearson's correlation (r) and mean absolute error (MAE) in seconds for MT QE (PE time estimation) when using both the original training set and a synthetic training set obtained using pseudo-extensive time-predicting AEMs (modified BLEU) with three different corpora.

Pearson's correlation between the estimated PE time using both the original and the synthetic corpora to train the regressor were then compared.

The results of this evaluation, carried out with pseudo-extensive time-predicting and NTE-predicting AEMs for MT QE are shown in Table 4. As can be observed, the MAE obtained with the synthetic corpora are comparable to those obtained with the original training corpora, even though the synthetic corpora used were automatically annotated and are 25% smaller than the original corpora. In the case of the PE time, the Pearson's correlation is comparable, while it is significantly lower in the case of NTE. These results show the usefulness of extensive and pseudo-extensive AEMs for predicting PE time in applications other than optimizing MT.

4.4. Tuning MT with the new AEMs: a sanity check

The most objective way to test the usefulness of the new AEMs would be to tune two different SMT systems with the same development set, one following general practice (i.e., using document-level BLEU) and the other one using the sum of one of the new sentence-level AEMs over the whole development set, and then having professional translators edit the output of both. This would allow us to search for possible savings in PE time and number of edits, much in the same way as reported by Denkowski (2015). While straightforward, this is an expensive experiment that should only be carried out when one has good indications that it will lead to a conclusive result. In addition, the unpredictability of the behaviour of different professional translators may make it more difficult to extract conclusions from such experiment, especially in this very early stage of the research. Therefore, in order to obtain preliminary and more reliable initial results, we will resort to a quick "casting out nines" sanity check, as follows.

We repeatedly and randomly extract simulated development sets of $n_{dev} = 100$ sentences each from the test sets described in Section 3.1 without replacement.

The repeat rate is 0.4 times the size of the test set, to get stable statistics. Over each one of these sets $\{(MT(s_j), t_j)\}_{j=1}^{n_{dev}}$, we will compute three *budgeting* features:

- The total length of references $L = \sum_{k=1}^{n_{dev}} |t_i|$, which will be used as a baseline predictor of total effort for that development set which does not take quality into account.
- A measure based on document-level BLEU over the whole development set, $D = (1 - BLEU((MT(s_j), t_j)\}_{j=1}^{n_{dev}}) \times L$, which takes quality into account by establishing a document-level discount based on BLEU. Minimizing D is equivalent to maximizing $BLEU((MT(s_j), t_j)\}_{j=1}^{n_{dev}})$, which is common practice.
- A measure, $S = \sum_{k=1}^{n_{dev}} \hat{T}(s_k, MT(s_k), t_k; \mu)$, based on the sentence-level AEMs proposed in this paper.

We then study the correlation among them and with actual total effort for that development set $E = \sum_{k=1}^{n_{dev}} T(s_k MT(s_k))$. If the AEM designed is indeed an improvement, the correlation of S with E should be better than that of D (current practice) and much better than that of L (dummy baseline). The results are shown in Table 5. The main findings are as follows:

- The correlation of the pseudo extensive (S-pseudo) and best extensive (S-ext.) AEMs between them and with current BLEU optimization practice (D) is excellent (0.91 or higher). This would mean that current BLEU optimization practice should roughly lead to equivalent results compared to using the new AEMs proposed here.
- The correlation of S-ext., S-pseudo, and current practice (D) with E (time) is reasonable for WMT'13 and WMT'14 while it is only moderate for WMT'16. Correlation with E (edits) is reasonably good for the three corpora.
- The correlation of total length L (which does not take quality into account) with E (time and edits) is surprisingly high and not too far from that obtained with actual AEMs. This could point at limitations of BLEU,[13] as well as of the simple AEMs proposed in this paper, but could also be due to the fact that the underlying MT systems had already been optimized using BLEU and that under those conditions, length is a good enough prediction of PE effort.

5. Concluding remarks

This paper introduces new automatic evaluation measures (AEM) for MT aimed at approximating post-editing (PE) effort. Such metrics would allow optimizing MT systems with respect to PE effort, therefore potentially reducing the cost of translation for dissemination purposes.

We have analyzed the performance of simple AEMs based on extensive and pseudo-extensive features for predicting PE time and the number of translation edits performed during PE (HNTE). The results allow us to conclude that: (a) the AEMs pro-

[13] For instance, Denkowski and Lavie (2012) showed that BLEU did not significantly change after post-editing.

Correlation	Dataset	E (time)	E (edits)	L	D	S-pseudo	S-ext.
E (time)	WMT'13	1.000	0.611	0.588	0.648	0.646	0.609
	WMT'14		0.837	0.649	0.740	0.739	0.728
	WMT'16		0.411	0.356	0.433	0.434	0.455
E (edits)	WMT'13	0.611	1.000	0.688	0.783	0.790	0.810
	WMT'14	0.837		0.605	0.690	0.690	0.645
	WMT'16	0.411		0.417	0.598	0.576	0.770
L	WMT'13	0.588	0.688	1.000	0.878	0.876	0.924
	WMT'14	0.649	0.605		0.906	0.914	0.942
	WMT'16	0.356	0.417		0.684	0.726	0.760
D	WMT'13	0.648	0.783	0.878	1.000	0.999	0.941
	WMT'14	0.740	0.690	0.906		0.999	0.953
	WMT'16	0.433	0.598	0.684		0.990	0.922
S-pseudo	WMT'13	0.646	0.790	0.876	0.999	1.000	0.965
	WMT'14	0.739	0.690	0.914	0.999		0.937
	WMT'16	0.434	0.576	0.726	0.990		0.914
S-ext.	WMT'13	0.609	0.810	0.924	0.941	0.965	1.000
	WMT'14	0.728	0.645	0.942	0.953	0.937	
	WMT'16	0.455	0.770	0.760	0.922	0.914	

Table 5. Pearson correlation observed between randomly-sampled development tests among PE effort E *(time and edits), total length* L*, total length times one minus BLEU (*D*), and effort predictions* S *using pseudo-extensive and extensive AEMs.*

posed perform similarly both on post-edited and independent references, which makes them easier to use to optimize MT systems; (b) the proposed AEMs would not seem to be able to improve current optimization practice (based on BLEU); (c) BLEU is still quite far from actually being able to reliably predict effort (supporting the findings by Denkowski (2015)); and (d) time prediction is however good enough to become useful in other related tasks, such as creating training corpora for MT QE.

Future work will evaluate more elaborate AEMs for predicting PE effort based on the features described and other MT-system-independent features, and will also analyse the impact of using such predictions when actually optimizing MT systems with respect to PE effort in real translation tasks (as was done by (Denkowski, 2015), but using features with simpler interpretation than the METEOR metric).

Acknowledgements: Work supported by the Spanish government through project EFFORTUNE (TIN2015-69632-R) and through grant PRX16/00043 for Mikel L. Forcada, and by the European Commission through QT21 project (H2020 No. 645452).

Bibliography

Bojar, Ondřej, Christian Buck, Chris Callison-Burch, Christian Federmann, Barry Haddow, Philipp Koehn, Christof Monz, Matt Post, Radu Soricut, and Lucia Specia. Findings of the 2013 Workshop on Statistical Machine Translation. In *Proceedings of the Eighth Workshop on Statistical Machine Translation*, pages 1–44, Sofia, Bulgaria, August 2013.

Bojar, Ondrej, Christian Buck, Christian Federmann, Barry Haddow, Philipp Koehn, Johannes Leveling, Christof Monz, Pavel Pecina, Matt Post, Herve Saint-Amand, et al. Findings of the 2014 Workshop on Statistical Machine Translation. In *Proceedings of the Ninth Workshop on Statistical Machine Translation*, pages 12–58, Baltimore, MD, USA, 2014.

Bojar, Ondřej, Rajen Chatterjee, Christian Federmann, Yvette Graham, Barry Haddow, Matthias Huck, Antonio Jimeno Yepes, Philipp Koehn, Varvara Logacheva, Christof Monz, Matteo Negri, Aurelie Neveol, Mariana Neves, Martin Popel, Matt Post, Raphael Rubino, Carolina Scarton, Lucia Specia, Marco Turchi, Karin Verspoor, and Marcos Zampieri. Findings of the 2016 Conference on Machine Translation. In *Proceedings of the First Conference on Machine Translation*, pages 131–198, Berlin, Germany, August 2016. URL `http://www.aclweb.org/anthology/W/W16/W16-2301`.

Byrd, Richard H, Peihuang Lu, Jorge Nocedal, and Ciyou Zhu. A limited memory algorithm for bound constrained optimization. *SIAM Journal on Scientific Computing*, 16(5):1190–1208, 1995.

Chen, Boxing and Colin Cherry. A Systematic Comparison of Smoothing Techniques for Sentence-Level BLEU. In *Proceedings of the Ninth Workshop on Statistical Machine Translation*, pages 362–367, Baltimore, Maryland, USA, June 2014. URL `http://www.aclweb.org/anthology/W/W14/W14-3346`.

Denkowski, Michael. *Machine Translation for Human Translators*. PhD thesis, Carnegie Mellon University, May 2015.

Denkowski, Michael and Alon Lavie. Challenges in Predicting Machine Translation Utility for Human Post-Editors. In *Proceedings of AMTA 2012*, 2012.

Forcada, Mikel L. and Felipe Sánchez-Martínez. A general framework for minimizing translation effort: towards a principled combination of translation technologies in computer-aided translation. In *Proceedings of the 18th Annual Conference of the European Association for Machine Translation*, pages 27–34, Antalya, Turkey, 2015.

Krings, Hans P and Geoffrey S Koby. *Repairing texts: empirical investigations of machine translation post-editing processes*, volume 5. Kent State University Press, 2001.

O'Brien, Sharon and Michel Simard. Introduction to special issue on post-editing. *Machine Translation*, 28(3-4):159–164, 2014.

Papineni, Kishore, Salim Roukos, Todd Ward, and Wei-Jing Zhu. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, 2002.

Shen, Shiqi, Yong Cheng, Zhongjun He, Wei He, Hua Wu, Maosong Sun, and Yang Liu. Minimum Risk Training for Neural Machine Translation. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1683–1692, Berlin, Germany, August 2016. URL `http://www.aclweb.org/anthology/P16-1159`.

Snover, Matthew, Bonnie Dorr, Richard Schwartz, Linnea Micciulla, and John Makhoul. A study of translation edit rate with targeted human annotation. In *Proceedings of the Meeting of the Association for Machine Translation in the Americas*, volume 200, pages 223–231, 2006.

Snover, Matthew, Nitin Madnani, Bonnie J. Dorr, and Richard Schwartz. Fluency, adequacy, or HTER?: exploring different human judgments with a tunable MT metric. In *Proceedings of the Fourth Workshop on Statistical Machine Translation*, pages 259–268, Athens, Greece, 2009. Association for Computational Linguistics.

Specia, Lucia and Atefeh Farzindar. Estimating machine translation post-editing effort with HTER. In *Proceedings of the Second Joint EM+/CNGL Workshop Bringing MT to the User: Research on Integrating MT in the Translation Industry*, pages 33–41, Denver, USA, 2010.

Specia, Lucia and Radu Soricut. Quality estimation for machine translation: preface. *Machine Translation*, 27(3-4):167–170, 2013.

Toral, Antonio and M. Víctor Sánchez-Cartagena. A Multifaceted Evaluation of Neural versus Phrase-Based Machine Translation for 9 Language Directions. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 1063–1073. Association for Computational Linguistics, 2017. URL `http://aclweb.org/anthology/E17-1100`.

Walt, Stéfan van der, S Chris Colbert, and Gael Varoquaux. The NumPy array: a structure for efficient numerical computation. *Computing in Science & Engineering*, 13(2):22–30, 2011.

Address for correspondence:
Mikel L. Forcada
`mlf@dlsi.ua.es`
Departament de Llenguatges i Sistemes Informàtics
Universitat d'Alacant, E-03690 Sant Vicent del Raspeig, Spain

PBML

The Prague Bulletin of Mathematical Linguistics
NUMBER 108 JUNE 2017 197–208

Unraveling the Contribution of Image Captioning and Neural Machine Translation for Multimodal Machine Translation

Chiraag Lala, Pranava Madhyastha, Josiah Wang, Lucia Specia

University of Sheffield

Abstract

Recent work on multimodal machine translation has attempted to address the problem of producing target language image descriptions based on both the source language description and the corresponding image. However, existing work has not been conclusive on the contribution of visual information. This paper presents an in-depth study of the problem by examining the differences and complementarities of two related but distinct approaches to this task: text-only neural machine translation and image captioning. We analyse the scope for improvement and the effect of different data and settings to build models for these tasks. We also propose ways of combining these two approaches for improved translation quality.

1. Introduction

There has been recent interest among the Machine Translation (MT) community in incorporating different modalities, such as images, to inform and improve machine translation, in contrast to learning from textual data only. For instance, the *Multimodal Machine Translation* (MMT) shared task (Specia et al., 2016) was introduced to investigate if images can potentially help the task of translating an image description (e.g. "*A brown dog is running after the black dog*") to a target language, given the description in a source language and its corresponding image as input (see Figure 1).

In the shared task, the organisers observed that image information is only useful in improving translations when used indirectly (e.g. for re-scoring n-best lists of text-only MT approaches). While this indicates that a text-only MT system is the primary contributor in MMT, it remains inconclusive whether image information can play a

 Corresponding author: clala1@sheffield.ac.uk
Cite as: Chiraag Lala, Pranava Madhyastha, Josiah Wang, Lucia Specia. Unraveling the Contribution of Image Captioning and Neural Machine Translation for Multimodal Machine Translation. The Prague Bulletin of Mathematical Linguistics No. 108, 2017, pp. 197–208. doi: 10.1515/pralin-2017-0020.

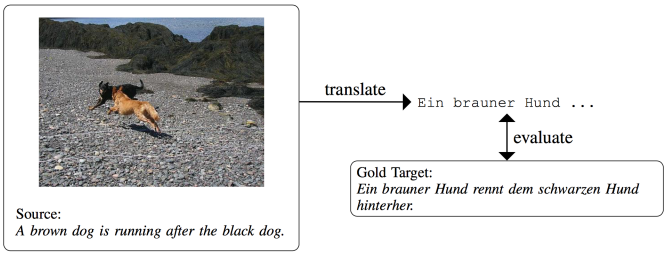


Figure 1: Multimodal Translation Task: source segment (English) and its human translation (German), against which system outputs are evaluated (Specia et al., 2016)

Figure 2: Example of an ambiguous word that could be solved with visual information. The word "hat" in English needs to be disambiguated in order to be translated as "Hut" in German (summer hat), rather than "Mütze" (winter hat)

more significant role. It would be counter-intuitive to simply rule out the contribution of images to the task, particularly when the text is descriptive of the image, which is the case in this dataset. An example (taken from our data) of where visual information can be helpful is shown in Figure 2. We, therefore, posit that visual information is indeed complementary to a text-only MT system for MMT, but the questions are: to what extent and in what way? To our knowledge, no extensive study has been done to understand the role that images play for the MMT task in a systematic manner.

To gain some insight into this matter, in this paper we isolate the text-only MT and the image description generation components of MMT. For the former, we use state-of-the-art Neural MT (NMT) models, which are based on a sequence-to-sequence neural architecture. For image captioning (IC)[1], we use state-of-the-art models based on multimodal recurrent neural networks as described in Vinyals et al. (2015) with default parameter settings. We build models for these two approaches using different datasets (parallel and target language only) and study their complementarities. Additionally, since the decoders of both the approaches perform approximately similar functions, we propose ways of combining the information coming from each model.

Our main contributions, therefore, are (i) an analysis of the individual contributions of a text-only NMT model and a monolingual but multimodal IC model to the MMT task by examining the effect of different data and model settings; and (ii) two

[1]We use the terms "image description" and "image caption" interchangeably.

new approaches for combining the outputs of NMT and IC models. In our experiments, the best-proposed combination approach outperforms the baseline.

2. Background

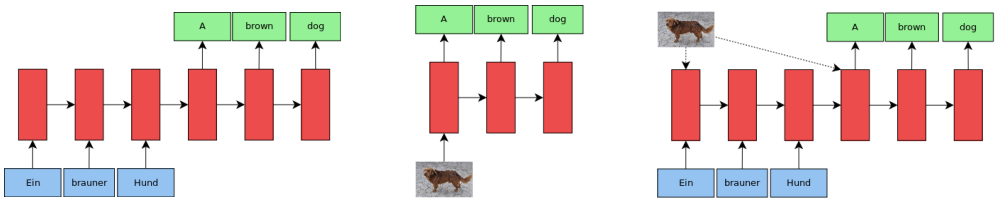
The standard approach in **Neural MT** uses an attention based encoder-decoder model that takes in a source sentence and encodes it using a Recurrent Neural Network (RNN) to produce a sequence of encoded vectors. The approach then decodes it using another RNN in the target language which is conditioned on the sequence of encoded vectors. The model searches through the encoded sequence vectors at each time step and aligns to the corresponding source hidden states adaptively (Bahdanau et al., 2015) (Figure 3a).

Early **Image Captioning** approaches were mainly based on generating a description using explicit visual detector outputs (Yao et al., 2010). We refer readers to Bernardi et al. (2016) for an in-depth discussion on various image captioning approaches. In recent years, multimodal RNN approaches have become dominant, achieving state-of-the-art results on the IC task (Vinyals et al., 2015). Such methods encode an input image as an embedding (e.g. Convolutional Neural Networks (CNN)) and learn an RNN for generating image descriptions conditioned on the image embedding. In this paper, we focus on such state of the art approaches, more specifically the system proposed by Vinyals et al. (2015) which uses a Long Short-Term Memory (LSTM) RNN to model the image descriptions (Figure 3b).

As a first attempt at **Multimodal Machine Translation**, Elliott et al. (2015) added image information at the encoder or the decoder in an NMT setup (Figure 3c) and found marginal improvements from doing so. The systems submitted to the subsequent shared task on Multimodal Machine Translation (Specia et al., 2016) mostly involved a type of NMT, i.e., an encoder-decoder approach, or used a standard phrase-based statistical MT (SMT) system. SMT systems made use of image information mostly during re-ranking, such as Shah et al. (2016). Hitschler et al. (2016) use image information by pivoting it on an external image captioning corpora. Most systems that make use of NMT add the image feature information into either the NMT encoder or decoder (Huang et al., 2016; Hokamp and Calixto, 2016), similar to Elliott et al. (2015) with various enhancements. Marginal improvements according to automatic evaluation metrics were found only for approaches using re-ranking. However, the results of the task do not provide an indication on whether this is inherently because of the task itself (i.e. images cannot help MT) or because of limitations of the methods proposed.

3. Experimental Settings

As Figure 3 shows, IC and NMT models are intrinsically similar from the perspective of decoding, producing the same type of output sequences. The primary differ-



(a) Neural MT *(b) IC* *(c) Multimodal NMT*

Figure 3: Typical architecture of NMT, IC, and MMT systems. In (a), the source sentence is encoded as a sequence of vectors and then decoded using a target language RNN. In (b), the input image is encoded as a vector, and a description is decoded using an RNN. In (c), the source sentence encoding is used as input to the decoder, and the image embedding is used as input to either the source encoder or target decoder

ence is the attention mechanism in NMT. In this section, we analyse the contributions of NMT and IC to a description translation task by studying various aspects of these systems independently and their impact on translation quality.

Dataset: We use the Multi30K dataset (Elliott et al., 2016), an extension of Flickr30K (Young et al., 2014) built for the WMT16 MMT task (Specia et al., 2016). Multi30K contains two variants: (i) one English description and a professionally translated German description per image (used in Task 1: multimodal translation); (ii) five English descriptions and five independently crowdsourced German descriptions per image (used in Task 2: image description generation). See Table 1 for detailed statistics. We use the data in the German–English (DE–EN) direction.

		Train	Val	Test	Tokens	Avg. Length
	Images	29,000	1,014	1,000	–	–
Task1	English	29,000	1,014	1,000	357,172	11.9
	German				333,833	11.1
Task2	English	145,000	5,070	5,000	1,841,159	12.3
	German				1,434,998	9.6

Table 1: Corpus statistics

Data Settings: To analyse the performance of the NMT and IC models with respect to different types of training data, we perform experiments in the following settings:

1. *Parallel*: The corpus for 'Task1' is used. Each image has a corresponding (DE, EN) description pair, where the DE description is a direct (professional) translation of the corresponding EN description.
2. *Comparable*: The corpus for 'Task2' is used. Each image has five independent (DE, EN) description pairs. The DE descriptions are obtained from the image only by crowdsourcing. They are much shorter than the English ones as com-

pared to the Task1 dataset (see Table 1). This is considered a comparable corpus, as the descriptions are not direct translations of each other.

3. *Out of Domain*: Here we train the models on larger datasets of different domains. For NMT, we take (News, etc.) data described in Sennrich et al. (2016), and for IC we take the MSCOCO corpus (Lin et al., 2014). These are large datasets and were not part of the MMT shared task at WMT16.
4. *Cross-comparable* (Only NMT): The corpus of 'Task2' is used to create a new dataset for NMT. Each of the five DE descriptions is randomly paired with each of the five EN descriptions resulting in 25 (DE, EN) description pairs per image. This is similar to the *Comparable* setting except that it is much larger.

All experiments were conducted using the Task1 test set of 1000 samples consisting one reference translation/description for each source sentence/image.

Toolkits: We use state-of-the-art toolkits: Nematus (Sennrich et al., 2016) for NMT and Show and Tell (Vinyals et al., 2015) for IC with default hyperparameters. We experiment with different beam sizes during decoding: 3, 10, 100 and 300. Besides the 1-best output, n-best outputs (where n is the beam size) are also generated from every model to provide a more comprehensive view of what the models can do. For NMT, in order to handle rare words, these are segmented into subwords using the Byte-Pair Encoding Compression Algorithm (Sennrich et al., 2015). We have also tried such a segmentation for IC, but no improvements were observed.

4. Analysis

In the following subsections, the effects of 'Data Setting' and 'Beam Size' on the performance of NMT and IC models are studied using 'Vocabulary Overlap', 'Perplexity', and the MT Metrics 'BLEU' and 'Meteor'. To study the effect of data settings, we fix the beam size to 10 and then train systems on the different training data sets. The data settings that gave the best performing NMT and IC systems are then fixed for the study on the effect of beam size, where we only vary the beam sizes. For a more holistic analysis, both 1-best and n-best outputs are used in our experiments.

4.1. Vocabulary Overlap and Perplexity

The vocabulary overlap between the system-generated outputs and gold standard references helps us to understand the performance of the systems at a very basic level. Given an NMT (or IC) system of beam size n, we denote i to be a test input (a DE sentence for NMT, an image for IC). Let $o_i^1, o_i^2, ..., o_i^n$ be the n-best hypotheses for input i, sorted in descending order by the log probability of o_i^k (i.e., the model score). Let r_i be the reference sequence for input i in the target language (EN). Let ϕ be the set function, $\oplus$ the concatenation operator, $\cap$ the intersection operator, and $|.|$ the cardinality.

We define four types of overlaps as follows:

$$\mathbb{V}_A(i) = \frac{|\phi(r_i) \cap \phi(o_i^1)|}{|\phi(r_i)|} \qquad \mathbb{V}_B(i) = \frac{|\phi(r_i) \cap \phi(o_i^1)|}{|\phi(o_i^1)|}$$

$$\mathbb{V}_C(i) = \frac{|\phi(r_i) \cap \phi(o_i^1 \oplus o_i^2 \oplus ... \oplus o_i^n)|}{|\phi(r_i)|} \qquad \mathbb{V}_D(i) = \frac{|\phi(r_i) \cap \phi(o_i^1 \oplus o_i^2 \oplus ... \oplus o_i^n)|}{|\phi(o_i^1 \oplus o_i^2 \oplus ... \oplus o_i^n)|}$$

$\mathbb{V}_A$ measures the proportion of words in the reference for Task1 captured by the 1-best output, while $\mathbb{V}_B$ measures the proportion of the words in the 1-best output found in the reference. $\mathbb{V}_C$ and $\mathbb{V}_D$ are similar to $\mathbb{V}_A$ and $\mathbb{V}_B$ respectively, except that the 1-best output is replaced by the concatenation of all n-best outputs. $\mathbb{V}_A$ and $\mathbb{V}_C$ correspond to word-overlap recalls, and $\mathbb{V}_B$ and $\mathbb{V}_D$ correspond to word-overlap precisions.

Perplexity scores measure how well the models (NMT and IC) can predict a sample. Given a system that generates a sequence $x_1, \cdots, x_m$ with probabilities $p_1, \cdots, p_m$, perplexity is defined as $\mathbb{P}(x) = 2^{\{-\sum_{i=1}^{m} p_i \log(p_i)\}}$. We use two types of perplexity measures $\mathbb{P}_A, \mathbb{P}_B$ based on whether the 1-best or n-best outputs of our systems are used: a) $\mathbb{P}_A(i) = \mathbb{P}(o_i^1)$ and b) $\mathbb{P}_B(i) = \frac{1}{n}\sum_{k=1}^{n} \mathbb{P}(o_i^k)$

	Data	$\mathbb{V}_A \uparrow$	$\mathbb{V}_B \uparrow$	$\mathbb{V}_C \uparrow$	$\mathbb{V}_D \uparrow$	$\mathbb{P}_A \downarrow$	$\mathbb{P}_B \downarrow$
NMT	News	61.24	63.41	69.83	**37.47**	11.25	12.57
	Task1	**66.11**	**68.27**	**73.02**	36.88	**4.78**	**5.76**
	Cross	26.22	44.23	34.91	19.76	11.16	13.11
	Task2	21.30	15.44	33.45	6.79	49.28	113.57
IC	MSCOCO	12.08	16.45	20.68	**11.16**	**10.22**	**12.38**
	Task1	11.38	14.19	24.76	6.35	19.50	39.59
	Task2	**17.70**	**26.29**	**30.04**	8.46	19.89	35.81

Table 2: Effect of training data studied using Vocabulary Overlaps $\mathbb{V}_A, \mathbb{V}_B, \mathbb{V}_C, \mathbb{V}_D$ (in %), and Perplexity $\mathbb{P}_A, \mathbb{P}_B$. All models are trained with a fixed beam size of 10

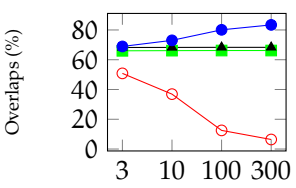
The sentences are pre-processed (removal of symbols and stop words, case-normalisation) to retain only content words. The vocabulary overlap and perplexity scores (averaged over all test inputs) are shown in Table 2 and Figure 4.

4.2. MT Metrics

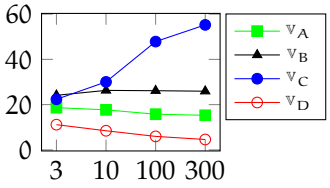
We evaluate the independent NMT and IC systems using BLEU (Papineni et al., 2002) and Meteor (Denkowski and Lavie, 2011). BLEU is computed using the script from Moses suite [2], and Meteor is computed using version 1.5 [3]. In addition, we also measure the ratio between the length of system-generated sequence over the length of reference ('len.'). The scores are tabulated in Tables 3 and 4.

[2] https://github.com/moses-smt

[3] http://www.cs.cmu.edu/~alavie/METEOR



(a) NMT and beam size



(b) IC and beam size

Beam	NMT $\mathbb{P}_A \downarrow$	NMT $\mathbb{P}_B \downarrow$	IC $\mathbb{P}_A \downarrow$	IC $\mathbb{P}_B \downarrow$
3	5.22	5.83	39.19	51.04
10	4.78	5.76	19.89	35.81
100	4.65	5.31	9.49	21.71
300	4.65	5.23	8.52	20.07

(c) Perplexity and beam size

Figure 4: Effect of beam size studied using vocabulary overlap $\mathbb{V}_A, \mathbb{V}_B, \mathbb{V}_C, \mathbb{V}_D$ (in %) and Perplexity $\mathbb{P}_A, \mathbb{P}_B$. Plot (a) shows vocabulary overlap of outputs of NMT system trained on Task1 data. Plot (b) shows vocabulary overlaps of outputs of IC system trained on Task2 data. Table (c) shows perplexity scores.

	Data	BLEU ↑	Meteor ↑	len. (%)
NMT	News	33.89	36.85	96.98
	Task1	**39.13**	**36.87**	**100.54**
	Cross	6.92	14.62	63.06
	Task2	3.08	12.83	158.07
IC	MSCOCO	3.11	9.56	78.45
	Task1	3.91	9.75	**86.37**
	Task2	**5.79**	**12.31**	75.55

Table 3: Effect of training data studied using MT evaluation metrics

	Beam	BLEU ↑	Meteor ↑	len. (%)
NMT	3	39.08	36.81	100.61
	10	39.13	36.87	100.54
	100	39.11	36.89	100.72
	300	39.11	36.89	100.72
IC	3	6.75	12.94	89.63
	10	5.79	12.31	75.55
	100	4.12	10.82	61.13
	300	3.83	10.47	58.73

Table 4: Effect of beam size studied using MT evaluation metrics

4.3. Discussion

Effect of Training Data: We observe that NMT models perform best when trained on the in-domain parallel Task1 data, with overlap $\mathbb{V}_A = 66.11\%$ and $\text{BLEU} = 39.13\%$ as summarised in Tables 2 and 3. We also observe that NMT performs sufficiently well when trained on the Out-of-Domain parallel News corpus with overlap $\mathbb{V}_A = 61.24\%$ and $\text{BLEU} = 33.89\%$. In the remaining comparable data settings (Cross and Task2) it performs very poorly, indicating that NMT system performance generally improves when constrained to parallel corpora and degrades when partially parallel corpora is added. The IC models perform best when trained on the in-domain Task2 data, which has 5 descriptions per image (see Table 1), with overlap $\mathbb{V}_A = 17.70\%$ and $\text{BLEU} = 5.79\%$ (or 20.52% when we use the five references of Task2). It performs poorly in other data settings. When compared to the NMT system, this can be seen as an indication that the ICs are better trained on larger in-domain data having multiple descriptions per image. We also observed that the IC system trained only on MSCOCO produced shorter sentences, resulting in lower perplexity scores.

Effect of Beam Size: By fixing Task1 data for NMT and Task2 data for IC and studying the effect of beam size, we observe that the NMT performance remains largely unchanged as the beam size changes (see Table 4) with $\texttt{BLEU} = 39.1\%$. On the other hand, the IC performance drops as beam size increases. We also observe that IC outputs shorter sentences with larger beam sizes. This is because an end-of-sentence token is more likely to be sampled (and sampled earlier) as beam size increases. Shorter captions are thus ranked higher as they end up having larger model scores (a product of target word probabilities). This may partly explain the performance decrease, although more work is needed to ascertain this. Another interesting observation from this experiment is that the n-best output from both NMT and IC is able to cover more content of the reference as the beam size n increases (See $\mathbb{V}_C, \mathbb{P}_A, \mathbb{P}_B$ in Figure 4). Especially for IC, the overlap $\mathbb{V}_C$ and perplexity measures show large improvements. For instance, $\mathbb{V}_C$ improves from 22.34% (beam 3) to 55.23% (beam 300). This shows that the n-best outputs are able to capture more information content in the reference as the beam size increases. In NMT we see a drastic fall in $\mathbb{V}_D$ from 50.83% (beam 3) to 6.41% (beam 300), which means that as the beam size increases the n-best output of NMT becomes very noisy, with many spurious words. We try to exploit these observations in our system combination strategies in later sections.

5. Combining NMT and IC for MMT

In the previous section, we analysed NMT and IC models independently and observed some important properties. Most notably, for IC the vocabulary overlap $\mathbb{V}_C$ increases drastically for larger beam sizes (see Figure 4) and becomes comparable to NMT models of smaller beam sizes. Recall that $\mathbb{V}_C$ is the overlap of content words in the n-best output (taken collectively) and the reference. This motivates us to explore the possibilities of improving MT by combining the n-best outputs of NMT and IC models of different beam sizes at the word-level.

We approach this task as that of re-ranking the n-best outputs of NMT models using the m-best outputs from IC models. To motivate this, we first explore the scope for improvement with re-ranking through an oracle experiment.

5.1. Scope for Re-ranking: Oracle Experiment

The oracle experiment assumes that we have an 'oracle' that always chooses the best translation out of the n-best outputs generated by the system. We compute an upper bound on the performance of re-ranking approaches using this oracle. For a given MT-metric (we used BLEU) we use the reference translation to obtain the best translation given an n-best list of translation hypotheses.

This experiment was performed on the outputs of NMT systems trained on Task1 for beam sizes 10, 30, 100, and 300. The results are shown in Figure 5. We observe that an ideal re-ranking approach could significantly improve NMT performance. As the

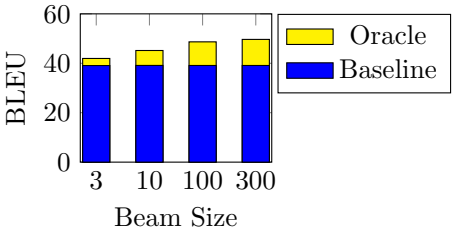


Figure 5: Scope for improvement, as indicated by the yellow bar over the baseline

beam size increases, the scope for obtaining a better translation generally improves. We also performed this experiment for IC systems, but no significant improvements were found. The best translation selected by the oracle is also observed to be usually close to the middle of the system-ranked n-best list. In the following sections, we focus on re-ranking the NMT hypotheses using IC outputs.

5.2. Re-ranking NMT using IC Word Probabilities

We propose to re-rank the n-best NMT translations using image information extracted as word probabilities in the m-best IC outputs. The decoders in both systems produce a word w with a probability $p_{nmt}(w)$ and $p_{ic}(w)$ respectively. We estimate new word scores for each word w by interpolating the information from both systems:

$$p_{new}(w) = (1 - \alpha) * p_{nmt}(w) + \alpha * p_{ic}(w)$$

where, $p_{new}(w)$ is the new word score, $p_{nmt}(w)$ is the word probability from the NMT system, $p_{ic}(w)$ is the *aggregated* word probability from the IC system, and α is a hyper-parameter in the range $[0, 1]$ tuned on the validation set using grid search. For a translation hypothesis $(w_1, w_2, ..., w_k)$, its score is computed as a product of these new word-level scores $\prod_{i=1}^{k} p_{new}(w_i)$. We re-rank the n-best NMT hypotheses using the new scores. We propose three ways of aggregating the word probability $\tilde{p}_{ic}(w^t)$ for the t^{th} instance of w in the m-best IC outputs:

1. AVERAGE: $p_{ic}^{avg}(w) = \frac{1}{L} \sum_{t=1}^{L} \tilde{p}_{ic}(w^t)$
2. SUM: $p_{ic}^{sum}(w) = \sum_{t=1}^{L} \tilde{p}_{ic}(w^t)$
3. MAX: $p_{ic}^{max}(w) = \max_{t \in [1,2,...,L]} \tilde{p}_{ic}(w^t)$

where the word w occurs L times in the m-best IC outputs. We set $p_{ic}(w) = 0$ if w does not occur in any of the outputs.

5.3. Re-ranking NMT by similarity with IC Outputs

Here we explore re-ranking NMT hypotheses by their similarity to IC outputs. The motivation is that if we assume the IC outputs accurately describe image content, a

more adequate translation can be selected from the NMT hypotheses if we include the IC outputs in the re-ranking process. We do this by using the BLEU metric as a measure of overlap between an NMT hypothesis and the m-best IC outputs. The NMT hypothesis that has the highest n-gram overlap with the IC outputs should be the most adequate translation. This implies that we are re-ranking the NMT hypotheses based on the information overlap score. For this paper, we use BLEU-4 with smoothing and brevity penalty as the overlap score. We call this approach 'BLEU-rerank'.

5.4. Results and Human Evaluation

For both system combination strategies, the best results are obtained using the NMT system trained on Task1 data and decoded with beam size 10 and the IC system trained on Task2 data with beam size 100 (except for BLEU-rerank where both NMT and IC systems have beam size 3). The highest ranked output after re-ranking is used for evaluation. We report the 1-best output of the same NMT system (before re-ranking) as the baseline. We summarise the results in Table 5. We observe that the method that uses *IC word probabilities* is able to select better sentences. The AVERAGE aggregation works best and gives a small improvement when evaluated with BLEU. Given that the improvement is only observed for BLEU, we resorted to manual evaluation to obtain a better understanding of our re-ranking approaches.

Re-Ranking	α	BLEU↑	Meteor↑
AVERAGE	0.41	**39.43**	36.72
SUM	0.0049	39.34	36.65
MAX	0.26	39.30	36.67
NMT BASELINE	–	39.13	**36.87**
BLEU-rerank	–	36.20	35.30

Table 5: Performance of re-ranking strategies

Judge	Either	Baseline	AVERAGE
A	17	15	18
B	5	19	26
C	22	9	19
D	19	11	20
E	27	9	14
Total	90 (36%)	63 (25%)	97(39%)

Table 6: Human evaluation: NMT vs MMT

Human evaluation: 31% of the 1-best outputs of AVERAGE differ from the baseline after re-ranking. To better understand the differences in these sentences, we asked humans to judge their quality. Five judges (proficient in English) were given 50 samples, each showing the source input image, reference translation, and the translation options from the two systems (without revealing the systems). The judges were asked to decide which option was better in terms of (i) proximity in meaning to the reference and (ii) fluency, giving precedence to the former. They could choose 'Either' when the two translations were equally good or bad. Table 6 summarises the results. All five judges preferred AVERAGE over the text-only baseline.

Figure 6 shows an example output comparing 1-best translation of the text-only baseline and our proposed 'AVERAGE' system combination strategy. The IC system-generated captions give high word probability scores to the words *rocky* and *mountain* compared to the words *body* and *water* [$\mathrm{p}_{\mathrm{ic}}^{\mathrm{avg}}(\mathtt{rocky}) = 0.42$; $\mathrm{p}_{\mathrm{ic}}^{\mathrm{avg}}(\mathtt{mountain}) = 0.28$;

Reference a dog treads through a shallow area of water located on a rocky mountainside.

Baseline a dog walks through a body of water, with a body of water in it.

AVERAGE a dog walks through a body of water, looking at a rocky mountain.

Figure 6: Example output translation for the baseline (text-only NMT) and the best MMT system combination (AVERAGE)

$p_{ic}^{avg}(body) = 0.00; p_{ic}^{avg}(water) = 0.00$]. This is probably because rocky mountain is more prominent in the image. This indicates that there is scope for developing system combination methods and joint models that combine both IC and NMT systems.

6. Conclusions

In this paper, we studied text-only NMT and IC systems independently from each other. The NMT system was found to be better when constrained to an in-domain parallel corpus; its performance degrades when trained on a partly parallel corpus. On the other hand, the IC system was found to be better when trained on a corpus that has multiple descriptions of the same image, enabling the model to capture more information content more reliably from the image. n-best outputs of the IC system are able to capture more information content for higher beam sizes. For NMT, the oracle experiment suggests that there is enormous potential to improve performance for higher beam sizes n if we can re-rank the n-best output wisely. However, we also see the $\mathbb{V}_D$ precision decreases dramatically for NMT with higher beam sizes, suggesting higher chances of spurious re-ranking and, hence, the need to find the right trade-off between more information and spurious information. In our attempt to combine outputs from NMT and IC, we found that system combinations can be helpful if we make use of word probabilities from NMT and IC systems. Our method interpolating these probabilities is able to use image information and outperforms the baseline. This shows evidence that image information has potential to improve MT. Creative and robust system combinations and joint models that exploit NMT and IC word probabilities are promising directions for future work.

Acknowledgements: This work was supported by the MultiMT project (H2020 ERC Starting Grant No. 678017). The authors also thank the anonymous reviewers for their valuable comments.

Bibliography

Bahdanau, Dzmitry, Kyunghyun Cho, and Yoshua Bengio. Neural Machine Translation by Jointly Learning to Align and Translate. In *ICLR*, 2015.

Bernardi, Raffaella, Ruket Cakici, Desmond Elliott, Aykut Erdem, Erkut Erdem, Nazli Ikizler-Cinbis, Frank Keller, Adrian Muscat, and Barbara Plank. Automatic Description Generation from Images: A Survey of Models, Datasets, and Evaluation Measures. *Journal of Artificial Intelligence Research*, 55:409–442, 2016.

Denkowski, Michael and Alon Lavie. Meteor 1.3: Automatic Metric for Reliable Optimization and Evaluation of Machine Translation Systems. In *WMT*, 2011.

Elliott, Desmond, Stella Frank, and Eva Hasler. Multi-Language Image Description with Neural Sequence Models. *CoRR*, abs/1510.04709, 2015.

Elliott, D., S. Frank, K. Sima'an, and L. Specia. Multi30K: Multilingual English-German Image Descriptions. In *5th Workshop on Vision and Language*, pages 70–74, 2016.

Hitschler, Julian, Shigehiko Schamoni, and Stefan Riezler. Multimodal Pivots for Image Caption Translation. In *Association for Computational Linguistics*, pages 2399–2409, 2016.

Hokamp, Chris and Iacer Calixto. Multimodal neural machine translation using minimum risk training, 2016. URL `https://www.github.com/chrishokamp/multimodal_nmt`.

Huang, Po-Yao, Frederick Liu, Sz-Rung Shiang, Jean Oh, and Chris Dyer. Attention-based Multimodal Neural Machine Translation. In *WMT*, pages 639–645, 2016.

Lin, Tsung-Yi, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft COCO: Common Objects in context. In *European Conference on Computer Vision*, pages 740–755. Springer, 2014.

Papineni, Kishore, Salim Roukos, Todd Ward, and Wei-Jing Zhu. BLEU: A Method for Automatic Evaluation of Machine Translation. In *ACL*, pages 311–318, 2002.

Sennrich, Rico, Barry Haddow, and Alexandra Birch. Neural Machine Translation of Rare Words with Subword Units. In *Proceedings of ACL*, pages 1715–1725, 2015.

Sennrich, Rico, Barry Haddow, and Alexandra Birch. Edinburgh Neural Machine Translation Systems for WMT 16. In *WMT*, pages 371–376, 2016.

Shah, Kashif, Josiah Wang, and Lucia Specia. SHEF-Multimodal: Grounding Machine Translation on Images. In *WMT*, pages 660–665, 2016.

Specia, Lucia, Stella Frank, Khalil Sima'an, and Desmond Elliott. A Shared Task on Multimodal Machine Translation and Crosslingual Image Description. In *WMT*, pages 543–553, 2016.

Vinyals, Oriol, Alexander Toshev, Samy Bengio, and Dumitru Erhan. Show and Tell: A Neural Image Caption Generator. In *IEEE Conference on Computer Vision & Pattern Recognition*, 2015.

Yao, Benjamin Z., Xiong Yang, Liang Lin, Mun Wai Lee, and Song Chun Zhu. I2T: Image Parsing to Text Description. *Proceedings of the IEEE*, 98(8):1485–1508, 2010. ISSN 0018-9219.

Young, Peter, Alice Lai, Micah Hodosh, and Julia Hockenmaier. From image descriptions to visual denotations: New similarity metrics for semantic inference over event descriptions. *Transactions of the Association for Computational Linguistics*, 2:67–78, 2014.

Address for correspondence:
Chiraag Lala
`clala1@sheffield.ac.uk`
Department of Computer Science, The University of Sheffield
Regent Court, 211 Portobello, Sheffield, S1 4DP, United Kingdom

PBML

The Prague Bulletin of Mathematical Linguistics
NUMBER 108 JUNE 2017 209–220

Comparing Language Related Issues for NMT and PBMT between German and English

Maja Popović

Humboldt University of Berlin

Abstract

This work presents an extensive comparison of language related problems for neural machine translation and phrase-based machine translation between German and English. The explored issues are related both to the language characteristics as well as to the machine translation process and, although related, are going beyond typical translation error classes. It is shown that the main advantage of the NMT system consists of better handling of verbs, English noun collocations, German compound words, phrase structure as well as articles. In addition, it is shown that the main obstacles for the NMT system are prepositions, translation of English (source) ambiguous words and generating English (target) continuous tenses. Although in total there are less issues for the NMT system than for the PBMT system, many of them are complementary – only about one third of the sentences deals with the same issues, and for about 40% of the sentences the issues are completely different. This means that combination/hybridisation of the NMT and PBMT approaches is a promising direction for improving both types of systems.

1. Introduction

Neural machine translation (NMT), a new paradigm to statistical machine translation (SMT), has emerged very recently and has already surpassed the performance of the mainstream approach in the field, phrase-based MT (PBMT) for a number of language pairs. In PBMT, different models (translation, reordering, target language, etc.) are trained independently and combined in a log-linear scheme in which each model is assigned a different weight by a tuning algorithm. On the contrary, in NMT all the components are jointly trained to maximise translation quality. On one side, NMT represents a simplification – a large recurrent network trained for end-to-end

 Corresponding author: maja.popovic@hu-berlin.de
Cite as: Maja Popović. Comparing Language Related Issues for NMT and PBMT between German and English. The Prague Bulletin of Mathematical Linguistics No. 108, 2017, pp. 209–220. doi: 10.1515/pralin-2017-0021.

translation is considerably simpler than a PBMT system which integrates multiple components and processing steps. On the other side, the NMT process is less transparent.

So far, the translations produced by NMT systems have been evaluated mostly in terms of overall performance scores, both by automatic and by human evaluations. This has been the case of last year's news translation shared task at the First Conference on Machine Translation (WMT16). In this translation task, outputs produced by different MT systems were evaluated (i) automatically, by various evaluation metrics, and (ii) manually, by means of ranking translations or by assigning them an overall quality score. In all those evaluations, the performance of each system is measured by means of an overall score which provides useful information about general performance of the system but does not provide any additional information.

To the best of our knowledge, only two detailed analyses of the NMT approach and comparisons with PBMT approach have been carried out so far. (Bentivogli et al., 2016) conducted a detailed analysis for the English-to-German translation of transcribed TED talks and found out that NMT (i) decreases post-editing effort, (ii) degrades faster than PBMT with sentence length and (iii) results in a notable improvement regarding reordering, especially for verbs. (Toral and Sánchez-Cartagena, 2017) go further in this direction by conducting a multilingual and multifaceted evaluation and found out that (i) NMT outputs are considerably different than PBMT outputs, (ii) NMT outputs are more fluent, (iii) NMT systems introduce more reorderings than PBMT systems, (iv) PBMT outperforms NMT for very long sentences and (v) NMT performs better in terms of morphological and reordering errors across all language pairs.

In this paper, we go in slightly different direction by identifying and comparing language related issues for two German-English systems, one NMT and one PBMT, in both translation directions. Identification of language related issues for machine translation has begun relatively recently (e.g. (Popović and Arčan, 2015), (Comelles et al., 2016)) and, although related, goes beyond the standard error classification task. Definition of issues is based both on general linguistic knowledge as well as on the phenomena related to the (machine) translation process.

The issues are manually identified for 267 English-to-German source sentences and 204 German-to-English source sentences from the WMT16 News domain data and their translations by NMT and PBMT systems.

The main goals of the experiments are:

1. to compare overall distributions of issues for the NMT and the PBMT system and identify the particular strengths of the NMT approach, i.e. particular weaknesses of the PBMT approach for each translation direction;
2. to examine the overlap between issue types in two systems in order to determine if the NMT approach simply handles all the phenomena better, or there are complementary differences. This is an important question for better understanding

potentials and limits of combination and hybridisation of the two approaches which already has shown some promising results (Niehues et al., 2016).

We choose the German-English language pair in both directions because it has been known as a rather hard one for PBMT and the improvements yielded by the NMT approach are large, especially when translating into German. Our analyses are conducted on the Edinburgh University submissions of NMT and PBMT systems to the WMT16 translation task for each language direction which were (one of) the best ranked. This (i) guarantees the reproducibility of our results as all the MT outputs are publicly available, (ii) ensures that the systems evaluated are state-of-the-art, as they are the result of the latest developments at a top MT research group worldwide. If the paper is accepted, the annotated texts with issue labels will be made publicly available, too.

We believe that our evaluation results will be of interest to the wider research community, both regarding development of NMT and PBMT systems as well as regarding development of MT evaluation and error analysis methods.

2. Related work

The first detailed analysis and comparison between the NMT and PBMT approach is carried out in (Bentivogli et al., 2016). They analysed 600 sentences from IWSLT transcriptions of TED talks (i.e. spoken language) translated from English into German. They conducted automatic analysis on manually post-edited data in terms of morphological, lexical and ordering errors together with the fine grained analysis of ordering errors and found out that the main advantage of NMT approach is better ordering, especially for verbs.

(Toral and Sánchez-Cartagena, 2017) performed a multifaceted automatic analysis based on independent human reference translations for nine language pairs from news domain. The analysis consists of output similarity, fluency measured by LM perplexity, degree of reordering as well as three broad error classes: morphological, reordering and lexical errors. The main findings confirm the results from previous publication, i.e. the reduction of morphological and reordering errors by NMT. In addition, both publications report degradation of the NMT approach for long sentences.

While both publications report results of an extensive analysis and comparison of NMT and PBMT approaches, neither of publications deals with language related issues based on the source and the target language properties and their differences.

The first step towards such analysis is reported in (Farrús et al., 2010) where a simple error scheme containing five broad classes is used for comparison of two Spanish-Catalan SMT systems. This scheme is then further expanded in (Comelles et al., 2016) by identifying and classifying relevant linguistic features for the English-Spanish language pair based on general linguistic knowledge as well as on the phenomena occurring in the given corpus. The linguistic issue taxonomy is used for development of

a linguistically motivated automatic evaluation metric VERTa (Comelles et al., 2012) which enables using different combinations of the described linguistic features.

Similar analysis is conducted in (Popović and Arčan, 2015) where problematic patterns for PBMT between South Slavic languages on one side and English and German on another side were identified and analysed.

Nevertheless, none of the publications dealing with linguistically motivated issues includes analysis of an NMT system, nor the German-English pair.

3. Language related issues

Identification of language related issues has begun rather recently, so there are still no strict guidelines regarding their definition. In any case, the issues have to be linguistically motivated so that they can reflect the (un)ability of a machine translation system to translate specific linguistic phenomena. However, they should not only contain traditional linguistic categories but also categories which are related to the (machine) translation process. The issues should be clearly defined and widely understandable so that the results can be easily understood and shared.

Although issue identification task is related to error classification task, it goes beyond it: some of the issues defined so far directly correspond to some typical error classification categories, such as "verb form" or "mistranslation", however for a number of issues such relation is still hard to find.

For example, when an MT system does not handle a source German compound properly, error categories in the English output can be "mistranslation", "missing word" (components are missing), "word order" (components are in incorrect position), but the issue label for each of these cases would be "compound word".

Annotation was carried out by researchers familiar with human and machine translation process. The source language, its reference translation, and the two translation outputs in random order were given to the annotators.

The most prominent issues for both translation directions are:

- **ambiguous source word**
  The obtained translation for the given word is in principle correct, but not in the given context.
- **article**
  Rules for articles in German and English differ – therefore, some of the articles are added, missing, or incorrectly translated as (in)definite. In addition, some of the German articles are incorrectly inflected.
- literal translation
  Word-by-word translated parts.
- **mistranslation**
  Incorrect translation of words or word groups.
- **source multiword expression**
  Failing to treat a multiword expression as a whole.

- **MT phrase structure**
  Phrases/chunks are not treated properly so that the (group(s) of) words are misplaced, mistranslated and/or incorrectly inflected. "MT" refers to the fact that these are not linguistic phrases.
- **preposition**
  Mostly mistranslated, sometimes omitted or added.
- **verb**
  Problems with translation of verbs: main, auxiliary, modal, participle, formation of tenses, order, etc.
  - **form**
    Verb inflection does not correspond to the person and/or the tense.
  - **order**
    Verb or verb parts are misplaced.
  - **missing**
    Verb or verb parts are missing.

For English-to-German translation:

- **noun collocation**
  English sequence consisting of a head noun and additional nouns and adjectives is incorrectly translated, often into an unintelligible construction.
  - **noun collocation + compound**
    English noun collocation which corresponds to an incorrectly formed German compound word. The German compound word is mistranslated, or there are problems with components: missing, added or separated.

For German-to-English translation:

- **German compound**
  German compound is mistranslated or remained untranslated, or there are problems with components: missing, added or in incorrect order.
- **English continuous verb tenses**
  Continuous verb tenses do not exist in German, so that English present/past continuous tense is often substituted by simple present/past tense, or there are problems with verb parts.

4. Data sets

The texts used in the described experiments consist of 267 English-to-German source sentences and 204 German-to-English source sentences from the WMT16 News domain data and their NMT and PBMT translations. The annotation process is still fully manual, so that annotating the whole test sets each consists of about 3000 sentences would be too intensive. Therefore the smaller subsets were extracted from the set of the sentences which participated in human ranking, in order to also enable future experiments concerning relationship between issues and ranks. For the same

direction	system	BLEU	chrF
en→de	NMT	35.0	61.9
	PBMT	31.5	58.5
de→en	NMT	42.5	66.5
	PBMT	38.9	66.2

Table 1. Overall automatic scores BLEU and chrF on analysed texts for both systems and both translation directions.

reason, only two systems were analysed, one NMT and one PBMT. (Partial) automatisation of the annotation process should be certainly part of the future work.

The NMT system (Sennrich et al., 2016) is based on attentional encoder-decoder and operates on subword units. In addition, back-translations of the monolingual News corpus is used as additional training data. This system is ranked as the best for both translation directions.

The PBMT system (Williams et al., 2016) is a Moses based system which follows the standard PBMT approach of scoring translation hypotheses using a weighted linear combination of features. The core features are 5-gram LM model, phrase translation and lexical translation scores, word and phrase penalties and a linear distortion score. Tuning of model weights is performed by k-best batch MIRA.

Although other systems were ranked better in the WMT16 task, we decided to use this one because it has been developed by the same group, and we believe that therefore the comparison is more reliable.

5. Results

5.1. Overall automatic scores

First, in Table 1 we report the overall BLEU (Papineni et al., 2002) and chrF (Popović, 2015) scores for the analysed texts. The NMT system clearly outperforms the PBMT system for both translation directions and by both scores. It can be noted that the absolute chrF improvement is larger for translation into German, indicating that NMT introduces morphological improvements.

5.2. Comparison of issue distributions

The frequencies of the most prominent issues for the NMT and the PBMT system are presented in Table 2. Since the issues are defined on the sentence level, the numbers in tables represent raw issue counts normalised by the total number of sentences. For example, the verb form issues for English→German translation are interpreted as follows: from 100 English source sentences, verb form problems occur in 4.9 sentences translated by NMT and in 9.4 sentences translated by PBMT.

In addition, percentages of correct sentences ("no issues") as well as of sentences for which it was difficult to define any particular issue ("difficult to analyse") are shown.

First, it can be seen that the percentage of correct sentences[1] is significantly higher for the NMT system than for the PBMT system. As for "difficult" sentences, there is almost no difference between the systems, only between the translation directions – there are more for English-to-German.

As for the issue types, for both translation directions the NMT system clearly outperforms the PBMT system for:

- verbs in the following aspects: form, order and omission
- articles
- English noun collocations and German compounds
- phrase structure

These findings, while shedding different kind of light on the strengths and weaknesses of the two approaches, also confirm the results reported in previous work, namely that one of the main advantages of the NMT approach is better dealing with morphology and ordering, especially for verbs. Verb forms and German compounds clearly represent morphological challenges, whereas both morphology and order are implicitly related to phrase structure and treatment of noun collocations. Since all these issues are strongly related to fluency, the fluency improvements reported in related work are corroborated, too.

The results also show that for some issue types the behaviour depends on the translation direction, so that NMT outperforms PBMT for:

- ambiguous words and literal translations for German to English
- mistranslation and multiword expressions for English to German

but for the opposite translation direction these issues are better handled by the PBMT system.

Furthermore, target English continuous tenses are slightly better handled by PBMT, and represent the most frequent obstacle for German-to-English NMT translation (11.7%).

Finally, it can be observed that the prepositions are rather problematic for both systems. They are the most frequent issue for the English-to-German NMT system and second frequent (after continuous tenses) for the other translation direction, so the future work on NMT improvement should take this into account.

Sentence length

Previous work reported significance of the sentence length, namely that the PBMT approach outperforms NMT for longer sentences. Therefore we also investigated issue distributions for different sentence lengths. Nevertheless, we have found neither

[1] About 8% of sentences is identical to the corresponding reference translation.

English→German issue type	system NMT	PBMT
no issues	**35.7**	20.2
difficult to analyse	5.6	6.4
(src) ambiguous word	15.4	**10.5**
article	**8.5**	15.8
literal	6.7	**6.0**
mistranslation	**5.6**	7.5
(src) multiword expression	**4.9**	5.2
(src) noun collocation	**4.5**	7.1
+ *(tgt) compound*	**1.9**	7.1
(MT) phrase structure	**1.1**	5.6
preposition	17.5	**17.2**
verb – *form*	**4.9**	9.4
– *order*	**1.5**	10.9
– *missing*	**1.5**	24.0

German→English issue type	system NMT	PBMT
no issues	**39.0**	26.3
difficult to analyse	3.9	3.9
(src) ambiguous word	**9.3**	10.7
article	**7.8**	13.7
compound	**4.4**	7.8
literal	**4.4**	9.8
mistranslation	9.3	**8.3**
(src) multiword expression	4.4	**3.4**
(MT) phrase structure	**2.0**	6.8
preposition	11.2	**10.2**
verb – *form*	**2.0**	2.9
– *order*	**0.5**	5.8
– *missing*	**1.0**	5.8
– *continuous tense*	11.7	**8.3**

Table 2. Percentage of issues (raw counts normalised over the total number of sentences) for English-to-German (above) and German-to-English (below) translation.

overlap degree	% of sentences	
	en→de	de→en
complete (100%)	27.3	31.9
high (>50%)	9.7	13.7
low (≤50%)	20.6	16.2
none (0%)	42.4	38.2

Table 3. Percentage of sentences with four distinct overlap degrees between NMT and PBMT issues: complete overlap (100%), high overlap (>50%), low overlap (≤50%) and no overlap (0%).

a relation between issue types and sentence length, nor advantages of the PBMT system for longer sentences. It should be noted that the maximal sentence length in our data set was 36 words, whereas the results reported in previous work show that important changes start for sentences longer than 40 words. Therefore this aspect should be investigated thoroughly in future work.

5.3. Overlap between PBMT and NMT issues

The results described in previous section have shown that the NMT system does not simply outperform the PBMT system by having less issues of all types, but that there are certain complementary differences. In order to explore overlapping and complementary issues, we carried out the following experiments.

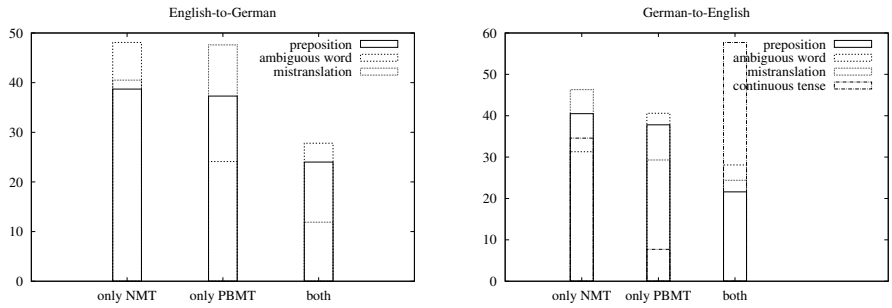
As a first step, we calculated overall overlap of the issues for each translation direction in the form of the F-score. For English-to-German this score is 37.9%, and for German-to-English 44.6%. These scores are not very high, indicating that there is a number of complementary issues.

The next step was to calculate the overlap F-score for each sentence and then divide the sentences into four groups: 1) complete overlap, same issues (100%), 2) high overlap (between 50 and 100%), 3) low overlap (between 0 and 50%) and 4) no overlap, completely different issues (0%).

The distributions of sentences over these four overlap degree groups are shown in Table 3 for both translation directions, and it can be seen that:

- only about one third of the sentences has identical issues;
- the majority (about 40%) of sentences have completely different issues;
- there are more sentences with low overlap than those with high overlap.

These findings show that, although the NMT approach surely performs better than the PBMT approach, there are complementary problems and errors. We believe this is an important finding because it means that there is a room for improvement of both systems in terms of combination and hybridisation.



(a) English→German (b) German→English

Figure 1. Distribution (%) of complementary and identical issues.

The last step in this direction was to examine which are the most frequent overlapping issues as well as how much of the prominent NMT issues is complementary with the PBMT ones.

First part of the analysis showed that the majority of the identical sentences are either correct, or are sentences for which it was hard to define issues.

As for the most prominent NMT issues, namely prepositions, ambiguous words, mistranslations and English continuos tenses, the percentages of complementary and overlapping occurrences is shown in Figure 1 for both translation directions. It can be seen that about 20-50% of total occurrences of the particular issues are complementary, i.e. do not overlap. The only exception is the verb continuous tense where the overlap is large. These results indicate that the combination of NMT and PBMT approach could ”help” dealing with prepositions and lexical issues (mistranslations and ambiguous words).

6. Summary and outlook

We have conducted an extensive comparison between NMT and PBMT language related issues for the German-English language pair in both translation directions. Our aim has been to shed additional light on the strengths and weaknesses of both approaches, as well as to explore if there are complementary issues.

Following the two main goals of our experiments presented in Introduction, our main findings are:

1. The particular strengths of the NMT approach are better handling of (i) verb order, forms and avoiding verb omissions, (ii) English noun collocations and German compound words, (iii) articles and (iv) phrase structure. All these is-

sues are completely or strongly related to morphology and word order, and to fluency as well, which corroborates the results reported in previous work.

2. Although the NMT approach in total has less issues, there is a number of sentences with complementary issues. This finding can help improvement of both systems by means of combination and/or hybridisation.

Additional important findings are:

- dominant problems for the NMT system are prepositions, translation of English ambiguous words into German and forming English verb continuous tenses;
- most occurrences of prepositions, ambiguous words and mistranslations are complementary.

It should also be noted that translating prepositions represents an important obstacle for both systems and it should be addressed in future work. Apart of this, there is a number of other directions for future work, such as (i) improvement of one or both systems by addressing some of the most prominent issues, (ii) exploring combination of two approaches, (iii) investigating other language pairs, (iv) working towards (partial) automatisation of the annotation process in order to achieve scalability.

We believe that our evaluation results will be of interest both for development of NMT and PBMT systems as well as for development of MT evaluation and error analysis methods. We conducted all experiments on publicly available data, and the annotated texts are also publicly available[2].

Acknowledgements

This research has received funding from the European Union's Horizon 2020 research and innovation programme TraMOOC under Grant Agreement No. 644333.

Bibliography

Bentivogli, Luisa, Arianna Bisazza, Mauro Cettolo, and Marcello Federico. Neural versus Phrase-Based Machine Translation Quality: a Case Study. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing (EMNLP 2016)*, pages 257–267, Austin, Texas, November 2016.

Comelles, Elisabet, Jordi Atserias, Victoria Arranz, and Irene Castellón. VERTa: Linguistic Features in MT Evaluation. In *Proceedings of the 8th International Conference on Language Resources and Evaluation (LREC 2012)*, Istanbul, Turkey, May 2012.

Comelles, Elisabet, Victoria Arranz, and Irene Castellón. Guiding Automatic MT Evaluation by Means of Linguistic Features. *Digital Scholarship in the Humanities*, September 2016.

Farrús, Mireia, Marta Ruiz Costa-Jussà, José Bernardo Mariño, and José Adrián Rodríguez Fonollosa. Linguistic-based Evaluation Criteria to Identify Statistical Machine Translation Errors. In *Proceedings of the 14th Annual Conference of the European Asso ciation for Machine Translation (EAMT 2010)*, pages 167–173, Saint-Raphael, France, May 2010.

[2] `https://github.com/m-popovic/german-english_pbmt-nmt-issues`

Niehues, Jan, Eunah Cho, Thanh-Le Ha, and Alex Waibel. Pre-Translation for Neural Machine Translation. In *Proceedings of the 26th International Conference on Computational Linguistics (CoLing 2016)*, pages 1828–1836, Osaka, Japan, December 2016.

Papineni, Kishore, Salim Roukos, Todd Ward, and Wie-Jing Zhu. BLEU: a Method for Automatic Evaluation of Machine Translation. In *Proceedings of the 40th Annual Meeting of the Association for Computa tional Linguistics (ACL 2002)*, pages 311–318, Philadelphia, PA, July 2002.

Popović, Maja. chrF: Character n-gram F-score for Automatic MT Evaluation. In *Proceedings of the 10th Workshop on Statistical Machine Translation (WMT 2015)*, pages 392–395, Lisbon, Portugal, September 2015.

Popović, Maja and Mihael Arčan. Identifying Main Obstacles for Statistical Machine Translation of Morphologically Rich South Slavic languages. In *Proceedings of the 18th Annual Conference of the European Association for Machine Translation (EAMT 2015)*, Antalya, Turkey, May 2015.

Sennrich, Rico, Barry Haddow, and Alexandra Birch. Edinburgh Neural Machine Translation Systems for WMT16. In *Proceedings of the 1st Conference on Machine Translation (WMT 2016)*, pages 371–376, Berlin, Germany, August 2016.

Toral, Antonio and Víctor Manuel Sánchez-Cartagena. A Multifaceted Evaluation of Neural versus Statistical Machine Translation for 9 Language Directions. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics (EACL 2017)*, Valencia, Spain, April 2017.

Williams, Philip, Rico Sennrich, Maria Nadejde, Matthias Huck, Barry Haddow, and Ondřej Bojar. Edinburgh's Statistical Machine Translation Systems for WMT16. In *Proceedings of the 1st Conference on Machine Translation (WMT 2016)*, pages 399–410, Berlin, Germany, August 2016.

Address for correspondence:
Maja Popović
maja.popovic@hu-berlin.de
Humboldt University of Berlin
Unter den Linden 6, Berlin, Germany

PBML

The Prague Bulletin of Mathematical Linguistics
NUMBER 108 JUNE 2017 221–232

Rule-Based Machine Translation for the Italian–Sardinian Language Pair

Francis M. Tyers,[a b] Hèctor Alòs i Font,[a] Gianfranco Fronteddu,[e] Adrià Martín-Mor[d]

[a] UiT Norgga árktalaš universitehta, Tromsø, Norway
[b] Arvutiteaduse instituut, Tartu Ülikool, Tartu, Estonia
[c] Universitat de Barcelona, Barcelona
[d] Universitat Autònoma de Barcelona, Barcelona
[e] Università degli Studi di Cagliari, Cagliari

Abstract

This paper describes the process of creation of the first machine translation system from Italian to Sardinian, a Romance language spoken on the island of Sardinia in the Mediterranean. The project was carried out by a team of translators and computational linguists. The article focuses on the technology used (Rule-Based Machine Translation) and on some of the rules created, as well as on the orthographic model used for Sardinian.

1. Introduction

This paper presents a shallow-transfer rule-based machine translation (MT) system from Italian to Sardinian, two languages of the Romance group. Italian is spoken in Italy, although it is an official language in countries like the Republic of Switzerland, San Marino and Vatican City, and has approximately 58 million speakers, while Sardinian is spoken principally in Sardinia and has approximately one million speakers (Lewis, 2009).

The objective of the project was to make a system for creating almost-translated text that needs post-editing before being publishable. For translating between closely-related languages where one language is a majority language and the other a minority or marginalised language, this is relevant as MT of post-editing quality into a lesser-resourced language can help with creating more text in that language.

As described below, Sardinian is not a fully-standardised language. This means that linguistic resources are scarce, even if the orthographic norm chosen for this

 Corresponding author: francis.tyers@uit.no
Cite as: Francis M. Tyers, Hèctor Alòs i Font, Gianfranco Fronteddu, Adrià Martín-Mor. Rule-Based Machine Translation for the Italian–Sardinian Language Pair. The Prague Bulletin of Mathematical Linguistics No. 108, 2017, pp. 221–232. doi: 10.1515/pralin-2017-0022.

project was the Limba Sarda Comuna (*Common Sardinian Language*, or LSC), the one officially approved by the island's autonomous government in 2006. In fact, the main aim of the project was to create a tool that would foster text production in Sardinian, especially in areas such as administration and Wikipedia.

The remainder of the article is laid out as follows: In section 2 we provide some linguistic background to Sardinian. This is followed by a description of the platform used to build the MT system in section 3. Section 4 describes the development of the system, including resources that were reused. Then section 5 gives an evaluation of the system. Finally, we comment on possible future work in section 6 and give some conclusions in section 7.

2. Sardinian

The Sardinian language is a Romance language spoken by approximately one million people on the island of Sardinia, together with other Romance languages such as Tabarchino Ligurian (on the islands of San Pé and Sant'Antióccu), Algherese Catalan (in the city of L'Alguer), Sassarese (in the city of Sassari) and Gallurese Corsican (in Gaddùra).[1]

At the institutional level, some of these languages are recognised by the regional government. However, the use of Sardinian language is virtually non-existent at any educational level, as well as in many fields of the public sphere (media, newspapers, administration, etc.). Still, the use of Sardinian is widespread. According to (Oppo, 2007) only 2.7% per cent of the population in Sardinia does not have any competence (either active or passive) in "any local language".

Sardinian, classified as "definitely endangered" by UNESCO,[2] is spoken across most of the island despite the fact that, because of its great internal variety, two macro-varieties are often distinguished: northern (Logudorese and Nuorese) and southern (Campidanese). The existence of these two macro-varieties is one of the controversial factors when it comes to the standardisation of the language. At present, there are movements who advocate for different standardisation models and which, broadly, correspond to northern and southern regions.

On the one hand, there is a group that defends a double standard, following the Norwegian model. This model, which is basically followed in the south, has received endorsement by the provincial government of Casteddu, which has officially adopted a "southern" standard described in the document *Arrègulas po ortografia, fonètica, morfologia e fueddàriu de sa Norma Campidanesa de sa Lìngua Sarda* (Comitau Scientìficu po sa Norma Campidanesa de su Sardu Standard, 2009). On the other hand, the Limba Sarda Comuna (LSC) has been proposed as the standard form for all varieties of Sardinian. It is an evolved version of the Limba Sarda Unificada (LSU), which was in turn the result of an experts' committee called by the Sardinian government in 2001.

[1] Toponyms are written in the local languages. There are, apart from these, other linguistic islands which result from migrations, such as Venetian and Romanisku.

[2] `http://www.unesco.org/languages-atlas/en/atlasmap/language-id-337.html` and `www.unesco.org/culture/languages-atlas/en/atlasmap/language-id-381.html`

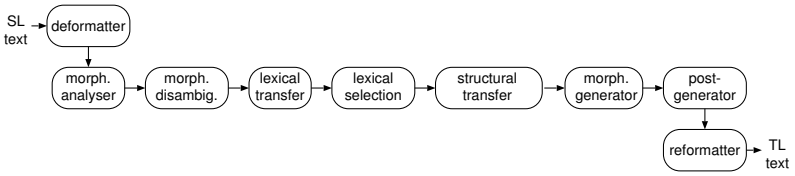


Figure 1. The modular architecture of the Apertium MT platform. Modules communicate using Unix text pipes.

In 2006, the Sardinian government adopted the LSC as a co-official language, alongside Italian, for the publication of official documents. The LSC is also the form chosen by several publishing houses and websites.

The existence of these two proposals implies that all initiatives concerning the Sardinian language must first take a stand on the issue of the standardisation model. The Sardinian Wikipedia, for instance, allows its users to mark the variety in which they write by adding a flag.

In October 2016, at the time of the writing of this article, the Sardinian Wikipedia has 5,230 content pages,[3] out of which 1,525 are written in Logudorese,[4] 776 in LSC,[5] and 295 in Campidanese.[6] Other digital products, such as Facebook (Beccu and Martín-Mor, 2017), Telegram (Martín-Mor, 2017) and Ubuntu,[7] have been partially localised into Sardinian basing mainly on the LSC model.

Indeed, according to Cheratzu (2015), textual and literary production in LSC is clearly greater in number than any other. Therefore, basing on textual production and resource availability, we decided to use LSC as the standard form of the Sardinian language in our project. Italian was chosen as the source language for our project. Despite the fact that linguistic resources (and competent writers) are scarce even for LSC, it was deemed appropriate, given the fragile situation of the Sardinian language, to facilitate the creation of contents in LSC from Italian (i.e., documents issued by the government, websites, newspapers, etc.).

3. Platform

The system is based on the Apertium machine translation platform (Forcada et al., 2011). The platform was originally aimed at the Romance languages of the Iberian peninsula, but has also been adapted for other, more distantly related, language pairs. The whole platform, both programs and data, are licensed under the Free Software

[3] https://sc.wikipedia.org/wiki/Ispetziale:Statistics
[4] https://sc.wikipedia.org/wiki/Categoria:Logudoresu
[5] https://sc.wikipedia.org/wiki/Categoria:Limba_Sarda_Comuna
[6] https://sc.wikipedia.org/wiki/Categoria:Campidanesu
[7] https://wiki.ubuntu.com/Ubuntu-Sardu/

Foundation's General Public Licence (GPL)[8] and all the software and data for the 43 supported language pairs (and the other pairs being worked on) is available for download from the project website.

3.1. Pipeline

A typical translator built with Apertium consists of 9 modules which communicate between each other using standard Unix pipes. This eases diagnosis, the insertion of new modules, etc. The modules comprise of the following:

- A **deformatter** which encapsulates any formatting (e.g. HTML or XML tags etc.) information in the input stream.
- A **morphological analyser** which for each surface form in the stream returns a sequence of possible analyses.
- A **part-of-speech tagger** which out of the possible analyses for a given word returns the most probable analysis. This is based on either first-order HMM or on HMM in combination with Constraint Grammar (Bick and Didriksen, 2015).
- A **lexical transfer** module which for each unambiguous source language lexical form returns one or more target language lexical forms.
- A **lexical selection** module which for each source language lexical form with more than one target language translation uses a set of rules operating on source-language context to choose the most adequate translation in the target language.
- A **structural transfer** module which performs syntactic and morphological operations to convert the source language intermediate representation into the target language intermediate representation. Common operations include insertion, deletion and substitution of lexical units, agreement between lexical units for e.g. gender, number and case, etc. The structural transfer module calls the lexical transfer module.
- A **morphological generator** which for each target language lexical form returns a surface (inflected) form.
- A **postgenerator** which performs orthographic operations, for example elision (such as *da+il=dal* in Italian).
- A **reformatter** which de-encapsulates any formatting, leaving it untouched.

Figure 1 gives an example pipeline. The data used by these modules are by and large specified in XML files and compiled into binary forms for use by the modules.

4. Development

The development of the Italian-Sardinian pair owes a lot to previous work on other language pairs. In this case, most of the lexical and morphological resources for Italian were taken from the Italian–Catalan pair (Toral et al., 2011), while part of the lexical and morphological resources for Sardinian was taken from the Sardinian–Catalan pair existing a prototype in the Apertium project. In parallel to our development of the

[8] https://www.gnu.org/licenses/gpl-3.0.en.html

Italian–Sardinian pair, developers from Prompsit Language Engineering were working on an Italian–Spanish pair, so we cooperated in the improvement of the resources for Italian.

4.1. Analysis

The development began with an analysis oriented to:

- collecting free linguistic resources for the dictionaries;
- collecting monolingual and bilingual corpora;
- systematically comparing the source and the target languages in order to understand what structural changes exist between them.

The contrastive analysis between Italian and Sardinian led to more than one hundred examples of translations the translator was expected to give, but a morpheme-by-morpheme translation would not, e.g.

- Nella mia terra. → In sa terra mea. (“In my land”)
- Bellissimi. → Bellos a beru. (“Very beautiful’)
- Darmi. → Mi dare. (“To give me”)

These observed differences were used in creating the transfer rules.

4.2. Morphological dictionaries

The Italian morphological dictionary is, for the most part, the one used in the Italian–Catalan translator. However, some work has been done to extend and fix verbal paradigms. In addition, some 2,000 lemmas were added from the free/open-source resource *Morph-it* (Zanchetta and Baroni, 2005).

A first version of the Sardinian morphological dictionary already existed. It was based on the “experimental” norms of LSC (Regione Autonoma della Sardegna, 2006). It was augmented with data from the spell checker provided by the regional government of Sardinia.[9]

An important lack of proper nouns in the spell checker was detected, so we partially solved it adding a few hundreds of the most common person and family names in Sardinia, as well as the names of all Sardinian municipalities and Italian regions. It is worth adding that many place names are not yet standardised, e.g. the names of the countries and capitals. We added a few of the most common.

4.3. Morphological disambiguation

Romance languages have a fair amount of morphological ambiguities. Fortunately for developers of rule-based machine translation systems between these languages, they share most ambiguities, so most of the time selecting the wrong morphological analysis does not imply a bad translation, a *free ride*. For instance, this is generally the case for words finishing in -ista (like *comunista*, ‘communist’) that may be both adjectives or nouns. Since this ambiguity happens to be in both the source and the

[9] http://www.sardegnacultura.it/cds/cros/

Dictionary	Entries
Sardinian	51,743
Italian	35,099
Sardinian–Italian	25,484

Table 1. Dictionaries in the MT system. The final translator is assembled as the intersection of the entries in these dictionaries.

target language, e.g. a wrong analysis of *comunista* as a noun in *il partito comunista* would still give a good translation as *su partidu comunista*.

Probably the most frequent ambiguity in Italian, which is shared by French, Spanish and Catalan too, is *la* that can be both a definite article (feminine *the*) or a pronoun (*her*). In Sardinian these two analyses have different forms so it was necessary to resolve the ambiguity.

In addition to training the tagger on a corpus of 17,000 words from TED talks and Wikinews,[10] we added a set of 30 rules using rules written using Constraint Grammar (CG) (Bick and Didriksen, 2015). CG rules for Italian mainly deal with the disambiguation between imperative verbal forms with enclitic pronouns and adjectives (e.g. *centrali* as 'central', masculine plural, or 'centre them'), and contractions of prepositions and determiners (e.g. *dalle* as 'from the.F.PL' or 'give.IMP.2.SG them'; *dai*, 'from the.M.PL', 'give.IMP.2.SG' or 'give.PRI.2.SG'; *dei*, 'of the.M.PL' or 'gods').[11]

Not every morphological ambiguity can be easily solved. A clear case is *sono*, which can be "I am" or "they are". This ambiguity does not exist in Sardinian: "I am" is *so*, while "they are" is *sunt*. Both Italian and Sardinian are pro-drop languages, the subject pronoun can be omitted since it can be almost always inferred from the context (especially from the verb form). So it happens that we often have to guess whether it is about "I" or "they" when dealing with *sono*. By default we assume third person based on our target domain of encyclopaedic texts.

4.4. Transfer lexicon

The transfer lexicon was one of the tasks of the project that has taken longer because of the lack of free bilingual dictionary. In total 25,484 lemmas have been added to the bilingual dictionary, about a half of them by hand using frequency lists of words. Most of the time Antonino Rubattu's *Universal Dictionary Italian-Sardinian* and Mario Casu's *Logudorese-Italian vocabulary* were consulted. However, when using the dictionaries we made efforts to choose a form which was also found in the LSC spell checker.

[10] Corpus provided by Prompsit Language Engineering, `http://www.prompsit.com`

[11] = masculine, F = feminine, SG = singular, PL = plural, IMP = imperative, PRI = present of indicative, 2 = second person.

4.5. Lexical selection

Because of the short time in which the translator was developed only 35 lexical selection rules have been added. The lack of bilingual corpora did not allow us to automatically infer any rules. For instance, a difficult case is the word *corso*, which may be both "street" and "Corsican". Both meanings are found often in similar contexts and have different translations in Sardinian. Rules define that, if the noun is found in plural or is preceded by the preposition "in", "Corsican" is preferred, otherwise "street" is chosen.

4.6. Structural transfer rules

Apertium, as a rule, translates lemmas and morphemes one by one. Obviously, this does not always work, even for closely related languages. Structural transfer rules are responsible for modifying morphology or word order in order to produce "adequate" target language. In all, we have defined 89 such transfer rules.

4.6.1. Noun-phrase internal agreement

Most of the rules deal with noun-phrase internal agreement both in gender and number. Two situations have to be distinguished. On one hand, the target language has combinations of gender and/or number that do not exist in the source language. About 8% of the nouns have been labelled in the bilingual dictionary as requiring that the gender or the number needs to be determined when translating from Italian into Sardinian. In this case, the actual gender and/or number is obtained from other words in the noun phrase.

On the other hand, a noun in the target language may have a gender and/or a number different than in the source one. This is the case for 7% of the nouns in the bilingual dictionary. In this case, the gender and/or the number of the other words of the noun phrase must be modified to agree with the name.

4.6.2. Possessives

Possessives also require a correct delimitation of noun phrases since they must be moved from its beginning to the end (1).

(1) La sua apparente indifferenza .
S' aparente indiferèntzia sua .
"His apparent indifference."

4.6.3. Tenses

Tenses in Sardinian tend to be often analytical. A number of tenses which are synthetic in Italian, as well in most of the Romance languages, are conjugated in Sardinian

by means of verbal periphrasis, e.g. the future (2a) and conditional (2b) and historical. In addition, LSC does not have the absolute past tense of Italian, and uses the present perfect (2c).

(2)	a.	Canterò	b.	Canterei	c. Cantai
		Apo a cantare		Dia cantare	Aia cantadu
		"I will sing"		"I would sing"	"I sang"

All these transformations have been done by means of specific transfer rules.

4.6.4. Clitic pronouns

In Italian clitic pronouns must be placed after the verbs in infinitive, imperative and gerund forms, as well as with past participles when used as past gerunds. Instead, in Sardinian in infinitive forms clitics should be placed before the verb. As a result, for instance *cantarla* ("to sing it") must be translated as *la cantare*.

4.6.5. Change of the auxiliary verb

In Italian the present continuous construction uses the auxiliary *stare*, while in Sardinian the auxiliary *èssere* is used instead of *istare* (3).

(3) Io sto studiando.
Deo so istudiende.
"I am studying."

4.7. Post-generation rules

After the generation of the raw version of the translation some additional processing has to be done. In most of the cases, this means to apostrophise. For instance, *l'accumulazione* ("the accumulation") is translated first of all as *sa acumulatzione*, where a special symbol is produced by the morphological generator, warning that the word *sa* is liable to receive modifications. A set of rules define in which case words in Sardinian are apostrophised. In the same way, the Sardinian words *no* and *ne* ("no" and "nor") may be changed to *non* and *nen* according to the context.

5. Evaluation

The system has been evaluated in two ways. The first is its coverage.[12] The second is the error rate of two pieces of text produced when comparing with a post-edited version of them.

[12]Here coverage is defined as naïve coverage, that is for any given surface form at least one analysis is returned. This may not be complete.

Corpus	Tokens	Coverage (%)
Wikipedia 10%	34,736,257	89.3
UD Italian	285,199	96.4

Table 2. Naïve vocabulary coverage. This is the percentage of tokens which receive at least one analysis from the morphological analyser. The coverage of Wikipedia is lower due to the large number of proper nouns and foreign words.

Words	Unknown words	WER	TER
2,033	9.4%	9.9%	6.3%

Table 3. Word Error Rate and unknown words over the 2,033 word test corpus.

5.1. Coverage

Table 2 presents the lexical coverage of the system over two corpora. The first was a subset of the Italian Wikipedia, which was created by randomly selecting 10% of the sentences from the Italian Wikipedia as of May 2016. The second corpus is the text from the Italian treebank in the Universal Dependencies project.[13]

5.2. Translation quality

We measured translation quality using two metrics: Word error rate (WER), which is based on the Levenshtein distance (Levenshtein, 1966) and was calculated for using the `apertium-eval-translator` tool; and Translation Error Rate (TER, Snover et al. (2006)). Metrics based on word error rate have been chosen for a number of reasons. Firstly we would like to be to compare the system against systems based on similar technology, and to assess the usefulness of the system in a real setting, that is of translating for **dissemination**. Secondly, the reference translation is a postedition, whereas most MT evaluation metrics use pre-translated references. Using a more commonly used metric in an uncommon setting would give deceptively good results.

A corpus of 2,033 words (53 sentences) was extracted from Wikipedia. The average length of a sentence was 42 words. This was the first paragraphs of the last two texts put in the section "vetrina" ("showcase") at the time of the GSoC final evaluation (more or less 1000 words per text). Wikipedia texts were selected, as this is one of the major uses for Apertium translators, especially as they are used by the Wikimedia Content Translation Tool.[14] The section "vetrina" is a pseudo-random selection (not done by the machine translator developers) of quality Wikipedia articles.

[13] `http://universaldependencies.org`

[14] `https://www.mediawiki.org/wiki/Content_translation`

The vast majority of unknown words are proper names (foreign person, family and place names) as well as foreign words (e.g. in French or English).

The scores are similar to or slightly better than those for other translators in the Apertium platform for Romance languages, for example the Catalan–-Occitan system achieves a WER of 9.6% (Armentano-Oller and Forcada, 2006) and the Spanish–Aragonese 16.8%, (Martínez Cortés et al., 2012).

5.3. Qualitative evaluation

Along with the quantitative evaluation of post-edition effort, we also performed a qualitative evaluation to determine where the system can be improved. Based on the final evaluation text, we have detected two major issues: 1) incorrect disambiguation of the verb *avere*; and 2) the absolute past tense transfer rule. In the examples that follow, the Italian phrase is presented on the first line, followed by the current translation into Sardinian produced by the system on the second, the correct translation on the third, and an English translation on the fourth.

5.3.1. Incorrect disambiguation of "avere"

The Italian verb *avere* ("to have") may be both an auxiliary and a lexical verb. These have different translations in Sardinian (4). The distinction between both verbs *avere* is done in the tagger. Nevertheless, it happens that when the auxiliary is separated from the participle by an adverb, *avere* is wrongly tagged as a lexical verb (5).

(4) a. Ho cantato.
Apo cantadu.
"I have sung."

b. Ho un gatto.
Tèngio unu gatu .
"I have a cat."

(5) Non aver adeguatamente protetto la Francia.
* Non tènnere in manera adeguada amparadu sa Frantza.
Non àere in manera adeguada amparadu sa Frantza.
"Not having adequately protected France".

This issue has to be solved in the morphological disambiguation step, for example using CG rules.

5.3.2. Absolute past tense

As seen before, an absolute past tense exists in Italian, but not in LSC, in which the present perfect is used instead. A transfer rule constructs the past perfect adding the Sardinian auxiliary verb *àere* ("to have") with the same person and number as the Italian verb and the past participle of the Sardinian translation of the lemma. Nevertheless, in Sardinian, as well as in Italian, several verbs are conjugated with the

auxiliary verb "to be", particularly the verbs of movement and the verb "to be" itself. The current transfer rule is too simple and does not take into account this fact 6a, so needs to be improved.

(6)

a. Sfuggì.
* Aiat isfugidu.
Fiat isfugidu.
"He escaped."

b. Fu.
* Aiat istadu.
Fiat istadu.
"He was."

6. Future work

Aside from fixing the problems outlined in section 5.3, we would also like to see more translation systems for Sardinian. We have an experimental system for Sardinian–Catalan which is particularly relevant as Catalan is one of the larger languages in direct contact with Sardinian. We are also interested in working on Corsican as it is also spoken in Sardinia.

7. Conclusions

We have presented the first ever MT system from Italian to Sardinian. The performance is similar to other translators created using the same technology. It translates texts sufficiently well for post edition, although there remains a lot of work to do with respect to improving lexical coverage, and some work to do on improving the disambiguation and transfer rules. The system is available as free/open-source software under the GNU GPL and the may be downloaded from Apertium SVN.[15]

Acknowledgements

We would like to thank Mikel Forcada for his constant encouragement and comments on an earlier version of this manuscript. Thanks also go out to Diegu Corràine for clarifications on the LSC standard. The project was partially funded by a stipend from the Google Summer of Code. We would also like to thank Gema Ramírez Sánchez, and the anonymous reviewers.

Bibliography

Armentano-Oller, Carme and Mikel L. Forcada. Open-source machine translation between small languages: Catalan and Aranese Occitan. In *5th SALTMIL workshop on Minority Languages*, pages 51–54, 2006.

Beccu, A. and A. Martín-Mor. Sa localizatzione de Facebook in sardu. *Revista Tradumàtica*, 14, 2017.

[15] http://www.apertium.org

Bick, Eckhard and Tino Didriksen. CG-3 – Beyond Classical Constraint Grammar. In *Proceedings of the 20th Nordic Conference of Computational Linguistics, NODALIDA 2015, May 11-13, 2015, Vilnius, Lithuania*, pages 31–39. Linköping University Electronic Press, Linköpings universitet, 2015.

Cheratzu, Francesco. Sa Chirca. In Mura, Riccardo and Maurizio Virdis, editors, *Caratteri e strutture fonetiche, fonologiche e prosodiche della lingua sarda. Il sintetizzatore vocale SINTESA*. 2015.

Comitau Scientìficu po sa Norma Campidanesa de su Sardu Standard. Arrègulas po ortografia, fonètica, morfologia e fueddàriu de sa Norma Campidanesa de sa Lìngua Sarda, 2009.

Forcada, M. L., M. Ginestí-Rosell, J. Nordfalk, J. O'Regan, S. Ortiz-Rojas, J. A. Pérez-Ortiz, F. Sánchez-Martínez, G. Ramírez-Sánchez, and F. M. Tyers. Apertium: a free/open-source platform for rule-based machine translation. *Machine Translation*, 25(2):127–144, 2011.

Levenshtein, Vladimir I. Binary codes capable of correcting deletions, insertions, and reversals. *Soviet Physics Doklady*, 10(8):707–710, 1966.

Lewis, M. Paul, editor. *Ethnologue: Languages of the World*. SIL International, Dallas, TX, USA, sixteenth edition, 2009.

Martín-Mor, A. La localització de l'apli de missatgeria Telegram al sard: l'experiència de Sardware i una aplicació docent. *Revista Tradumàtica*, 14, 2017.

Martínez Cortés, Juan Pablo, Jim O'Regan, and Francis Tyers. Free/Open Source Shallow-Transfer Based Machine Translation for Spanish and Aragonese. In *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12)*, Istanbul, Turkey, 2012. European Language Resources Association (ELRA).

Oppo, Anna. Conoscere e parlare le lingue locali. In Oppo, Anna, editor, *Le lingue dei sardi: una ricerca sociolinguistica*, chapter 1, pages 6–45. Regione Autonoma della Sardegna, 2007.

Regione Autonoma della Sardegna. Limba Sarda Comune. Norme linguistiche di riferimento a carattere sperimentale per la lingua scritta dell'Amministrazione regionale, 2006. URL `http://www.regione.sardegna.it/documenti/1_72_20060418160308.pdf`.

Snover, Matthew, Bonnie Dorr, Richard Schwartz, Linnea Micciulla, and John Makhoul. A Study of Translation Edit Rate with Targeted Human Annotation. In *Proceedings of the Conference of the Association for Machine Translation in the Americas*, 2006.

Toral, Antonio, Mireia Ginestí-Rosell, and Francis M. Tyers. An Italian to Catalan RBMT system reusing data from existing language pairs. In Sanchez-Martínez, F. and J.A. Perez-Ortiz, editors, *Proceedings of the Second International Workshop on Free/Open-Source Rule-Based Machine Translation*, pages 77–81, 2011.

Zanchetta, Eros and Marco Baroni. Morph-it! A free corpus-based morphological resource for the Italian language. *Corpus Linguistics 2005*, 1(1), 2005. ISSN 1747-9398.

Address for correspondence:
Francis M. Tyers
`francis.tyers@uit.no`
Giela ja kultuvvra instituhta
UiT Norgga árktalaš universitehta,
N-9018 Romsa,
Norway

PBML

The Prague Bulletin of Mathematical Linguistics
NUMBER 108 JUNE 2017 233–244

Continuous Learning from Human Post-Edits for Neural Machine Translation

Marco Turchi,[a] Matteo Negri,[a] M. Amin Farajian,[ab] Marcello Federico[a]

[a] Fondazione Bruno Kessler, Trento, Italy
[b] University of Trento, Italy

Abstract

Improving machine translation (MT) by learning from human post-edits is a powerful solution that is still unexplored in the neural machine translation (NMT) framework. Also in this scenario, effective techniques for the continuous tuning of an existing model to a stream of manual corrections would have several advantages over current batch methods. First, they would make it possible to adapt systems at run time to new users/domains; second, this would happen at a lower computational cost compared to NMT retraining from scratch or in batch mode. To attack the problem, we explore several online learning strategies to stepwise fine-tune an existing model to the incoming post-edits. Our evaluation on data from two language pairs and different target domains shows significant improvements over the use of static models.

1. Introduction

In the last couple of years, after more than a decade of supremacy in shared evaluation campaigns like WMT (Bojar, 2016) and IWSLT (Cettolo et al., 2015), phrase-based SMT approaches have been significantly outperformed by neural solutions. However, despite the impressive progress of the so-called encoder-decoder NMT architectures (Bahdanau et al., 2014), MT is still far from being a solved problem. Together with the high computational training costs, one of the downsides of NMT is that performance can be significantly affected by situations in which training and testing are performed on heterogeneous data (*e.g.* coming from different domains or featuring different vocabulary and sentence structure). In this challenging condition, the availability of

Corresponding author: turchi@fbk.eu
Cite as: Marco Turchi, Matteo Negri, M. Amin Farajian, Marcello Federico. Continuous Learning from Human Post-Edits for Neural Machine Translation. The Prague Bulletin of Mathematical Linguistics No. 108, 2017, pp. 233–244.
doi: 10.1515/pralin-2017-0023.

task-specific (*e.g.* in-domain) data makes it possible to reduce the performance drops by means of fine-tuning procedures that are much faster than full model retraining.

Fine-tuning is usually applied in "batch" conditions, in which a general out-of-domain NMT model is further trained on in-domain data *before testing*. This paper investigates its adoption at the core of an "online" learning approach in which, *at test stage*, the NMT model is continuously adapted to a stream of incoming parallel sentence pairs. This scenario is relevant for the so-called computer-assisted translation (CAT) framework, which now represents the standard operating environment in the translation industry. Given a source sentence to translate (*src*), translators working with a CAT tool operate on machine-derived suggestions (*tgt*) correcting them, when necessary, into post-edited (*pe*) translations of the desired level of quality. Even if not perfect, MT suggestions normally require less post-editing effort compared to manual translation from scratch. In this "translation as post-editing" process, new data in the form of *(src, tgt, pe)* triples are continuously generated, thus providing a wealth of material to tune and adapt existing NMT models to specific users and domains.

The exploitation of human post-edits in a continuous learning NMT framework represents an ideal scenario for deploying online learning techniques. In machine learning, online learning is defined as the task of using data that becomes available in a sequential order to stepwise update a predictor for future data. The new points used for the update often consist in labeled instances provided as external feedback (*i.e.* a "true" label representing the expected response for each given input). At each step, the difference between a prediction $p(x_i)$ and the corresponding true label $\hat{p}(x_i)$ is used by the learner to refine the next prediction $p(x_{i+1})$. In this way, a general model can evolve over time by integrating external feedback in order to reduce the distance between its predictions and the expected output. Such evolution can result in a general performance improvement but also, depending on the working scenario, in an adaptation to the specificities of the target application domain.

Cast as an online learning problem, our task consists in leveraging a stream of human post-edited data for continuous NMT adaptation. Along this direction, our contributions can be summarized as follows: (1) we define and explore for the first time an application-oriented framework for continuous NMT adaptation from human feedback, which is suitable for deployment in the CAT framework; (2) we propose different strategies to approach the problem; (3) we evaluate them in two different scenarios (different target languages, domains and levels of training/test data mismatch).

2. Related work

Previous work on online MT adaptation is motivated by the problem of performance degradation when training and testing on heterogeneous data. In phrase-based MT, this problem has been widely explored. The proposed solutions include the use of incremental expectation-maximization (EM) and suffix arrays to update the statistics of a generic model (Ortiz-Martínez et al., 2010, Ortiz-Martínez, 2016,

Germann, 2014), cache-based models (Bertoldi et al., 2013), discriminative approaches based on structured perceptrons (Wäschle et al., 2013), incremental Bayesian language models (Denkowski et al., 2014), and hierarchical methods (Wuebker et al., 2015).

In NMT, online methods have not been explored yet. In fact, adaptation approaches mostly rely on batch fine-tuning procedures that are carried out on a small corpus of "in-domain data".[1] To cope with training/test heterogeneity, fine-tuning consists in exploiting the availability of in-domain data representative of the test set to perform a focused additional training step (Luong and Manning, 2015). Despite some risk of overfitting to the small size of the in-domain data, this practice often results in significant performance gains. An interesting variant, closer to our approach, is proposed in (Li et al., 2016). It presents an on-the-fly local adaptation method which, for each incoming test sentence, performs fine-tuning on source-reference pairs extracted from the parallel corpus used to train the general model. This solution, however, does not take into account human feedback (post-edits), as the retrieval step is carried out on a static pool of parallel training data. In contrast, in our online scenario we explore different strategies for continuous NMT model update by fine-tuning on a dynamic pool that incorporates a stream of human post-edited data from a given (possibly new) domain. Different from (Li et al., 2016), moreover, our retrieval step is based on faster and more powerful information retrieval techniques (ngram-based search with Lucene), which reward longer matches of relevant terms (as opposed to Levenshtein distance, and the other similarity methods proposed in (Li et al., 2016), which treat all the matching terms equally).

Finally, among the strategies explored in this paper, we also consider the case in which the general model evolves over time (*i.e.* the updated model for sentence *n* becomes the starting model for sentence *n+1*). In (Li et al., 2016), instead, the original model is always restored before processing each incoming sentence.

3. Integrating User Feedback

In order to exploit human feedback for continuous NMT model update, we explore three possible strategies, in which post-edited data are respectively used: *i)* for global model improvements after translating an input sentence (§ 3.1), *ii)* as additional knowledge for local improvements before translating the input sentence (§ 3.2), or *iii)* for both global and local improvements (§ 3.3).

3.1. Adaptation "a posteriori"

This strategy makes a direct use of user feedback for updating a general model as in any standard online MT framework, that is by using human feedback in the form

[1] With the expression "in-domain" we broadly refer to data that differ from those used for training the model. This mismatch can be due to an actual difference in terms of semantic domain, but also to other discrepancies in terms of style, vocabulary, sentence structure, etc.

of *(src,pe)* pairs to stepwise update the MT model *after translating* each segment. After receiving the human post-edit (*pe*) of the translation (*tgt*) of a given segment (*src*), the goal is to learn from the *(src,pe)* pair and induce the model to better translate the next input segment. This is done by performing a further fine-tuning step of the original model, which consists of one (or more) training iterations over the *(src,pe)* pair.

Overall, adaptation a posteriori is rather conservative since, at each step, the changes of the general model are induced only from a parallel sentence pair consisting of a source segment coming from the target domain and its human post-edit.

3.2. Adaptation "a priori"

This strategy, inspired by the approach of Li et al. (2016), makes an indirect use of user feedback. It relies on an update step to locally adapt the model to each incoming segment *before translating* it. Given an input sentence (*src*), parallel sentence pairs in which the source side is similar to *src* are retrieved from the data used to train the general model. The retrieved material is used to fine-tune the general model, which results in a local model that will be used to translate the input sentence. Although in the approach of Li et al. (2016) the starting general model is the same for each input sentence, nothing prevents to take advantage from new *(src,pe)* pairs as long as they become available. To this aim, instead of keeping fixed the pool of parallel data accessed for the local update, we experiment with a pool that is continuously populated with the previously collected *(src,pe)* instances. Differently from (Li et al., 2016), in which the data for local adaptation are retrieved by computing similarities based on Levenshtein distance, word embeddings and the NMT encoder's hidden states, we adopt standard information retrieval techniques. In particular, we use the Shingle[2] filter of Lucene (McCandless et al., 2010), which performs ngram-based searches that reward at the same time relevant and longer matches. In our experiments, for each query (*i.e.* input sentence), the top matching source sentence retrieved from the pool and the corresponding translation are used to perform the local fine-tuning step.

Though more focused compared to the previous approach, adaptation a priori is potentially more risky. Indeed, at each step, the local adaptation of the general model is based on similar (but not necessarily relevant/useful) sentence pairs.

3.3. Double adaptation

The two previous approaches can be combined in the general scheme depicted in Figure 1. In this case, given an input sentence (*src*), the general NMT model (GM1) is first adapted locally by performing a fine-tuning step on similar data retrieved from the parallel data pool (training pairs + previously collected (*src,pe*) pairs). Then, the resulting adapted model (LM1) is used to translate the sentence. After receiving the

[2] goo.gl/HzeSAI

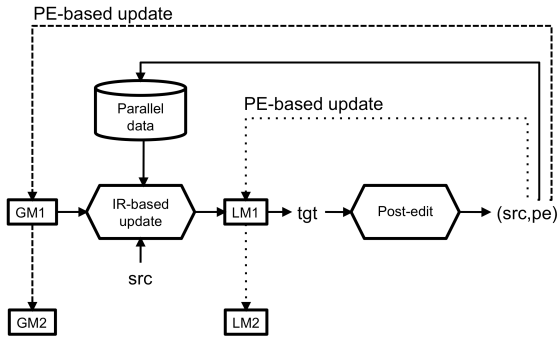


Figure 1. Double adaptation process.

human post-edit (*pe*) of the output translation (*tgt*), two options are possible. One is to exploit the (*src,pe*) pair to update the general model, which will be used as the starting model (GM2) for the next input segment. The second option is to exploit the (*src,pe*) pair to update the local model (LM1), which will be used as the starting model (LM2) for the next input segment. By adopting the first option, the translation process will rely on a chain of continuously evolving generic models (GM1, GM2, ..., GMn). By adopting the second option, the translation process will rely on a chain of models (GM1, LM1, LM2, ..., LMn) that, starting from the initial general model, evolves through local adaptations.

4. Experimental Setup

4.1. Approaches

Figure 2 illustrates the approaches compared in our experiments. The first one (a) is our baseline. It consists of a static NMT model (GM1), which is used to process the entire stream of data without changing over time (*i.e.* without learning from the *(src,pe)* sentence pairs obtained as human feedback). The second approach (b) is the adaptation "a posteriori" described in § 3.1, in which a general model is continuously fine-tuned to each incoming *(src,pe)* segment (GM1 for the first segment, GM2 for the second, and so on). The third and fourth approaches (c and d) represent the adaptation "a priori" described in § 3.2. In one case (c), for the first input segment to translate, a general model (GM1) is locally fine-tuned to similar sentence pairs retrieved from the pool of parallel data (recall that similarity is computed between the input segment and the source side of the instances in the pool). After translation, the same initial model (GM1) is used for the second input segment, and so on. In the other case (d), the locally-adapted model (LM1) is kept after translating the first input segment and used as starting model for the second one. The fifth and sixth so-

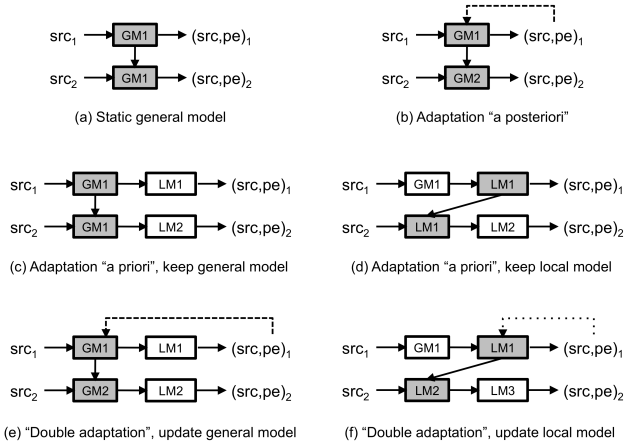


Figure 2. Static (a) vs online (b, c, d, e, f) NMT approaches.

lutions (e and f) represent the double adaptation method described in § 3.3. In (e), the general model (GM1) is locally fine-tuned (LM1) to translate the first input segment. Then, after translation and post-editing, human feedback is used to fine-tune again the general model, which will become the starting model (GM2) for the second input segment, and so on. In (f), the second fine-tuning step is applied to the local model.

4.2. Data

Our evaluation is carried out on two different language pairs and domains. The first scenario consists in translating information technology (IT) English sentences into German using a large set of heterogeneous parallel data to train the NMT system. In the second scenario, the NMT is trained on a small quantity of domain-specific data and it is used to translate medical English segments into Latvian. The two conditions pose different challenges. In the En_De setting, the initial NMT model is trained on a large general dataset that scarcely represents the target domain. *Hence, continuous learning mainly acts as a domain-adaptation process.* For En_Lv, the model is trained on domain-specific data, but in limited quantity. *Hence, the goal is to improve the overall translation quality by leveraging the new incoming data.* Regarding the target languages, Latvian is a Baltic language that is much more inflected that German. This results in a more sparse vocabulary that can affect translation performance.

For training the En_De NMT system, we merged the Europarl v7 (Koehn, 2005) and Common Crawl datasets released for the translation task at the 2016 Workshop on Statistical Machine Translation (WMT'16 (Bojar, 2016)) and random sampled 3.5 million sentence pairs. As domain-specific test set, we randomly selected 3k instances

from the training data released for the automatic post-editing task at WMT'16 (ibid.). This dataset consists of 12k *(src,tgt,pe)* triples, in which the source sentences come from an IT manual and the post-edits are generated by professional translators. In our experiments, the source sentences are translated by our NMT system and we assume that the existing post-edits are corrections of the NMT output.

For training the En_Lv NMT system, we used a subset of the EMEA parallel corpus proposed in (Pinnis et al., 2016). The test set is obtained by extracting 3k consecutive segments from randomly selected EMEA documents. Post-edits were generated by professional translators who corrected the output of our NMT system. Some data statistics are reported in Table 1.

	En_De			En_Lv		
	train	dev	test	train	dev	test
Number of sentence pairs	3.5M	2K	3K	385K	2K	3K
Source language tokens	63M	18K	50K	60.5M	20K	54K
Target language tokens	7.5M	37K	55K	6.8M	34K	50K

Table 1. Statistics about the En_De and En_Lv training, dev and test corpora.

4.3. NMT System

All the experiments are conducted with an in-house developed and maintained branch of the Nematus toolkit,[3] which is an implementation of the attentional encoder-decoder architecture (Bahdanau et al., 2014). Models were trained by splitting words into sub-word units using byte pair encoding (BPE), which Sennrich et al. (2016) indicates as an effective way to handle large vocabularies (*e.g.* to deal with rare words and highly inflected languages). Word segmentation was carried out by combining the source and target side of the training set and setting the number of merge rules to 40,000 for both language pairs. We used mini-batches of size 100, word embeddings of size 1024, and hidden layers of size 1024. The maximum sentence length was set to 50. The models were trained using Adam (Kingma and Ba, 2015) with an initial learning rate of 0.001, reshuffling the training corpora at each epoch. In both language pairs, the training of the generic systems was stopped after 20 epochs. Dropout is disabled.

5. Impact of gradient descent optimization algorithms

Following Li et al. (2016), in all the scenarios proposed in § 3 our NMT models are always updated using one single sentence pair. This is quite unusual for the NMT training common practice, in which batches containing dozens of sentence pairs are

[3] https://github.com/rsennrich/nematus

normally used. Leaving for future work the investigation on how to exploit larger sets of retrieved sentences, we run several experiments to measure the impact on performance of different optimization algorithms when using only one sentence pair.

The most used family of optimization algorithms is based on gradient descent, which is a way to minimize an objective function $J(\Theta)$, where $\Theta \in \mathbb{R}^d$, by updating the Θ parameters in the opposite direction of the gradient of the objective function $\nabla_\Theta J(\Theta)$. The learning rate η determines the size of the steps we take to reach a (local) minimum. Among the several optimizers proposed in literature, in our experiments we test: *i)* stochastic gradient descent (Sgd) (Bottou, 2010); *ii)* Adagrad (Duchi et al., 2011), *iii)* Adadelta (Zeiler, 2012) and Adam (Kingma and Ba, 2015). The main differences between these methods lie on the use of gradient information from the past time steps and on the way learning rates are updated. Sgd performs a parameter update for each training example ignoring the past gradient information and using a fix η chosen a priori. Differently from Sgd, Adagrad adapts the learning rate to the parameters using the past information and by performing larger updates for infrequent parameters and smaller updates for the frequent ones. Adadelta extends Adagrad by restricting the window of accumulated past gradients to some fixed size in order to mitigate the problem of a too fast monotonic decrease of the learning rate, which rapidly gets close to zero when all past gradients are retained. Adam introduces a bias correction mechanism and a better handling of moment information to induce faster parameter variations in the right direction.

	adam	adagrad	adadelta	sgd 1	sgd 0.1	sgd 0.01	sgd 0.001
En_De	39.1	30.1	44.4	37.2	**50.2**	47.5	44.0
En_Lv	28.3	16.2	38.7	34.5	**47.9**	47.3	46.8

Table 2. Results (BLEU) of different optimization algorithms.

In this set of experiments, we only consider the "a posteriori" adaptation strategy, which uses reliable in-domain *(src,pe)* pairs (see Figure 2(b)). In contrast with "a priori" adaptation, which operates on similar but potentially noisy retrieved instances, we believe that reliable insights will more likely come from this setting. Several learning rates are tested for Sgd (*i.e.* 1, 0.1, 0.01 and 0.001), while the other optimizers are initialized with a learning rate of 0.01. The BLEU results for both language pairs are reported in Table 2. Sgd generally performs better than the other optimizers that result in significantly lower scores. Looking at the different values of the Sgd learning rate, the performance improves when a larger η is used. For both languages, this is valid up to η equal to 0.1 while, for larger values, also Sgd results in poor translations.

The superiority of Sgd in our scenario contrasts with the results achievable in NMT when learning from a batch of sentence pairs, which usually favor dynamic optimizers. Our explanation is that gradients computed on a batch are more stable and less

affected by differences between sentence pairs. For this reason, optimizers that can leverage past gradient information are usually more reliable. When working with only one sentence pair, segments from the same document may have different structure, words and length, which makes gradient information from the past potentially misleading and causing instability in the optimizer. Since this problem would likely be exacerbated when adapting to diverse and potentially noisy data in the "a priori" setting, Sgd seems to be a safer solution for our case. In the remainder of the paper, all the experiments are run using Sgd with learning rate of 0.1.

6. Analysis of continuous learning strategies

Table 3 reports the results of a comparison between the adaptation strategies discussed in §4.1 and two baselines that do not exploit human feedback. The first one (Static) is an NMT model that is kept unchanged during the processing of the entire test set. The second one ("a priori w/o PE") is our re-implementation of (Li et al., 2016), which locally adapts the original NMT model to each test sentence by finding the most similar instance in the training data. After each translation, the locally-updated model is replaced by the initial general model, which is used as a starting point for the next sentence. This approach resembles our "a priori – keep general model" adaptation strategy (method (c) in Figure2) with the exception that human post-edits are not added to the pool of data accessed by Lucene. The reported results are obtained by iterating for 1 and 5 epochs over each sentence pair during updating.

By comparing the BLEU scores of the static and our re-implementation of (Li et al., 2016), it becomes evident that simple local adaptation has a marginal impact on the results (even negative for En_De with 5 epochs, with a drop of ~2 BLEU points). Although this contrasts with the results of Li et al. (2016), what is interesting to note here (more than comparing similar approaches on different language directions and data) is the scarce contribution, in our testing conditions, of retrieving instances from the static pool of training data. More visible improvements are in fact yielded by the application of the "a priori with PE" strategy, which takes advantage of a data pool that constantly grows by integrating human post-edits. With 1 fine-tuning epoch, the new domain-specific information results in slight improvements, on both language pairs, both over the baseline and over the "a priori w/o PE" adaptation. The gain is small (and not significant) on En_Lv, probably due to the fact that the original NMT model is domain-specific, hence already adapted to the target domain. In this case, performance is almost identical either if we Keep the General model after processing each sentence (K.G., which corresponds to method (c) in Figure 2) or if we Keep the Local model (K.L., method (d)). For En_De, improvements are significant in both conditions, especially when keeping the local model (+1.7 for K.L. vs. +0.5 for K.G.). This suggests that, despite the risks inherent to the "a priori" strategy, which adapts the NMT model to the retrieved sentence pairs independently from their degree of similarity with the sentence to translate, the locally-adapted model can be useful also for

		Static	a priori w/o PE	a priori with PE		a post.	Double	
				K. G.	K. L.		U. G.	U. L.
En_De	1 epoch	42.7	42.8	43.3*	44.5*	**50.2***	50.0*	48.4*
	5 epochs		40.9†	41.7†	41.8†	**49.2***	47.9*	46.3*
En_Lv	1 epoch	46.8	46.8	46.9	47.0	**47.9***	47.8*	47.4*
	5 epochs		46.9	47.2	47.3*	**48.3***	48.0*	47.5*

*Table 3. Results of different adaptation strategies. * and † respectively indicate statistically significant improvements/degradations with respect to the static system. Significance tests are performed with paired bootstrap resampling (Koehn, 2004).*

the next incoming sentences. With 5 fine-tuning epochs, instead, we observe mixed results. On En_De, in which training and test are heterogeneous, local adaptation overfits to sentences that can feature low similarity with the input and is definitely harmful (both K.G. and K.L. results are significantly below the baseline). On En_Lv, for which training and test data are homogeneous, we observe slight improvements, which are significant when keeping the local model (+0.5 BLEU with K.L.).

The use of post-edits a posteriori ("a post." column) results in a significant improvement over the baseline on both language pairs (+7.5 for En_De and +1.1 for En_Lv). We interpret these coherent gains as an indication that continuous NMT adaptation to reliable domain-specific sentence pairs reinforces the model capability to translate the incoming sentences. Again, overfitting by running more epochs yields mixed results. On En_De (heterogeneous data) performance drops but is still significantly better compared to all previous methods, while on En_Lv (homogeneous data), more epochs yield the best result. The difference between "a priori" and "a posteriori" adaptation also emerges when combining them together ("Double" column). In general, updating the General model (U.G, method (e) in Figure 2) achieves better results than Updating the Local model (U.L. method (f)), though slightly worse than "a posteriori" adaptation.

7. Conclusion and Future Work

We addressed the problem of improving an existing NMT model by continuously learning from human feedback. As opposed to batch learning techniques, continuous learning from a stream of incoming post-edits represents a promising solution for cutting the costs of resource/time-demanding routines to periodically retrain NMT models from scratch. Moreover, it would make it possible to adapt systems' behavior to users and domains while the system is in use, thus making the improvements visible in a short time and reducing the human post-editing workload. To achieve these objectives we explored different strategies, in which an NMT model is fine tuned: *i)* a posteriori (*i.e.* after receiving the human post-edit of a translated sentence), *ii)* a priori (*i.e.* locally, before translation, by learning also from previous feedback), or *iii)*

both (*i.e.* before and after translation). We experimented in different settings, with two language combinations and two target domains, either homogeneous or diverse with respect to the data used to train the initial NMT model. Our best results reveal significant gains both over a static NMT model used as a baseline and over an adaptive solution (the most similar to our *a priori* adaptation strategy), which does not exploit human feedback. Several interesting aspects have not been discussed and deserve attention in future work. From the technical side, our initial exploration of the impact of using different parameter optimizers and running different numbers of fine-tuning epochs can be extended and complemented with the analysis of: *i)* alternative instance selection techniques (*e.g.* similarity thresholds applied to the retrieved data), *ii)* dynamic, instance-specific ways to set the learning rate and the number of epochs depending on the similarity of the retrieved material with respect to an input sentence, and *iii)* the impact of fine-tuning on more than one sentence at a time. From the application side, the evaluation with multiple target domains, possibly involving professional translators operating in a computer-assisted translation environment is the first step in our agenda.

Acknowledgements

This work has been partially supported by the EC-funded H2020 projects QT21 (grant no. 645452) and ModernMT (grant no. 645487).

Bibliography

Bahdanau, Dzmitry, Kyunghyun Cho, and Yoshua Bengio. "Neural Machine Translation by Jointly Learning to align and translate". *arXiv preprint arXiv:1409.0473*, 2014.

Bertoldi, Nicola, Mauro Cettolo, and Marcello Federico. Cache-based Online Adaptation for Machine Translation Enhanced Computer Assisted Translation. In *Proc. of the XIV Machine Translation Summit*, pages 35–42, Nice, France, September 2013.

Bojar, Ondřej et al. Findings of the 2016 Conference on Machine Translation. In *Proc. of the First Conference on Machine Translation*, pages 131–198, Berlin, Germany, August 2016.

Bottou, Léon. "Large-Scale Machine Learning with Stochastic Gradient Descent". In *Proc. of COMPSTAT'2010*, pages 177–187, Paris, France, August 2010. Springer.

Cettolo, Mauro, Jan Niehues, Sebastian Stüker, Luisa Bentivogli, Roldano Cattoni, and Marcello Federico. The IWSLT 2015 Evaluation Campaign. In *Proc. of the 12th International Workshop on Spoken Language Translation (IWSLT 2015)*, Da Nang, Vietnam, 2015.

Denkowski, Michael, Chris Dyer, and Alon Lavie. Learning from Post-Editing: Online Model Adaptation for Statistical Machine Translation. In *Proc. of the 14th Conference of the European Chapter of the Association for Computational Linguistics*, Gothenburg, Sweden, April 2014.

Duchi, John, Elad Hazan, and Yoram Singer. Adaptive Subgradient Methods for Online Learning and Stochastic Optimization. *Journal of Machine Learning Research*, 2011.

Germann, Ulrich. Dynamic Phrase Tables for Machine Translation in an Interactive Post-editing Scenario. In *Proc. of the Workshop on interactive and adaptive machine translation*, pages 20–31, Vancouver, BC, Canada, 2014.

Kingma, Diederik P. and Jimmy Ba. Adam: A Method for Stochastic Optimization. In *Proc. of the 3rd Int. Conference on Learning Representations*, pages 1–13, San Diego, USA, May 2015.

Koehn, Philipp. Statistical Significance Tests for Machine Translation Evaluation. In *Proceedings of the Empirical Methods on Natural Language Processing*, pages 388–395, 2004.

Koehn, Philipp. Europarl: A Parallel Corpus for Statistical Machine Translation. In *Proc. of the tenth Machine Translation Summit*, pages 79–86, Phuket, Thailand, 2005.

Li, Xiaoqing, Jiajun Zhang, and Chengqing Zong. "One Sentence One Model for Neural Machine Translation". *arXiv preprint arXiv:1609.06490*, 2016.

Luong, Minh-Thang and Christopher D. Manning. Mixture-Model Adaptation for SMT. In *Proc. of the 12th International Workshop on Spoken Language Translation*, pages 76–79, Da Nang, Vietnam, December 2015.

McCandless, Michael, Erik Hatcher, and Otis Gospodnetic. *Lucene in Action*. Manning Publications Co., Greenwich, CT, USA, 2010.

Ortiz-Martínez, Daniel. Online Learning for Statistical Machine Translation. *Computational Linguistics*, 42(1):121–161, 2016.

Ortiz-Martínez, Daniel, Ismael García-Varea, and Francisco Casacuberta. Online Learning for Interactive Statistical Machine Translation. In *Proc. of NAACL-HLT 2010*, pages 546–554, Los Angeles, California, June 2010.

Pinnis, Marcis, Rihards Kalnins, Raivis Skadins, and Inguna Skadina. What Can We Really Learn from Post-editing? In *Proc. of AMTA 2016 vol. 2: MT Users' Track*, pages 86–91, Austin, Texas, November 2016.

Sennrich, Rico, Barry Haddow, and Alexandra Birch. Neural Machine Translation of Rare Words with Subword Units. In *Proc. of the 54th Annual Meeting on Association for Computational Linguistics*, pages 1715—1725, Berlin, Germany, August 2016. Association for Computational Linguistics.

Wäschle, Katharina, Patrick Simianer, Nicola Bertoldi, Stefan Riezler, and Marcello Federico. Generative and Discriminative Methods for Online Adaptation in SMT. In *Proc. of Machine Translation Summit XIV*, pages 11–18, Nice, France, September 2013.

Wuebker, Joern, Spence Green, and John DeNero. Hierarchical Incremental Adaptation for Statistical Machine Translation. In *Proc. of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1059–1065, Lisbon, Portugal, September 2015.

Zeiler, Matthew D. "ADADELTA: An Adaptive Learning Rate Method". *arXiv preprint arXiv:1212.5701*, 2012.

Address for correspondence:
Marco Turchi
turchi@fbk.eu
Via Sommarive 18, Povo, 38123 Trento, Italy

PBML

The Prague Bulletin of Mathematical Linguistics
NUMBER 108 JUNE 2017 245–256

Applying N-gram Alignment Entropy to Improve Feature Decay Algorithms

Alberto Poncelas, Gideon Maillette de Buy Wenniger, Andy Way

ADAPT Centre, School of Computing,
Dublin City University, Dublin, Ireland

Abstract

Data Selection is a popular step in Machine Translation pipelines. Feature Decay Algorithms (FDA) is a technique for data selection that has shown a good performance in several tasks. FDA aims to maximize the coverage of *n*-grams in the test set. However, intuitively, more ambiguous n-grams require more training examples in order to adequately estimate their translation probabilities. This ambiguity can be measured by alignment entropy. In this paper we propose two methods for calculating the alignment entropies for *n*-grams of any size, which can be used for improving the performance of FDA. We evaluate the substitution of the n-gram-specific entropy values computed by these methods to the parameters of both the exponential and linear decay factor of FDA. The experiments conducted on German-to-English and Czech-to-English translation demonstrate that the use of alignment entropies can lead to an increase in the quality of the results of FDA.

1. Introduction

In recent years the amount of data available has increased significantly. Now it is possible to find vast amounts of data for use as training data in Machine Learning. The field of Statistical Machine Translation (SMT) is no exception to this phenomenon. However, as shown in Ozdowska and Way (2009), having more data does not always lead to better results. In contrast, the performance can increase by limiting the training data to a smaller but more relevant set. This is why the use of data selection techniques has become a common step in the creation of an MT pipeline.

 Corresponding author: alberto.poncelas@adaptcentre.ie
Cite as: Alberto Poncelas, Gideon Maillette de Buy Wenniger, Andy Way. Applying N-gram Alignment Entropy to Improve Feature Decay Algorithms. The Prague Bulletin of Mathematical Linguistics No. 108, 2017, pp. 245–256. doi: 10.1515/pralin-2017-0024.

The data selection technique we are using in this paper is Feature Decay Algorithms (FDA) (Biçici and Yuret, 2011; Biçici et al., 2015; Biçici and Yuret, 2015) which has obtained good results in several Workshops on both MT and quality estimation tasks. FDA collects a limited set of best sentence pairs for model training from a parallel training corpus using the (source-side) information of the test set. FDA first extracts features from the test set, and initializes them. Then, for every sentence selection iteration, FDA: 1) re-scores these features based on the already selected sentences and 2) selects the best sentence from the parallel corpus given the re-scored features, and adds it to the selected training data.

There have been previous attempts to improve FDA by using alignment entropies for unigram features (Poncelas et al., 2016). This makes sentences containing specific unigrams more (or less) likely to be selected and thus different numbers of occurrences of those unigrams are obtained in the final training data.

In this paper we propose two methods that can be used for calculating not only the alignment entropies of a unigram, but for any n-gram of any size. In addition we explore the performance of these methods when used to determine the value of different parameters in the mathematical model of FDA.

We perform experiments on German-to-English and Czech-to-English translation and show that it is possible to calculate a set of weights that can be used to extend FDA and obtain better results according to several evaluation metrics.

The remainder of the article is structured as follows. In Section 2 we give an outline of work that is closely related to this paper. In Section 3 we describe different extensions we propose to improve the performance of FDA. In Section 4 we describe the experiments we have designed and describe the data that has been used. In Section 5 we analyse the obtained results and perform a comparison for the different proposed extensions. We conclude in Section 6 and provide avenues for future work.

2. Related Work

The technique of data selection to be used is FDA. This is a method for selecting a subset from a set of parallel sentences to be used as training data for a Machine Translation System. This technique performs data selection by iteratively obtaining the most appropriate sentence pairs from a candidate pool and adding them to a selected pool, which ultimately becomes the training data when the process finishes.

2.1. Feature Decay Algorithms

FDA is a method that aims to maximize the coverage of n-grams in the test set. It does so by scoring each sentence during sentence selection as a weighted sum of the words, or more generally n-grams, which that particular sentence covers from the test set (the document we want to translate). Furthermore, the weight of previously selected n-grams is decreased in proportion to the number of times the n-gram has

already been included. This process is called feature decay. Once all the sentences have been scored, the one with the highest score will be transferred from the candidate pool and included in the selected pool. This process is iteratively repeated.

The values of the features of the selected sentence are decreased as in (1):

$$decay(f) = init(f)\frac{d^{C_L(f)}}{(1 + C_L(f))^c} \quad (1)$$

L is the selected pool, c is the linear decay factor, while d is the exponential decay factor.[1]

$C_L(f)$ is the count of the feature f in L, which makes the most frequent features decay faster, thereby allowing an increase in variability of *n*-grams in the training data. The initialization function is defined in (2):

$$init(f) = \log(|U|/C_U(f))^i |f|^l \quad (2)$$

where $|U|$ is the size of the training data, $C_U(f)$ is the count of the feature f in the training data and $|f|$ is the number of tokens of f.

2.2. Alignment Entropy of Unigram as Extension of FDA

FDA treats all *n*-grams equally, the default parameters of (2) are static. It does not distinguish according to how ambiguous the translation of an *n*-gram is. But intuitively, more ambiguous n-grams require more training examples in order to adequately estimate their translation probabilities. For example, proper names like "Smith" that can be unambiguously translated require fewer occurrences in a training set. Therefore the importance of this feature should decay faster than other words such as "for" or "at" which can have several possible translations.

A method for measuring how ambiguous the translations are for a given *n*-gram is to use alignment entropy. Entropy measures uncertainty, as defined in 3:

$$entropy(x) = -\sum_i p(x_i) * \log(p(x_i)) \quad (3)$$

The alignment entropy can be calculated by using the alignment probabilities in (3). These alignment probabilities can be retrieved from word-alignment models like FastAlign (Dyer et al., 2013) or GIZA++ (Och and Ney, 2003).

Let s be an *n*-gram in the source language and t an *n*-gram in the target language. We can define A_s as the set of *n*-grams in the target language that are potential trans-

[1]Strictly speaking, for c in the range $(0, 1)$, c, in the denominator of formula (1), adds decay that is sublinear in $C_L(f)$, while for c in the range $(1, \infty)$ it adds decay that increases faster than linear, though not exponential. However, in the experiments in this paper, c is in the range $(0, 1)$, so the effect the factor involving c is at most linear, so we just refer to it as "linear" for simplicity.

lations of s, and $p(s,t)$ be the probability of s being translated as t. Accordingly, the alignment entropy of s can be calculated as in (4):

$$\mathtt{alignEnt}(s) = \frac{\sum_{t \in A_s} p(s,t) * \log(p(s,t))}{\log(|A_s|)} \quad (4)$$

In order to have alignment entropies in the $[0,1]$ range, the entropies are divided by the the $\log$ of the number of possible translations, $\log(|A_s|)$, which is the maximum possible entropy.

The score obtained in (4) can be used in (1) as the value of one of the decay factors, d or c. As a result the alignment entropy can have an influence on the decay.

In (Poncelas et al., 2016) experiments were carried out using unigrams as features and changing the parameter d in (1). The alignment probabilities were obtained by using FastAlign and GIZA++, showing that probabilities calculated by GIZA++ achieved better results.

3. Computing and Applying Alignment Entropies

In this paper we propose two possible alternatives for estimating the alignment entropy of a any order *n*-gram. In addition, we want to explore the performance when extending the different decay factors.

3.1. Extending the Exponential and Linear Decay in FDA

In FDA, the decay function (1) has two parameters: the linear decay factor c in the range $[0,\infty)$ with a default value of 0.0, and exponential factor d, in range $(0,1]$ with a default value of 0.5. We are interested in exploring the impact in the performance when changing these values. The aim is to analyze the three possible combinations: change exponential decay exclusively, linear decay exclusively, and both the exponential and linear decay. Note that when changing both decay factors we are using the same set of weights in both parameters.

3.2. Computing 3-gram Alignment Entropy in FDA

While the unigram alignment entropy can be computed by using the conditional probabilities retrieved from FastAlign or GIZA++ (because they are already word-to-word translation probabilities), computing an *n*-gram alignment is not straightforward. It is not reasonable to expect that, for example, a 3-gram in the source language should be mapped to a 3-gram in the target language as well.

In order to estimate the alignment entropy for any size *n*-grams we propose the following two alternative entropy instantiations:

A *mean-of-unigram method*: Compute the alignment entropy of the unigrams using an alignment tool. For the words whose alignments could not be retrieved, we

assign them an entropy equal to the mean of the entropies of the rest of the words. Then we can estimate the entropy of the *n*-gram as the mean of the entropies of the words in the *n*-gram.

B *ngram-to-unigram method*: Assume that for every sentence pair $\langle l_s, l_t \rangle$, an *n*-grams s in the source sentence l_s is only aligned to a single word (unigram) chosen from the target sentence l_t with which it appears. Furthermore, assume all these alignments are equally likely. Then to compute the alignment entropy for s:

1) Extract from the parallel corpora the set L of line-pairs $\langle l_s, l_t \rangle$ that contain s in in the source side: $L = \{\langle l_s, l_t \rangle : s \in l_s\}$
2) Compute a multiset S_s of translation tuples containing s:
   For every line-pair $\langle l_s, l_t \rangle \in L$, for every word $w_t \in l_t$ extract an n-gram alignment tuple $\langle s, w_t \rangle$. (Assuming every words *w* in the target side is a potential translation candidate for s)
3) Compute the alignment probability distribution P_s from S_s using relative frequence estimation.
4) Finally, compute the entropy over the thus computed distritutions P_s.

We expect *n*-grams with lower entropies to be aligned to a lower variety of words on the target side. This provides us with an estimation of how difficult is to find a translation. In addition *n*-grams that tend to appear in in-domain contexts will have less translation candidates and therefore lower entropies. The probabilities calculated using this method can be used in (4) for computing the alignment entropy of the *n*-gram.

4. Experiments

The goal of the designed experiments is to test the effect on the performance of the different aligment entropies (explained in Section 3.2) used when changing different decay factors (explained in Section 3.1). We will refer to this modified factor as entropy-modified decay. Therefore, the designed experiments are the following:

- Baseline experiment: Execute FDA with the default values in the parameters.
- *mean-of-unigram* experiment: Use as alignment entropy H the mean of the alignment entropy retrieved by GIZA++ of its containing words (method A in the section 3.2). Substitute H for c (linear decay), d (exponential decay) or both.
- *ngram-to-unigram* experiment: Calculate the alignment entropy H as if the *n*-grams were aligned to a single word in the target side (method B in the section 3.2). Substitute H for c (linear decay), d (exponential decay) or both.

We are interested in observing the effect of these variants in different languages and using features of different sizes. Therefore each of these experiments were carried out using German (a language with a relatively strict word order), and Czech (a language with free word order) as source languages. In addition, we used FDA1

(using unigram as features) and FDA3 (features of up to 3-grams, which is what FDA computes by default).

The data sets used in the experiments are based on the ones used in the work of Biçici (2013) and Poncelas et al. (2016): (i) *Languages*: German-to-English and Czech-to-English; (ii) *Training data*: The training data provided in the WMT 2015 (Bojar et al., 2015) translation task setting a maximum sentence length of 126 words (4.5M sentence pairs, 225M words, in German-to-English corpus and 11M sentence pairs, 355M words, in Czech-to-English corpus); (iii) *Tuning data*: We use 5K randomly sampled sentences from development sets from previous years; (iv) *Language Model*: 8-gram Language Model (LM) built using the target-language side of the selected data via the KenLM toolkit (Heafield, 2011) using Kneser-Ney smoothing; (v) *Selected sentences*: Select 66.4 million words in total (source- and target-language sides) in each experiment; (vi) *Test set*: Documents provided in the WMT 2015 Translation Task.

We train SMT systems on the selected data using the Moses toolkit (Koehn et al., 2007) with the standard features and using GIZA++ for word alignment. We include several evaluation metrics: BLEU (Papineni et al., 2002), NIST (Doddington, 2002), TER (Snover et al., 2006), METEOR (Banerjee and Lavie, 2005) and CHRF (Popovic, 2015). These scores give an estimation of the quality of the output of the experiment when comparing to a translated reference. In general, the higher the score is, the better the translation is estimated to be (except for TER, which is a translation error measure and so lower is better).

5. Results

In Table 1 and Table 2 we present the mean of 4 MERT (Och, 2003) tuning executions for the different experiments. In the columns we show the baseline (FDA with default values, $\mathtt{d} = 0.5$ and $\mathtt{c} = 0.0$), exponential decay (FDA substituting the entropies for d and keeping $\mathtt{c} = 0.0$), linear decay (FDA substituting the entropies for c and keeping $\mathtt{d} = 0.5$) and linear and exponential decay (substituting the entropies for both c and d). In Table 1 and Table 2 we also compute statistical significance at level p=0.01 when compared with the baseline using Bootstrap Resampling (Koehn, 2004) for BLEU, TER and METEOR scores.

We can observe that choosing good alignment entropies combined with changing the proper decay factors can obtain better results than the default baseline. In this section we compare the performance of the extensions for FDA1 and FDA3, the comparison of changing different decay factors, and the comparison of the obtained alignment entropies.

5.1. Comparison of FDA1 with FDA3

Considering that the features extracted in FDA1 are a subset of the ones from FDA3 one would expect to have better results when using features of larger order *n*-grams.

	baseline		**entropy-modified exponential decay**		**entropy-modified linear decay**		**entropy-modified linear and exponential decay**	
	FDA1	FDA3	FDA1	FDA3	FDA1	FDA3	FDA1	FDA3
de → en								
BLEU	0.2285	0.2282	0.2170	0.2235	0.2276	**0.2307***	0.2198	0.2232
NIST	6.9407	6.9237	6.7984	6.8734	6.9124	**6.9573**	6.8345	6.8825
TER	0.5966	0.5955	0.6035	0.5982	0.5989	**0.5918***	0.6002	0.5981
METEOR	0.2864	0.2851	0.2804	0.2827	0.2842	**0.2859***	0.2819	0.2832
CHRF3	50.124	49.937	49.001	49.528	49.854	49.884	49.321	49.743
CHRF1	50.727	50.705	49.841	50.265	50.553	**50.836**	50.077	50.301
cs → en								
BLEU	0.2127	0.2184	0.2102	0.2146	0.2121	**0.2190**	0.2073	0.2137
NIST	6.6518	6.6983	6.6295	6.6375	6.6408	**6.7004**	6.5740	6.6247
TER	0.5973	0.5955	0.6221	0.6205	0.6202	0.6152	0.6252	0.6200
METEOR	0.2805	0.2827	**0.2815**	0.2806	0.2805	**0.2832**	0.2790	0.2796
CHRF3	48.178	48.578	48.078	48.316	48.029	**48.605**	47.647	48.160
CHRF1	49.250	49.604	49.145	49.245	49.201	49.589	48.822	49.161

Table 1. Results of the average of the scores after 4 tuning executions for the baseline, and mean-of-unigram *experiment. The results in bold indicate an improvement over the baseline. The asterisk means the result is statistically significant.*

However, we observe that it is not always the case. An example of this is the German-to-English translation for the default FDA. As we can see in the baseline column in Table 1 or Table 2 the results when using features of size 1 are better than those of size 3 for the BLEU, NIST, METEOR, CHFR3 and CHFR1 evaluation scores.

We also observe that the extensions proposed in this paper affect FDA3 and FDA1 differently. Extensions that improve an evaluation metric in FDA1 do not necessarily translate into improvements in FDA3. The METEOR score for Czech-to-English translation in Table 1 (entropy-modified exponential decay column) increases from 0.2805 (the baseline) to 0.2815, while the same evaluation score in FDA3 decreases from 0.2875 to 0.2806. The opposite is also true, not all the extensions yielding improvements with FDA3 do the same with FDA1.

5.2. Exponential Decay vs Linear Decay

Looking at Table 1 and Table 2, we observe that it is not necessarily preferable to change one decay factor over the other. Different sets of weights perform better

	baseline		**entropy-modified exponential decay**		**entropy-modified linear decay**		**entropy-modified linear and exponential decay**	
	FDA1	FDA3	FDA1	FDA3	FDA1	FDA3	FDA1	FDA3
de → en								
BLEU	0.2285	0.2282	0.2271	**0.2297**	0.2247	**0.2286**	0.2278	**0.2305***
NIST	6.9407	6.9237	6.9270	**6.9618**	6.9107	**6.9284**	6.9367	**6.9713**
TER	0.5973	0.5955	0.5997	0.5974	0.5982	0.5967	0.5982	0.5966
METEOR	0.2864	0.2851	0.2851	**0.2869***	0.2846	0.2849	0.2856	**0.2867***
CHRF3	50.124	49.937	50.075	**50.221**	49.957	49.771	50.070	**50.263**
CHRF1	50.727	50.705	50.640	**50.826**	50.517	50.679	50.721	**50.857**
cs → en								
BLEU	0.2127	0.2184	0.2088	**0.2202***	**0.2145***	**0.2197**	**0.2142***	**0.2211***
NIST	6.6518	6.6983	**6.5560**	**6.7224**	**6.6712**	**6.7136**	**6.6630**	**6.7447**
TER	0.6187	0.6154	0.6296	**0.6140**	**0.6182**	**0.6152**	**0.6184**	**0.6127***
METEOR	0.2805	0.2827	0.2799	**0.2844***	**0.2816***	**0.2832**	**0.2817***	**0.2851***
CHRF3	48.178	48.578	47.866	**48.768**	**48.293**	**48.666**	**48.365**	**48.827**
CHRF1	49.250	49.604	48.950	**49.736**	**49.344**	**49.630**	**49.392**	**49.850**

Table 2. Results of the average of the scores after 4 tuning executions for the baseline, and ngram-to-unigram *experiment. The results in bold indicate an improvement over the baseline. The asterisk means the result is statistically significant*

changing different decay factors. For example, in FDA3, the scores obtained in the *mean-of-unigram* experiment work better for most of the scores when changing the linear decay factor, while in *ngram-to-unigram* experiment changing the exponential decay performs better for almost every score.

In FDA1 the use of our novel extension is even more unclear, as the only statistically significant improvement occurs in Czech-to-English translation when changing the linear decay (BLEU and METEOR rows in Table 2).

5.3. Changing One Decay Factor vs Changing Both Decay Factors

In Section 5.2, we have concluded that the performance of the decay factor depends on the set of weights used as inputs. Note that in these experiments we change both factors with the same values, so we propose as future work a more fine-grained evaluation of the performance using different entropies in each decay factor. Despite the dependency on the weights, we find that, in FDA3, it is possible to find a set (Table

	de → en		cs → en	
	mean	std	mean	std
mean-of-unigram	0.6008	0.2035	0.5333	0.1926
ngram-to-unigram	0.7450	0.1244	0.7314	0.1310

Table 3. Mean and standard deviation of the aligment entropies distribution for FDA3.

2, last column) that can improve the baseline for almost every score[2], and it is the only extension in obtaining statistically significant improvement for more than one evaluation metric in both languages.

5.4. Comparison of *mean-of-unigram method* and *ngram-to-unigram method*

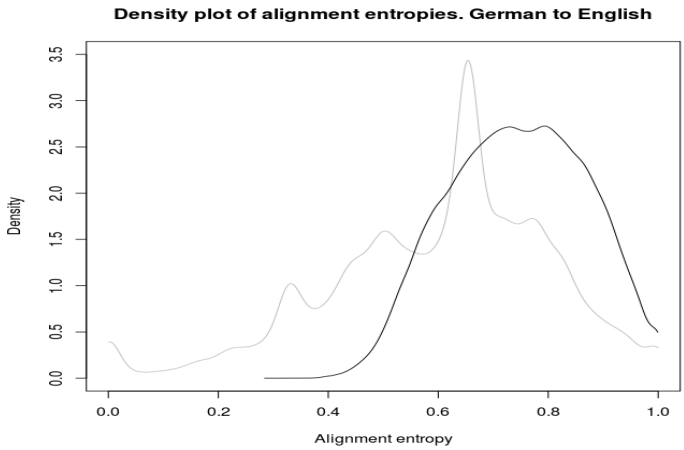


Figure 1. Density plot of the alignment entropies obtained in mean-of-unigram *(grey) and* ngram-to-unigram *(black) experiments for FDA3 and for German-to-English translation.*

In The *ngram-to-unigram* experiment we are assuming that every word in the target language may be a potential candidate translation for a given n-gram. Therefore we expect it to produce a set of higher entropies.

In order to have a deeper understanding of the distributions of the entropies in the experiments, in Figure 1 and Figure 2 we show the distribution for German-to-English and Czech-to-English translations, respectively. In Table 5.4 we also include

[2]The single case where the score is not improved, is the TER score for German-English translation.

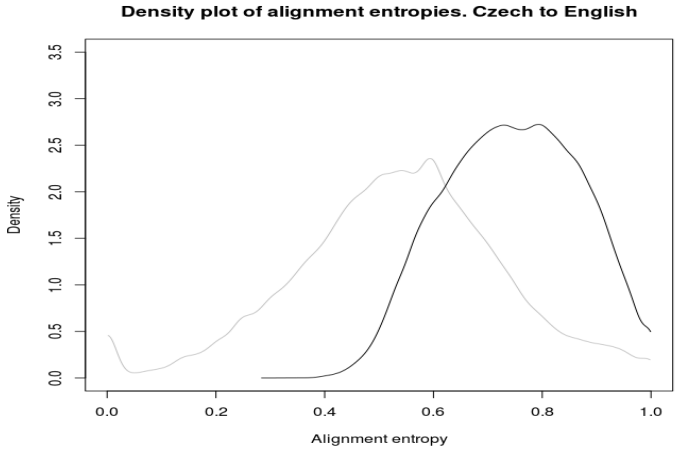


Figure 2. Density plot of the alignment entropies obtained in mean-of-unigram *(grey) and* ngram-to-unigram *(black) experiments for FDA3 and for Czech-to-English translation.*

the statistics of these distributions. They confirm our hypothesis that distribution for *ngram-to-unigram* is centered in higher entropies: 0.745 and 0.7314. In contrast, for *mean-of-unigram* they are 0.6008 and 0.5333. Note also that none of the entropies in *ngram-to-unigram* experiment have a value below 0.3. This makes de values of the features in *ngram-to-unigram* experiment decay slower.

We can observe that the results obtained by the *ngram-to-unigram* experiment for FDA3 are generally better than those of *mean-of-unigram*. While in the first case (Table 1) only one extension performs better than the baseline, in the second case (Table 2) in every extension we obtain improvements for at least two evaluation metrics.

For FDA1, even if the results are not equally satisfactory, we can observe statistically significant improvements in the *ngram-to-unigram* experiment for two of the extensions in Czech-to-English translation.

6. Conclusions and Future Work

In this work we have tried to improve the results of FDA by setting new, n-gram-specific, weights in the decay function, that depend on the uncertainty of the n-grams. In order to do that we proposed two methods for calculating the uncertainty. These methods give an insight into the amount of occurrences an *n*-gram needs in the training data, based on how ambiguous the translation is. We observe that different weights work better for different parameters. Accordingly, finding a good set of values is not enough; it is also necessary to find which parameter performs better. However we demonstrated that it is possible to find a combination that can have a positive

impact on the output. Our findings have proven to be useful both for German-to-English and Czech-to-English translation. An additional finding in this work is that when using unigram features in the default FDA set-up, the output can be as good as (or even better than) using higher order n-gram features.

In the future, we intend to conduct experiments to explore whether having different distributions of the entropies (e.g. more left or right skewed, or different standard deviations) can improve the results. The entropies used in this work where the same for exponential and linear decay factors. Having different sets of weights for each parameter might be beneficial. In addition we want to analyse the outcome when using alignment entropies as input to the init function as well. The source languages used in this work are morphologically richer than the target language. We are also interested in knowing if the improvements are preserved when performing the translation in the reverse direction.

Finally, we want to find a method for obtaining an optimal size of the selected training data.

Acknowledgements

This research is supported by the ADAPT Centre for Digital Content Technology, funded under the SFI Research Centres Programme (Grant 13/RC/2106). This project has received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No 713567.

Bibliography

Banerjee, Satanjeev and Alon Lavie. METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In *Proceedings of the ACL workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization*, pages 65–72, Ann Arbor, Michigan, 2005.

Biçici, Ergun. Feature decay algorithms for fast deployment of accurate statistical machine translation systems. 2013.

Biçici, Ergun and Deniz Yuret. Instance selection for machine translation using feature decay algorithms. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, pages 272–283, Edinburgh, Scotland, 2011.

Biçici, Ergun and Deniz Yuret. Optimizing instance selection for statistical machine translation with feature decay algorithms. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 23(2):339–350, 2015.

Biçici, Ergun, Qun Liu, and Andy Way. ParFDA for fast deployment of accurate statistical machine translation systems, benchmarks, and statistics. In *Proceedings of the Tenth Workshop on Statistical Machine Translation*, pages 74–78, Lisbon, Portugal, 2015.

Bojar, Ondřej, Rajen Chatterjee, Christian Federmann, Barry Haddow, Matthias Huck, Chris Hokamp, Philipp Koehn, Varvara Logacheva, Christof Monz, Matteo Negri, Matt Post, Carolina Scarton, Lucia Specia, and Marco Turchi. Findings of the 2015 Workshop on Statistical

Machine Translation. In *Proceedings of the Tenth Workshop on Statistical Machine Translation*, pages 1–46, Lisboa, Portugal, September 2015. Association for Computational Linguistics. URL http://aclweb.org/anthology/W15-3001.

Doddington, George. Automatic evaluation of machine translation quality using n-gram co-occurrence statistics. In *Proceedings of the second international conference on Human Language Technology Research*, pages 138–145, San Diego, CA, 2002.

Dyer, Chris, Victor Chahuneau, and Noah Smith. A simple, fast, and effective reparameterization of IBM model 2. In *Proceedings of Human Language Technologies: Conference of the North American Chapter of the Association of Computational Linguistics, Proceedings*, pages 644–648, Atlanta, Georgia, USA, 2013.

Heafield, Kenneth. KenLM: Faster and smaller language model queries. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, pages 187–197, Edinburgh, Scotland, 2011.

Koehn, Philipp. Statistical Significance Tests for Machine Translation Evaluation. In *Proceedings of EMNLP 2004*, pages 388–395, Barcelona, Spain, 2004.

Koehn, Philipp, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondřej Bojar, Alexandra Constantin, and Evan Herbst. Moses: Open source toolkit for SMT. In *Proceedings of 45th annual meeting of the ACL on interactive poster & demonstration sessions*, pages 177–180, Prague, Czech Republic, 2007.

Och, Franz. Minimum error rate training in statistical machine translation. In *ACL-2003: 41st Annual Meeting of the Association for Computational Linguistics, Proceedings*, pages 160–167, Sapporo, Japan, 2003.

Och, Franz and Hermann Ney. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51, 2003.

Ozdowska, Sylwia and Andy Way. Optimal bilingual data for French-English PB-SMT. 2009.

Papineni, Kishore, Salim Roukos, Todd Ward, and Wei-Jing Zhu. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the 40th ACL*, pages 311–318, Philadelphia, PA, USA, 2002.

Poncelas, Alberto, Andy Way, and Antonio Toral. Extending Feature Decay Algorithms using Alignment Entropy. 2016.

Popovic, Maja. chrF: character n-gram F-score for automatic MT evaluation. In *Proceedings of the Tenth Workshop on Statistical Machine Translation*, pages 392–395, Lisbon, Portugal, 2015.

Snover, Matthew, Bonnie Dorr, Richard Schwartz, Linnea Micciulla, and John Makhoul. A study of translation edit rate with targeted human annotation. In *Proceedings of the 7th Conference of the Association for Machine Translation in the Americas*, pages 223–231, Cambridge, Massachusetts, USA, 2006.

Address for correspondence:
Alberto Poncelas
alberto.poncelas@adaptcentre.ie
ADAPT Centre, School of Computing,
Dublin City University, Ireland

PBML

The Prague Bulletin of Mathematical Linguistics
NUMBER 108 JUNE 2017 257–269

Optimizing Tokenization Choice for Machine Translation across Multiple Target Languages

Nasser Zalmout, Nizar Habash

Computational Approaches to Modeling Language Lab
New York University Abu Dhabi, United Arab Emirates

Abstract

Tokenization is very helpful for Statistical Machine Translation (SMT), especially when translating from morphologically rich languages. Typically, a single tokenization scheme is applied to the entire source-language text and regardless of the target language. In this paper, we evaluate the hypothesis that SMT performance may benefit from different tokenization schemes for different words within the same text, and also for different target languages. We apply this approach to Arabic as a source language, with five target languages of varying morphological complexity: English, French, Spanish, Russian and Chinese. Our results show that different target languages indeed require different source-language schemes; and a context-variable tokenization scheme can outperform a context-constant scheme with a statistically significant performance enhancement of about 1.4 BLEU points.

1. Introduction

In Statistical Machine Translation (SMT), words are usually designated as the basic tokens of translation and language modeling. However, especially for morphologically complex languages, using sub-lexical units obtained after morphological preprocessing has been shown to improve the machine translation performance over a word-based system (Popović and Ney, 2004; Habash and Sadat, 2006). For any language, several word tokenization choices, henceforth *tokenization schemes*, can be generated based on the word's in-context morphological analysis. These schemes vary by the intended amount of verbosity for the language and application context, and considered a blueprint for the tokenization process. Tokenization using these schemes is usually

 Corresponding author: nasser.zalmout@nyu.edu
Cite as: Nasser Zalmout, Nizar Habash. Optimizing Tokenization Choice for Machine Translation across Multiple Target Languages. The Prague Bulletin of Mathematical Linguistics No. 108, 2017, pp. 257–269.
doi: 10.1515/pralin-2017-0025.

performed as a preprocessing step to the SMT system, where the choice of the scheme is fixed and predetermined. The limitation of a predetermined single tokenization raises many questions: (a) would the best source language tokenization choice vary given different target languages? (b) would combining the various tokenization options in the training phase enhance the SMT performance? and (c) would considering different tokenization options at decoding time improve SMT performance?

The goal of the approach presented in this paper is to eliminate the fixed predetermined scheme selection that spans the entire text, and target languages, and allow for word-level tokenization scheme selection. This notion of word-level tokenization optimization can be achieved indirectly by combining training tokenization options, directly by lattice decoding of the various tokenization options, or through another indirect approach by learning a classifier on optimal tokenization choices. We apply these techniques on Arabic, where most tokenization contributions for SMT focus on Arabic-English translation, with little investigation of other target languages. We study the Arabic tokenization behavior against five target languages: English, French, Spanish, Russian and Chinese. We also introduce a new tokenization scheme to match some of their linguistic features.

2. Arabic Linguistic Issues

Arabic is a morphologically complex language, with various morphological features that control several inflectional variations, such as gender, number, person and voice, producing a large number of rich word forms. Moreover, clitics in Arabic are written attached to the word and thus increase its ambiguity, making word boundaries harder to detect properly. These morphological structures and attached clitics pose a special challenge for NLP tasks in general. These issues are particularly challenging for the tasks that are highly sensitive to the verbosity of the underlying sentences, like SMT, where each morpheme can be aligned to specific target language word. Figure 1 shows an example of such alignment, where a three-word Arabic sentence is aligned to an eight-word English sentence. Tokenization handles this issue by splitting the different clitics with various levels of verbosity, which helps reducing sparsity, perplexity, and out of vocabulary words.

The tokenization process depends on the morphological structure of the word, to identify the suitable morphemic decomposition. Hence, the first step in the tokenization process is to obtain the various morphological analyses of the given word, and choose the most likely one given the contextual surrounding, through a disambiguation process. The next step is choosing the tokenization scheme that the tokenization tool should use given the disambiguated morphological analysis. These schemes serve as a blueprint for the tokenization process, by controlling the types of clitics to be segmented, hence controlling the level of verbosity of the output texts.

There have been several tokenization schemes proposed in literature for Arabic, some of which include the schemes below, with examples provided at Table 1. An important observation about all these schemes, however, is that their outputs are not mutually exclusive, so multiple schemes might sometimes result in the same tokenization.

- Simple Tokenization (D0): Splits off punctuation and numbers, and optionally normalizing some linguistic phenomena.
- D1, D2, and D3: Decliticizations; using different levels of conjugation clitics splits.
- Penn Arabic Treebank (ATB) tokenization: Splits all clitics except the definite article.

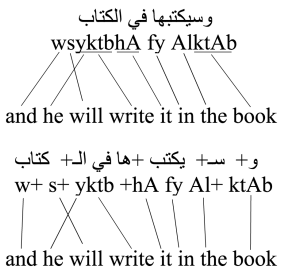


Figure 1. An example of Arabic alignment with English.

Other schemes include the MR (Morphemes); breaks up words into stem and affixival morphemes, and English-like scheme; using lexeme and English-like POS tags.

Selecting the relevant tokenization schemes is predetermined and fixed given the context and application, along with the intended level of verbosity. Moreover, for Arabic SMT, most of the previous contributions on tokenization focus on translating from Arabic to English or vice versa, generalizing tokenization selections to other languages and application domains.

Tokenization Scheme		Example	
D0	no tokenization	wsyktbhA	llTAlb
D1	split CONJ	w+ syktbhA	llTAlb
D2	split CONJ and PART	w+ s+ yktbhA	l+ AlTAlb
ATB	Arabic Treebank	w+ s+ yktb +hA	l+ AlTAlb
D3	split all clitics	w+ s+ yktb +hA	l+ Al+ TAlb

Table 1. Various Arabic tokenization schemes for the sentence wsyktbhA llTAlb 'and he will write it for the student'. Arabic words are presented in Buckwalter transliteration.

3. Background and Related Work

There have been several approaches for Arabic tokenization in literature. Lee et al. (2003) use a look-up table for the various prefixes, stems, and suffixes used in the tokenization process. Habash and Sadat (2006) presented various schemes for tokenizing Arabic text for MT, in addition to the Arabic Treebank tokenization (Maamouri et al., 2004). Diab et al. (2007) presented an SVM-based approach for tokenization. They use a classification based model, where each letter in a word is tagged with a label

indicating its morphological identity. FARASA (Abdelali et al., 2016) uses SVM-rank to rank potential word segmentations. MADAMIRA (Pasha et al., 2014); the current state-of-the-art tool for Arabic morphological analysis and disambiguation, obtains the disambiguated morphological analysis of the word, and feeds it to a tokenization engine. MADAMIRA utilizes MADA (Habash and Rambow, 2005; Roth et al., 2008) for morphological disambiguation. The top morphological analysis is then used for tokenization deterministically through one of the tokenization schemes. We use MADAMIRA to get the various word-level tokenization options, resulting from the various tokenization schemes, then analyze these for the optimal tokenization.

The issues of fixed and text-level selection of tokenization schemes has been previously addressed in literature, for morphologically complex languages in general, and Arabic in particular. Sadat and Habash (2006) presented a technique for maximizing the line-level output BLEU score of the SMT system by combining/consulting outputs of various SMT systems. A "deeper" version of their work that handles tokenization in decoding phase requires a "privilege" scheme, which creates a bias in the system. Moreover, their overall system focuses on optimizing over the SMT output, rather than selecting optimal tokenized inputs. Elming and Habash (2007) used the various tokenization options to build a machine learning model to enhance the quality of word alignments, rather than SMT. Other approaches for unsupervised morphological segmentation includes the work of Mermer (2010) for Turkish-English translation. They use IBM model-1 to formulate the translation objective function as the posterior probability of the training corpus according to a generative segmentation-translation model. Their model, however, didn't exhibit any significant BLEU enhancement. One of the notable contributions within this domain is the work of Dyer et al. (2008) and (Dyer, 2009), where they use a word lattice that encodes the surface forms (unsegmented words) as an option, and the full morphological breakdown of the surface form as another option. In this scope, the lattice is used to model a back-off system for the full morphological segmentation, rather than encoding the various tokenization schemes. Word lattices have also been used for a number of different applications in MT, including the work of Zhang et al. (2007), who use word lattices to model the different chunk-level reordering options.

We use a similar approach to Dyer's (Dyer et al., 2008) for lattice-based decoding of tokenization options, but through encoding all tokenization options at the lattice instead of using it as a backoff model to full morphological breakdown as they use it.

Word lattices and confusion networks are used in NLP mainly to model ambiguity in the input/output, and can be used to represent any finite set of strings. Word lattices, though, have the capability of representing an exponential number of sentences in polynomial space. The words within the lattice represent alternative choices of words in hypothesis, and the edges are used to model the weight or probability score.

4. Approach and Experimental Setup

We first build scheme-specific SMT systems for each language, with six schemes each. We then experiment with a simple scheme combination method, by combining different copies of the training set, each tokenized with a different scheme. Then we apply decoding-time scheme selection, through word lattice decoding of the test set. We finally develop a machine learning tool to learn the optimal tokenizations, as a tradeoff between execution complexity and accuracy.

MT Toolkits and Evaluation We use the Moses toolkit (Koehn et al., 2007) with default parameters to develop the machine translation systems, GIZA++ (Och and Ney, 2003) for alignment, and KenLM (Heafield et al., 2013) to build a 5-gram language model. We use BLEU score (Papineni et al., 2002) and METEOR (Banerjee and Lavie, 2005) for evaluation. Koehn (2004) presents a model for applying statistical significance tests over SMT evaluation metrics. He uses the bootstrap resampling method to measure the p-level statistical confidence. We use this approach for statistical significance tests throughout this paper, with p-value of 0.05.

Data and Preprocessing We use the Multi UN corpus (Eisele and Chen, 2010) throughout the experiments presented in this paper. We chose the Multi UN corpus to study tokenization behavior across several target languages without introducing additional variations. The UN corpus is a good fit as it is parallel for Arabic across five other languages, unlike other commonly used MT corpora.

We use 200,000 lines (circa 5.5 million words) for training, 1,000 lines (circa 25,000 words) for tuning, 3,000 lines (circa 90,000 words) for testing, and 9.5 million lines (circa 280 million words) for language models. The numbers are very similar across all languages we work with. We work with the relatively medium dataset sizes to best capture the tokenization effect, where data sparsity becomes of more relevance. The sparsity issue is particularly important when translating low-resource languages or domains (unlike English for example), which are of interest in this paper.

The preprocessing of the training data includes eliminating the lines beyond the length of 80 words. However, different tokenization schemes will result in different line lengths, which might cause imbalances among the different options. We therefore eliminate the lines across all files whose D3 tokenization exceeds 80 words. Considering D3, the most verbose scheme, as the basis for this elimination guarantees that there won't be any file containing lines exceeding 80 words.

We tokenize the Arabic content using the MADAMIRA toolkit (Pasha et al., 2014), with the alef/yaa normalization, to the various tokenization schemes (D0, D1, D2, ATB, D3). We also use off-the-shelve tools to tokenize the other five languages covered in the paper. We use the available tokenizers at Moses for English, Spanish and Russian, and use the Stanford Word Segmenter from the Stanford NLP Group for Chinese and French (using the TokenizerAnnotator tool).

***D3*: A New Tokenization Scheme** Many languages don't have a clear equivalent of the definite article *"the"*, or *"Al"* in Arabic, like Russian and Chinese. We suggest that removing the definite article in the tokenized source text (Arabic) when translating to these languages might enhance performance. To approach this issue we include a new tokenization scheme in our analysis, by removing the definite article *"Al"* from D3 scheme; which is the only scheme that splits the definite article among the schemes we work with. We designate this new scheme as *D3*.

5. Results and Analysis

We use the same dataset throughout the different experiments, with the same training/tuning/testing splits covered earlier. Each section below presents a different approach into tackling the scheme selection for Arabic tokenization.

5.1. Scheme Specific SMT Systems

The first set of experiments study the various tokenization schemes in isolation. We develop a total of 30 machine translation systems in this part, each corresponding to a specific tokenization scheme for each of the five target languages. Table 2

	English		French		Russian		Spanish		Chinese	
	BLEU	METEOR	BLEU	METEOR	BLEU	METEOR	BLEU	METEOR	BLEU	METEOR
D0	39.80	0.3736	26.79	0.4478	28.56	0.4659	40.70	0.6019	32.04	0.4815
D1	41.25	0.3805	27.71	0.4586	29.47	0.4827	40.92	0.6096	33.23	0.4954
D2	41.62	0.3839	27.89	0.4627	29.85	0.4880	**41.85**	0.6134	33.30	0.4971
D3	41.85	0.3807	27.89	0.4618	29.49	0.4881	41.47	**0.6153**	31.73	0.4848
ATB	41.91	0.3837	**27.91**	**0.4644**	30.38	0.4938	41.61	0.6140	33.30	0.4975
D3*	**41.94**	**0.3846**	27.76	0.4626	**30.55**	**0.4964**	41.66	0.6148	**33.51**	**0.4986**

Table 2. Scheme-specific SMT systems - baselines

presents the BLEU and METEOR for the 30 machine translation systems developed for analyzing the effect of varying tokenization schemes.

The character-level Chinese system outperforms word-level evaluation significantly, matching the results of Habash and Hu (2009). Both sets of results are directly correlated, however, so we present the character-level scores only.

In the Arabic-English systems both ATB and D3* perform closely. This behavior is consistent with the generally used tokenization scheme for Arabic with English as target language in literature, mostly working with ATB. French, on the other hand, shows consistent behavior favoring the ATB scheme for machine translation. The re-

sults for Spanish show that D2 and D3 outperform ATB and D3*. D3* performs the best in both BLEU and METEOR for Chinese and Russian, so our hypothesis proved right.

5.2. Training on Combined Schemes

The first schemes combination method we try is based on simple concatenation of the source training dataset copies (copies of the same previous training dataset), having each tokenized with a different tokenization scheme. The tokenization options resulting from the tokenization schemes are not mutually exclusive, so multiple schemes might result in the same tokenization in certain cases.

The dataset itself is copied and concatenated, so this doesn't constitute a bigger training set, it is rather a richer representation of the same set with additional tokenization options. For sanity check regarding the data duplication, we conducted side experiments by training the MT systems based on the individual schemes, having the training dataset duplicated six times. This did not result in any improvement, so we confirm that any overall improvement is not the result of duplicating the training data. We also duplicate each target language to match the source language (Arabic).

We then perform 30 additional experiments to test each individual tokenization scheme against this combined corpus. We tokenize the testing dataset for each language using each of the six schemes, and use it as a separate testing set for the system trained on the combined corpus.

Table 3 provides the results for the various experiments. The results show a noticeable improvement across all languages and for both BLEU and METEOR. This shows that providing more tokenization options at the training phase enhances the overall MT system performance. The results also show that ATB performs better than the other schemes across English, French, and Chinese, beating the scores for the D3*, even for Chinese where it showed considerable improvement earlier. A potential analysis is that concatenating the training files might have created a bias in the phrase-table model towards phrases that include the definite article, since all other schemes include the article within the tokenization (whether attached or segmented).

Russian and Spanish remain consistent in favoring D3* and D2 respectively, since Russian performs quite closely for D2, ATB and D3* at around 31 BLEU points.

5.3. Word Lattice Input

The word lattice decoding follows the noisier channel model (Dyer et al., 2008). Word lattices are primarily used to model ambiguity in NLP systems, this ambiguity can be referred to by an observed ambiguity signal, which produces a set of source-language strings $f' \in F(s)$. The objective function within this scope would be: $\hat{e} = \mathrm{argmax}_e \mathrm{max}_{f' \in F(s)} \Pr(e)\Pr(f'|e)\Pr(s|f')$. The different probabilities within the formula include: $\Pr(e)$, the language model; $\Pr(f'|e)$, the translation model, and $\Pr(s|f')$,the tokenization model.

	English		French		Russian		Spanish		Chinese	
	BLEU	METEOR	BLEU	METEOR	BLEU	METEOR	BLEU	METEOR	BLEU	METEOR
D0	42.11	0.3740	27.82	0.4535	29.80	0.4918	41.33	0.6133	32.53	0.4866
D1	42.71	0.3815	28.18	0.4620	30.47	0.4950	41.84	0.6151	33.53	0.4963
D2	42.90	0.3861	28.25	0.4676	**31.01**	0.4994	**42.18**	**0.6165**	33.76	0.4988
D3	41.01	0.3816	27.96	0.4658	30.47	0.4958	41.75	0.6147	32.53	0.4902
ATB	**43.11**	**0.3880**	**28.26**	**0.4690**	31.00	0.5001	42.03	0.6156	**33.93**	**0.5013**
D3*	42.29	0.3849	28.02	0.4615	31.00	**0.5007**	41.45	0.6147	33.73	0.5004

Table 3. SMT results for systems trained on combined schemes

We use the lattice decoding functionality at Moses (Koehn et al., 2007), which uses an approximate variation of this model through maximum entropy. Moses uses Python Lattice Format (PLF) to represent the lattice input. When Moses translates input encoded as a word lattice, the translation it chooses maximizes the translation probability along any path in the input. In the case of confusion networks, however, this means maximizing the translation probability along all distinct tokenization options for each surface form. We build the lattice out of the testing set tokenized with the six tokenization schemes. We use a customized version of the tools used at the (Salloum and Habash, 2012) paper (acquired through personal communication), to encode the lattice in the PLF format.

The results, presented at Table 4, show statistically significant improvement relative to the baselines of the scheme-specific systems, and a statistically significant improvement also relative to the simple combined schemes approach. To better under-

	English		French		Russian		Spanish		Chinese	
	BLEU	METEOR	BLEU	METEOR	BLEU	METEOR	BLEU	METEOR	BLEU	METEOR
Lattice Input	**43.33**	0.3860	**28.59**	**0.470**	**31.28**	**0.5033**	**42.31**	**0.6185**	**34.03**	**0.5016**

Table 4. SMT results for lattice based testing input

stand the resulting optimal tokenization choices, we calculate their similarity against all schemes. We observe that circa 92% of the selected optimal tokenizations are similar to D2, for all five languages. ATB is also very similar to the selected optimal tokens, with average of 91%. The next most similar scheme is D1 (around 90%) then D0 (around 84.5%) and finally D3 (around 69%).

D0	EDw	AlmHkmp	AldA}mp	lltHkym	,	lAhAy
ATB	EDw	AlmHkmp	AldA}mp	l+ AltHkym	,	lAhAy
D3	EDw	Al+ mHkmp	Al+ dA}mp	l+ Al+ tHkym	,	lAhAy
Lattice	EDw	Al+ mHkmp	AldA}mp	l+ AltHkym	,	lAhAy
English	Member of	the permanent court		of arbitration	,	the Hague

Table 5. An example of the resulting lattice tokenization

Table 5 shows an example of the lattice-based tokenization, compared to various other tokenization schemes. The lattice output maintains the definite article Al+ "the" with the words AldA}mp "permanent" and AltHkym "arbitration", while segmenting the article for the word Al+ mHkmp "the court", matching the pattern regarding the definite article "the" at the English sentence.

Error Analysis: Definite Article Behavior The ratio of the definite article Al "the", which is tokenized only at the D3 scheme, for the lattice tokenization relative to the D3 tokenization is 11.7% only.This can be the actual optimal behavior statistically (baseline systems show that D3 performs lower than ATB, the closest scheme in verbosity). This behavior can also be attributed to biases in the combined-schemes training corpus against D3-specific tokens.

6. Learning Optimal Tokenization

The models presented thus far show a significant performance improvement, whether for the combined-schemes approach or for the lattice approach, with about 1.4 BLEU points. For any interestingly large datasets, however, these approaches have limitations to their extent of applicability.

6.1. Motivation and Approach

Despite the successful SMT performance boost for the presented approaches, the execution time for the various involved processes make these models relatively challenging for interestingly large corpora. Some of these processes are executed offline, like training. However, other computationally expensive online processes, like the lattice decoding, hinders the application of the lattice approach severely.

The intuition here is to push these computationally-heavy processes offline. Since the lattice decoding is one of the most demanding processes computationally in the presented pipeline, we propose a model that learns the optimal tokenization choices generated from the lattice decoding process. This model can then be used independently to generate the most relevant word-level tokenization choices. The learning

process is based on the best-paths generated from the lattice, so in effect, the learning process will be unsupervised for there is no need for manually tokenized gold data.

6.2. Machine Learning Process

The machine learning model is intended to provide the optimal tokenization for each word in the testing set, having the model trained on the data generated from the lattice decoding. We approach this problem by learning the optimal tokenization scheme tag for each word, rather than the actual lexical tokenization, from the lattice results. We then apply this model to the testing words. The resulting tags are then used to get the corresponding actual tokenization through a lookup table. The input to the lookup table is the scheme tag and surface form, while the output is the corresponding actual tokenization. We used Conditional Random Fields (CRF) for the learning algorithm, with each line as input sequence. The features we use include the surface form, lemma, part of speech tag (POS), and a boolean mask indicating the presence of the different types of proclitics and enclitics (question, conjunction, preposition, article, among others).

The tokenization options for each surface word are not mutually exclusive, that is, the resulting tokenizations from the different schemes for the same surface form might be similar. The tokenization options for the word "AlmHkmp" mentioned previously are the same for D0, D1, D2, and ATB, which is the same as the surface word. The only different tokenization option is for D3; by splitting the definite article: "Al+ mHkmp". Moreover, as covered at the Arabic tokenization schemes section, the tokenization schemes vary by verbosity as follows (increasing verbosity):

D0 <D1 <D2 <ATB <D3

Since the tokenization options might be similar across several tokenization schemes, we consider the verbosity of the selected scheme label in case the surface word has similar tokenization options to other schemes. The system can assign the most/least verbose scheme, which will be analyzed and discussed at the next section.

6.3. Experiments and Analysis

We apply the CRF approach on the Arabic-English system. We use a dataset of 50K lines (around 1.3M words) to train the system. We apply the lattice pipeline discussed earlier, and obtain the best paths resulting from the lattice decoding through Moses, and use these as the training set. Instead of using the actual training labels for the system evaluation, which might be prone to biases due to different tokenization schemes having similar outputs, we use the actual generated tokenized words, through simple accuracy scores. We then input the resulting tokenized content to the MT system, and use the BLEU score as another evaluation metric.

We use ATB as the baseline for our analysis, since it's the most widely used tokenization scheme for Arabic in literature, and it had the best performance in our baseline systems (along with D3*). Table 6 shows the evaluation scores for the machine

learning system. The system shows a clear improvement over the baseline. We further conducted another experiment regarding the verbosity ordering of the tokenization schemes. The result shows a clear improvement for the decreasing verbosity order (at 93.8%) relative to increasing verbosity (at 90.9%). The execution time for the learning

Evaluation Metric	Score
ATB baseline accuracy	91.73%
ATB baseline MT BLEU score (English)	41.91
CRF accuracy	93.80%
CRF MT BLEU score (English)	42.84

Table 6. The performance of the learnt tokenizer

approach is around 4X less than that of the lattice approach, considering the shared processes with the lattice approach as part of the offline tasks. The resulting BLEU score is 42.84; about 0.9 higher than the ATB baseline; a statistically significant boost, and 0.5 BLEU points lower than the lattice approach. These numbers make the case for using the learnt tokenizer, given the complexity of the lattice approach.

7. Conclusion

We presented several tokenization models that enhance the overall Statistical Machine Translation performance. We applied these models to Arabic and were able to conclude that combining different tokenization options at the training phase of the SMT system enhances the overall performance. We were also able to prove that considering all tokenization options at the decoding phase of the testing set further enhances the performance. We didn't see a significant behavior shift across the different languages when it comes to the schemes combination methods, but the scheme we suggested, D3*, proved efficient for Russian and Chinese. We finally presented a learning approach to model the optimal tokenization options based on the lattice decoding, to facilitate a more practical tokenization process.

Bibliography

Abdelali, Ahmed, Kareem Darwish, Nadir Durrani, and Hamdy Mubarak. Farasa: A Fast and Furious Segmenter for Arabic. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Demonstrations*, pages 11–16, San Diego, California, 2016. URL `http://www.aclweb.org/anthology/N16-3003`.

Banerjee, Satanjeev and Alon Lavie. METEOR: An Automatic Metric for MT Evaluation with Improved Correlation with Human Judgments. In *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, pages

65–72, Ann Arbor, Michigan, June 2005. URL http://www.aclweb.org/anthology/W/W05/W05-0909.

Diab, Mona, Kadri Hacioglu, and Daniel Jurafsky. *Automatic Processing of Modern Standard Arabic Text*, pages 159–179. Springer Netherlands, Dordrecht, 2007. ISBN 978-1-4020-6046-5. doi: 10.1007/978-1-4020-6046-5_9. URL http://dx.doi.org/10.1007/978-1-4020-6046-5_9.

Dyer, Chris. Using a Maximum Entropy Model to Build Segmentation Lattices for MT. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, NAACL '09, pages 406–414, Stroudsburg, PA, USA, 2009. Association for Computational Linguistics. ISBN 978-1-932432-41-1. URL http://dl.acm.org/citation.cfm?id=1620754.1620814.

Dyer, Christopher, Smaranda Muresan, and Philip Resnik. Generalizing Word Lattice Translation. In *Proceedings of ACL-08: HLT*, Columbus, Ohio, 2008.

Eisele, Andreas and Yu Chen. MultiUN: A Multilingual Corpus from United Nation Documents. In Tapias, Daniel, Mike Rosner, Stelios Piperidis, Jan Odjik, Joseph Mariani, Bente Maegaard, Khalid Choukri, and Nicoletta Calzolari (Conference Chair), editors, *Proceedings of the Seventh conference on International Language Resources and Evaluation*, pages 2868–2872. European Language Resources Association (ELRA), 5 2010.

Elming, Jakob and Nizar Habash. Combination of Statistical Word Alignments Based on Multiple Preprocessing Schemes. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Companion Volume, Short Papers*, pages 25–28, Rochester, New York, April 2007. URL http://www.aclweb.org/anthology/N/N07/N07-2007.

Habash, Nizar and Jun Hu. Improving Arabic-Chinese statistical machine translation using English as pivot language. In *Proceedings of the Fourth Workshop on Statistical Machine Translation*, pages 173–181. Association for Computational Linguistics, 2009.

Habash, Nizar and Owen Rambow. Arabic Tokenization, Part-of-Speech Tagging and Morphological Disambiguation in One Fell Swoop. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 573–580, Ann Arbor, Michigan, 2005. URL http://www.aclweb.org/anthology/P/P05/P05-1071.

Habash, Nizar and Fatiha Sadat. Arabic Preprocessing Schemes for Statistical Machine Translation. pages 49–52, New York, NY, 2006.

Heafield, Kenneth, Ivan Pouzyrevsky, Jonathan H. Clark, and Philipp Koehn. Scalable Modified Kneser-Ney Language Model Estimation. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, pages 690–696, Sofia, Bulgaria, August 2013. URL http://kheafield.com/professional/edinburgh/estimate_paper.pdf.

Koehn, Philipp. Statistical Significance Tests for Machine Translation Evaluation. In *Proceedings of EMNLP 2004*, pages 388–395, Barcelona, Spain, July 2004. Association for Computational Linguistics.

Koehn, Philipp, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondrej Bojar, Alexandra Constantin, and Evan Herbst. Moses: Open Source Toolkit for Statistical Machine Translation. In *Proceedings of ACL, demo session*, Prague, Czech Republic, 2007.

Lee, Young-Suk, Kishore Papineni, Salim Roukos, Ossama Emam, and Hany Hassan. Language Model Based Arabic Word Segmentation. pages 399–406, Sapporo, Japan, 2003.

Maamouri, Mohamed, Ann Bies, and Tim Buckwalter. The Penn Arabic Treebank : Building a Largescale Annotated Arabic Vorpus. In *Conference on Arabic Language Resources and Tools*. NEMLAR, 2004.

Mermer, Coşkun. Unsupervised Search for the Optimal Segmentation for Statistical Machine Translation. In *Proceedings of the ACL 2010 Student Research Workshop*, ACLstudent '10, pages 31–36, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics. URL `http://dl.acm.org/citation.cfm?id=1858913.1858919`.

Och, Franz Josef and Hermann Ney. A Systematic Comparison of Various Statistical Alignment Models. *Computational Linguistics*, 29(1):19–52, 2003.

Papineni, Kishore, Salim Roukos, Todd Ward, and Wei-Jing Zhu. BLEU: a Method for Automatic Evaluation of Machine Translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, PA, 2002.

Pasha, Arfath, Mohamed Al-Badrashiny, Ahmed El Kholy, Ramy Eskander, Mona Diab, Nizar Habash, Manoj Pooleery, Owen Rambow, and Ryan Roth. MADAMIRA: A Fast, Comprehensive Tool for Morphological Analysis and Disambiguation of Arabic. In *In Proceedings of LREC*, Reykjavik, Iceland, 2014.

Popović, Maja and Hermann Ney. Towards the Use of Word Stems and Suffixes for Statistical Machine Translation. In *Proceedings of the 4th International Conference on Language Resources and Evaluation (LREC)*, pages 1585–1588, Lisbon, Portugal, May 2004.

Roth, Ryan, Owen Rambow, Nizar Habash, Mona Diab, and Cynthia Rudin. Arabic Morphological Tagging, Diacritization, and Lemmatization Using Lexeme Models and Feature Ranking. In *Proceedings of ACL-08: HLT, Short Papers*, pages 117–120, Columbus, Ohio, 2008. URL `http://www.aclweb.org/anthology/P/P08/P08-2030`.

Sadat, Fatiha and Nizar Habash. Combination of Arabic Preprocessing Schemes for Statistical Machine Translation. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 1–8, Sydney, Australia, July 2006. Association for Computational Linguistics. URL `http://www.aclweb.org/anthology/P/P06/P06-1001`.

Salloum, Wael and Nizar Habash. Elissa: A Dialectal to Standard Arabic Machine Translation System. In *Proceedings of the 24th International Conference on Computational Linguistics (COLING 2012): Demonstration Papers*, pages 385–392, Mumbai, India, 2012.

Zhang, Yuqi, Richard Zens, and Hermann Ney. Improved chunk-level reordering for statistical machine translation. In *IWSLT*, pages 21–28, 2007.

Address for correspondence:
Nasser Zalmout
`nasser.zalmout@nyu.edu`
New York University Abu Dhabi, United Arab Emirates

PBML

The Prague Bulletin of Mathematical Linguistics
NUMBER 108 JUNE 2017 271–282

Providing Morphological Information for SMT Using Neural Networks

Peyman Passban, Qun Liu, Andy Way

ADAPT Centre, School of Computing, Dublin City University, Ireland.

Abstract

Treating morphologically complex words (MCWs) as atomic units in translation would not yield a desirable result. Such words are complicated constituents with meaningful subunits. A complex word in a morphologically rich language (MRL) could be associated with a number of words or even a full sentence in a simpler language, which means the surface form of complex words should be accompanied with auxiliary morphological information in order to provide a precise translation and a better alignment. In this paper we follow this idea and propose two different methods to convey such information for statistical machine translation (SMT) models. In the first model we enrich factored SMT engines by introducing a new morphological factor which relies on subword-aware word embeddings. In the second model we focus on the language-modeling component. We explore a subword-level neural language model (NLM) to capture sequence-, word- and subword-level dependencies. Our NLM is able to approximate better scores for conditional word probabilities, so the decoder generates more fluent translations. We studied two languages Farsi and German in our experiments and observed significant improvements for both of them.

1. Introduction

Phrase-based SMT (PBSMT) (Koehn et al., 2003) is the state-of-the-art model for providing automatic translations, but it suffers from serious problems. The performance of the PBSMT model considerably decreases in the presence of large vocabularies and a high rate of out-of-vocabulary words. These phenomena are closely tied to morphology-related issues frequently encountered in MRLs. Recently, neural machine translation (NMT) (Cho et al., 2014) has appeared as a very powerful alternative

 Corresponding author: peyman.passban@adaptcentre.ie
Cite as: Peyman Passban, Qun Liu, Andy Way. Providing Morphological Information for SMT Using Neural Networks. The Prague Bulletin of Mathematical Linguistics No. 108, 2017, pp. 271–282.
doi: 10.1515/pralin-2017-0026.

for PBSMT, which is able to generate competitive or in some cases even better results. However NMT suffers from the same problems. Incorrect word selection and generating wrong surface forms are direct consequences of such shortcomings. Accordingly, both paradigms have problems with MRLs, some of which we try to address here. Although we benefit from neural-network-based features, the main interest of the paper is the SMT approach and its enhancement, so we do not study NMT engines.

SMT can be viewed as a sequential pipeline which takes a sentence in a source language, manipulates it step by step and finally produces a target sentence. In such a multi-step process the most compatible data distribution is selected to train the best (task-specific) model. Data selection techniques are designed in this regard. Afterwards the training data is preprocessed during normalization and tokenization to be more readable/understandable for other subsequent steps. Source and target words are aligned to find cross-lingual lexical mappings. Phrases are extracted and models are trained correspondingly. At the final stage the best counterparts of source phrases are discovered through a search-based solution. Target phrases are combined together to make a coherent and fluent translation. In special cases some post-translation processing is also applied to the final translation. All of these steps are carried out using statistical models which rely heavily on word co-occurrences.

Neural models are known as powerful techniques to capture semantic information. They provide richer information than count-based and statistical models. There are several research papers which boost the aforementioned SMT sub-modules via neural techniques. Duh et al. (2013) uses neural networks (NNs) to select better sentences to train high-quality SMT engines. Tamura et al. (2014) explore neural alternatives instead of the EM-based model for word alignment. Li et al. (2014) design a neural reordering function.

In this paper we also try to model morphological information using NNs. To this end we propose two solutions: (i) we introduce morphological features for the factored translation model (Koehn and Hoang, 2007), and (ii) we manipulate a language model (LM) to incorporate morphological information.

The factored translation model (FTM) is one of the most suitable frameworks to include different annotations at decoding time, such as morphological information. The main problem with PBSMT is that it translates text phrases without any explicit use of linguistic information, which seems crucial for a fluent translation. In FTMs each word is extended by a set of annotations, so that a word in this framework is not only a token but a vector of factors, e.g. a simple word in PBSMT can be represented by a vector of {*word (surface form), lemma, part-of-speech (POS) tag, word class, morphological information*} in its factored counterpart. Clearly the new representation is richer than the word's surface-form. Since the main focus in FTMs is on word-level enrichments, it addresses the problem of morphology which is the main interest of this paper.

In word- or phrase-based approaches, each word is treated independently, i.e. *'studies'* has no relation to *'studied'*. If only one of them was seen during training, translation of the other one would be hard (or impossible) for any SMT engine, even

though both words come from the same stem. Translation knowledge of their shared stem along with auxiliary morphological information could help us translate both of them. This property not only provides solutions for these types of morphological issues but also addresses the data sparsity problem at the same time. A factored translation model follows a similar approach and performs better than other models (which rely on surface forms) for MRLs.

Translation in FTMs is generally broken up into two translation and one generation steps. A source lemma is translated into a target lemma. Morphological and POS factors are translated into target forms and the final form is generated based on the lemma and other factors. Factored models follow the same implementation framework as the phrase-based model. In these models the translation step operates at the phrase level whereas generation steps are word-level operators. For more information on FTMs see Koehn and Hoang (2007). In our modified FTM we have four factors of the *surface form*, *lemma*, *POS tag* and *morphology tag* for each word. It is clear how the first three factors are defined. The last factor is based on morphology-aware embeddings. First we train word embeddings (see Section 2.1) which preserve subword-level and morphological information. Then we cluster words based on their embeddings. The cluster label of each word indicates its morphological tag.

As previously mentioned, along with the translation model we try to enrich an LM. The LM is the main source of monolingual knowledge in translation which plays a key role in providing fluent translations. This module is the best means by which we can directly impose morphological constraints. In PBSMT models n-gram LMs are explored, whereas we benefit from a neural variant in our case. We selected Farsi (Fa) and German (De) for our experiments. Farsi is a morphologically rich and a low-resource language. Therefore, any small improvement in such a language could be a valuable achievement. Beside Farsi experiments we also evaluate our models on German. This language is well-studied in the field of MT and there exist plenty of experimental studies on German, but we use it to provide better comparisons with previous work and show the strength and weakness of our models.

2. Proposed Models

2.1. Enriching Word Embeddings with Subword Information

Words are not always usable in their original forms, as they are symbolic units and need to be transformed into numerical forms for some applications. Each word carries a particular type of information and has specific syntactic and semantic roles. The word's relation with other constituents is also a key property which is defined exclusively for each word. Considering all of these features, it is quite challenging to find a numerical counterpart for a word, which preserves all of these properties and represents the same word in a numerical feature space. To this end there are well-known models such as Salton et al. (1975) which try to transfer words and their syntactic

and semantic information. Recently, NNs have become the established state-of-the-art for creating distributed representations of words (and also other textual units such as characters etc.). Hinton (1986) proposed an NN-based embedding model for the first time, introducing the idea of a *shared learning space*, where the embeddings (word vectors) themselves are also trainable parameters of the model.

Word embeddings are real-valued representations in an n-dimensional feature space. Recent work has shown that these distributed representations can preserve meanings, as well as semantic and syntactic dependencies. However, existing word-based models have some deficiencies, especially with regard to MRLs. In these models, each word is treated as an atomic unit which is not an appropriate way of processing MCWs. In this section we propose a new technique designed to model intra-word relations. Word-based models (Mikolov et al., 2013; Pennington et al., 2014) are not able to (efficiently) transfer rare words. Our model composes word embeddings from subunit embeddings and tries to solve this problem.

There are several models to train word embeddings. One of the most successful models is *Word2Vec* proposed by Mikolov et al. (2013). Almost all other work has followed this unsupervised approach. The main intuition behind our model is the same, but the internal operation is quite different. *Word2Vec* is a simple feed-forward model in which a target random word of an input sequence is selected to be predicted by means of its surrounding context. Word vectors are updated with respect to error values of the prediction phase. More formally the network tries to compute $P(w_i|C)$ where w_i is the target word and C indicates its context. In the simplest scenario the context C is the preceding word just before the target word and the network includes one hidden layer h with the weight matrices $W_{i:h} \in \mathbb{R}^{|input| \times d}$ and $W_{h:o} \in \mathbb{R}^{d \times |\mathcal{V}|}$. $\mathcal{V}$ is the vocabulary set and d is the size of h. The probability of each word given its context is estimated via a *softmax* function, which is a scalar that maps values of its input vector into the range $[0, 1]$, so that new values can be interpreted as probabilities. This scalar is formulated as in (1):

$$P(w_i = j|C) = \frac{\exp(h_t.w^j + b^j)}{\sum_{j' \in \mathcal{V}} \exp(h_t.w^{j'} + b^{j'})} \tag{1}$$

where w_i is the j-th column of $W_{h:o}$ and b_j is a bias term. Input to the *softmax* function is $h_t \in \mathbb{R}^d$ and its output is $v \in \mathbb{R}^{|\mathcal{V}|}$. The j-th cell of v is interpreted as the probability of selecting the j-th word from $\mathcal{V}$ as the target word. Based on *softmax* values the word with the highest probability is selected and the error is computed correspondingly. Error values are back-propagated to the NN in order to update network parameters. Word embeddings are part of those parameters which are updated.

Our model is a simple extension of the basic *Word2Vec* model. In the basic model the surface form of words are taken into account, whereas we segment each word into its stem and affixes, and the embedding of the surface form of each word is a composition of its subunit embeddings, i.e. the embedding of w_i is obtained by $\mathcal{E}(w_i) = \left(\sum_{m \in \mathcal{M}(w_i)} \mathcal{E}(m)\right) + \mathcal{E}(w_i)$, where $\mathcal{E}$ is the embedding form. w_i may have

several morphemes (subunits) where $\mathcal{M}(w_i)$ is the set of all possible subunits of w_i. We show an example from our training corpus to clarify the mechanism of making word embeddings. For the given word *'pre.process.ing.s'* the embedding is generated with this computation: $\mathcal{E}(\mathrm{pre}) + \mathcal{E}(\mathrm{process}) + \mathcal{E}(\mathrm{ing}) + \mathcal{E}(\mathrm{s}) + \mathcal{E}(\mathrm{preprocessings})$.

In our model we treat the word's surface form as another internal subunit, because it makes the approach more robust to noisy morphological segmentations. This strategy also generates better embeddings. We process all sentences of our training corpora. Words are segmented using *Morfessor* (Smit et al., 2014). We have a unique embedding vector for each subunit in our neural architecture. Subunit embeddings reside in a look-up table whose values are updated during training. Based on the input sentence the target word is randomly selected and the context vector is generated. Each word's embedding in the context vector is a linear combination of its subunits. The model tries to predict the correct target word at the output layer. Based on the prediction the error value is computed and back-propagated to the network. In the back-propagation pass all network parameters including word and subunit embeddings are updated. After training the model we obtain high-quality embeddings which have information about morphological properties and subunits of words. In Section 3 we show the impact of using such embeddings in SMT engines.

2.2. Training Subword-Aware Neural Language Models

An LM measures how likely a sequence of words is to occur in a text. It addresses the fluency of the given sequence, so that a sequence with a good word order has a high probability. The leading types of LMs are count-based or n-gram models which function based on the Markov chain assumption. In such models the probability of a sequence is computed by the conditional probabilities of words given their history: $P(S) = P(w_1, ..., w_m) = \prod_{i=1}^{m} P(w_i|w_1^{i-1})$, where S is the given sequence with the length m. The model conditions the probability of each word over a chain of preceding words. Since computing the probability over the entire chain is not computationally feasible, it is usually limited to a bounded set of n previous words: $P(w_i|w_1^{i-1}) \simeq P(w_i|w_{i-n}^{i-1})$. The assumption states that the probability of a word is affected by its n preceding words. Obviously, a long history is preferable but such an assumption is made because of computational restrictions and limited data resources. These models are known as n-gram models and the limited-history problem is the main disadvantage of these models.

Recently, NLMs have been proposed as better alternatives for conventional LMs. NLMs are able to compute the word conditional probabilities over the entire chain and mitigate the history problem. They benefit from recurrent neural networks (Zaremba et al., 2014). As the name of these networks shows they have a recurrent mechanism; they process the input sentence word by word. At each step one word is taken as an input and the hidden state(s) is updated correspondingly. This loop continues until visiting the end of the sequence. When the process ends a summary of the entire

sequence resides in hidden states. As the network has access to such rich information it is able to provide a better estimation of word probabilities.

Although NLMs mitigate the history problem, similar to embedding models they also have serious problems with MRLs. In order to make NLMs compatible with MRLs, different models work at morpheme and character levels. We also propose a new hybrid (morpheme+character-level) model. For our NLM we could fine-tune the same architecture as in Section 2.1 (linear combination of subword embeddings), but character-aware models outperform subword-based NLMs. The state-of-the-art model for neural language modeling is the model by Kim et al. (2016) which relies on characters. Therefore, we also prefer to build our NLM over character-aware models.

In the character-aware framework words are segmented into characters. Each character has a dedicated embedding. All character embeddings are combined through a convolutional module. There is a set of different filters with different widths. The idea behind using different filters is to capture different n-gram information where the size of n-gram corresponds to the filter width. The maximum value of each filter, which is the most representative feature is selected to be combined with other maximum values from other filters. The combination of maximum values makes up the word's surface-form embedding. Word embeddings are passed through a highway layer (Srivastava et al., 2015) to make richer information for the following modules. The output of the highway layer is consumed by a Long Short-Term Memory (LSTM) unit (Hochreiter and Schmidhuber, 1997). LSTMs are memory-augmented recurrent models. Simple recurrent models are not able to model long-distance dependencies, but through an internal memory unit defined for LSTMs, they are able to remember the relation among words much better than simple models.

Our model is a simple extension to the character-aware NLM. The main responsibility of the convolutional module is to find relations among characters by using different filters. Instead of this neural computation we define a simpler but more straightforward technique to capture the same type of information. There are sets of consecutive characters in training corpora which always appear together. Since these characters are tied to each other and appear together, we do not decompose them, which means that instead of finding the relation of such a set of characters via different filters, we keep them together and explicitly inform the model about their relation. Therefore, we do not change the neural architecture but rather define a new preprocessing method.

In our model we extract all possible character n-grams. Each word with the length l (characters) could have up to $\frac{l \times (l+1)}{2}$ character n-grams, e.g for the word *'the'* we can extract these character n-grams: {*'t'*, *'h'*, *'e'*, *'th'*, *'he'*, *'the'*}. For each word, first we separate n-grams which are frequent. We keep those blocks (sets of consecutive characters which make the n-gram) as they are and do not segment them. Then we decompose the reminder into characters (if they are not frequent). In this model we start from higher order n-grams, i.e. for a word with l characters we start from $(l-1)$-grams. If

we cannot find a frequent subunit in the higher order (such as $l-1$), we look at lower orders (such as $l-2, l-3, ..., 2$). When we find a frequent l'-gram in a word, this means that there was no frequent character n-gram where $n > l'$. By use of an example from our Farsi[1] corpus (see Section 3) we try to clarify our segmentation method. For the word *'prdrāmdtrynhā'* meaning *'the people with the highest salary'*, the first frequent substring extracted is *'āmd'* which is a 3-gram constituent. This means that there is no frequent n-gram with $n > 3$. *'āmd'* also has the highest frequency among all other 3-grams, so in the presence of several frequent n-grams we select the most frequent one. *'āmd'* is separated and the segmentation model is applied to its preceding and following substrings. Each substring is considered as a new input to the model. We recursively apply the same procedure until all frequent substrings have been separated, which are *'āmd'*, *'dr'* and *'hā'* for this example. There are still three substrings remaining, namely *'pr'*, *'tr'* and *'yn'*. These three substrings are not considered as frequent in our setting, so they are all decomposed into characters. The final decomposition result by the proposed model is: *'prdrāmdtrynhā'* $\Rightarrow$ *'p.r.dr.āmd.t.r.y.n.hā'*. In our experiments we consider a constituent as frequent if it occurs more than 100 times in the entire training corpus.

Our NLM is the same as that of Kim et al. (2016) with one main difference. In the input layer of our model we have blocks instead of characters. Each block could include one or many characters. By using the character blocks we keep related characters together which means we do not need a convolutional (or any other neural) procedure. We explicitly define such information for the network through our blocks, and the convolutional module is a complementary layer to provide richer information about the relation of characters. Using this simple technique we are able to boost the character-aware NLM. We can use the same mechanism as in Section 2.1 in our NLM, namely each word can be represented via a linear combination of its subunits. Botha and Blunsom (2014) implemented this idea for language modeling and considerably improved the performance of previous NLMs. Although this model was quite successful, the model of Kim et al. (2016) outperforms it. Accordingly, we built our NLM via the character-aware model. Table 1 illustrates a simple comparison of these three approaches and shows the impact of our model.

Model	**German (De)**	**Farsi (Fa)**
Botha and Blunsom (2014)	296	-
Kim et al. (2016)	239	128
Proposed Model	**225**	**110**

Table 1. Perplexity scores of different NLMs (lower is better).

[1]We use the DIN transliteration standard to show the Farsi alphabets.

The table reports perplexity scores for different NLMs. The numbers reported for the German experiments from the first two models are taken from Kim et al. (2016). The German models are trained and evaluated on the same dataset as reported in the original paper (Kim et al., 2016). For the Farsi model we selected a corpus of 1 million words from the TEP++ corpus (see Section 3). We selected 1 million words to have the same size with the German corpus. The source code for the character-aware model is publicly available and we can run it on our Farsi corpus. Therefore, as we do not have access to the original morpheme-aware model, it is not possible to report its perplexity over the Farsi corpus.

3. Experimental Study

In this section we evaluate our models on Farsi and German. We selected Farsi as it is a morphologically rich and low-resource language. Because of such difficulties it is quite challenging to develop a reliable MT model for this language. Accordingly, we propose such complementary techniques to enrich existing models. German is one of the well-studied languages in the field of MT and there are plenty of resources and models for this language. Generally, German models are high-quality models and their translation and language models are rich enough to provide acceptable results. Because of large datasets, German models are diverse and cover almost all cases (words, phrases etc.), so they do not need such complementary techniques. It is also hard to show the impact of auxiliary information (morphological information in our case) for German as small improvements are usually lost in the presence of large datasets. Nonetheless we report German results for comparative purposes with acceptable improvements.

In the first experiment we trained De↔En and Fa↔En SMT models. To train the engines we used the TEP++ (Passban et al., 2015) and WMT-15 datasets[2] for Farsi and German, respectively. TEP++ is a collection of ~600K parallel sentences. We used 3K sentences each for testing and tuning, and the rest of the corpus for training. From the En↔De dataset we randomly selected 2M sentences for training. The German model is evaluated on newstest-2015 and tuned using newstest-2013. Our models are trained using Moses (Koehn et al., 2007) with its default configuration. The evaluation metric is BLEU (Papineni et al., 2002) and language models are trained on the target side of our corpora with SRILM (Stolcke, 2002). Language models are 5-gram models. In our FTMs, English and German words are lemmatized via the NLTK toolkit (Bird, 2006) and tagged using the Stanford POS tagger (Toutanova et al., 2003). For Farsi, words are lemmatized with an in-house lemmatizer and tagged with our neural model (Passban et al., 2016b). The English tagger uses the Penn Treebank tagset with 36 tags. The German model uses the STTS[3] tagset with 54 tags and the Farsi

[2] http://www.statmt.org/wmt15/translation-task.html.

[3] http://www.ims.uni-stuttgart.de/forschung/ressourcen/lexika/TagSets/stts-table.html.

model has 37 tags. Table 2 shows the impact of incorporating our embeddings into the SMT pipeline.

Direction	Baseline	$\textbf{Extend}_3$	$\textbf{Extend}_4^w$	$\textbf{Extend}_4^m$
En→De	21.11	21.42	21.57	**21.70**
De→En	29.50	29.58	29.71	**29.78**
En→Fa	21.03	22.14	22.27	**22.61**
Fa→En	29.21	30.53	30.67	**30.91**

Table 2. Enriching the FTM using morpheme-aware word embeddings.

In Table 2 **Baseline** is a PBSMT model and $\textbf{Extend}_3$ is an FTM with 3 factors of {*word, lemma, POS tag*}. $\textbf{Extend}_4^w$ and $\textbf{Extend}_4^m$ show factored models with additional morphological factors (4-factor models), where the first one relies on surface-form word embeddings (*Word2Vec*) and the second on our morpheme-aware embeddings. Word embeddings by nature are real-valued vectors, so they can be easily clustered. The cluster label of a word conveys morphological, syntactic and semantic information about the word. In our training mechanism we highlighted morphological information, so the cluster label could be interpreted as the morphology tag of words which defines the fourth factor. Bold numbers indicate that improvements are statistically significant compared to **Baseline** according to paired bootstrap re-sampling (Koehn, 2004) with $p = 0.05$.

Since Farsi and German are more complicated languages compared to English, we assign 1000 clusters for them and English words are categorized into 200 clusters. As Table 2 shows, the 3-factor model ($\textbf{Extend}_3$) outperforms the baseline PBSMT model. This is an expected result because FTMs are better alternatives for MRLs. The performance obtained by the 3-factor model could be further enhanced via word embeddings. The fourth factor in $\textbf{Extend}_4^m$ provides morphological information which is useful for the decoder to cope with complicated morphological constituents. We have a comparison between basic surface-form and morpheme-aware embeddings. $\textbf{Extend}_4^w$ is based on *Word2Vec* embeddings which inform the decoder with some general and high-level information about words. Such information is useful but not as impactful as the information provided by $\textbf{Extend}_4^m$, which relies on morpheme-aware embeddings and thus provides more specific/relevant information. This comparison demonstrates that the mechanism used in training our embeddings is able to capture morphological information.

In addition to the first experiment we designed another experiment to show the impact of the subword-aware NLM. The baseline model in Table 3 is a PBSMT model with a 5-gram language model. There are several ways to embed an NLM into the SMT pipeline. We could use the NLM to re-rank translation results. We could also

Direction	Baseline	n-gram^{w}	n-gram^{m}	Direction	Baseline	n-gram^{w}	n-gram^{m}
En→De	21.11	21.53	**21.88**	En→Fa	21.03	21.86	**22.36**
De→En	29.50	29.87	**30.43**	Fa→En	29.21	29.91	**31.05**

Table 3. Re-scoring word n-grams with NLMs.

restructure the decoder to score translation hypotheses by the NLM. We chose a third way which re-scores the word n-grams of the existing non-neural LM, i.e. we manipulate the n-gram LM with the NLM. The n-gram LM includes word n-grams and their associated scores (scores which are computed based on the word co-occurrences and the Markov chain assumption). We recompute those scores with our NLM and substitute the new scores with previous ones. In this experiment we use an LM whose word n-grams come from the statistical 5-gram model and their associated scores are computed by the subword-aware NLM. In Table 3 our NLM-based model is shown with **n-gram**^{m}. Results show that decoding with new scores is quite effective and improves translation performance. Along with our subword-aware NLM we trained another NLM which is a two-layer LSTM model and works over words (surface forms). We repeated the language-modeling experiment and re-scored word probabilities with the word-based LSTM model. The final system is **n-gram**^{w}. Although the LSTM-based model enhances the baseline model, its impact in not as great as **n-gram**^{m}. This comparison confirms that morphological information provided by **n-gram**^{m} is more impactful than those of the word-based NLM and n-gram LM.

4. Conclusion and Future Work

In this paper we proposed two new models to incorporate morphological information into the SMT pipeline. In the first model we enriched a factored SMT model via a new factor which relies on morphology-aware word embeddings. In our model we focus on Farsi. There are similar models (Zou et al., 2013; Passban et al., 2016a,c) which benefit from word embeddings to improve translation of Farsi and other languages. They train bilingual embeddings but in our model we used monolingual embeddings for the same task. In the second model we tried to manipulate the conventional n-gram LM and recompute the scores of word n-grams with a subword-aware NLM. Both methods are able to effectively improve existing SMT models. For our future work we will develop NMT models which have compatible architectures with MRLs and explicitly benefit from morphological information.

Acknowledgments

We thank the anonymous reviewers, as well as Meghan Dowling and Abigail Walsh for their helpful comments, and the Irish centre for high-end computing (`www.ichec.ie`) for providing computational infrastructures. This research is supported by Sci-

ence Foundation Ireland at ADAPT: Centre for Digital Content Platform Research (Grant 13/RC/2106).

Bibliography

Bird, Steven. NLTK: the natural language toolkit. In *COLING/ACL*, pages 69–72, 2006.

Botha, Jan A and Phil Blunsom. Compositional Morphology for Word Representations and Language Modelling. In *ICML*, pages 1899–1907, Beijing, China, 2014.

Cho, Kyunghyun, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation. In *EMNLP*, pages 1724–1734, Doha, Qatar, 2014.

Duh, Kevin, Graham Neubig, Katsuhito Sudoh, and Hajime Tsukada. Adaptation Data Selection using Neural Language Models: Experiments in Machine Translation. In *ACL (Volume 2: Short Papers)*, Sofia, Bulgaria, 2013.

Hinton, Geoffrey E. Learning distributed representations of concepts. In *Proceedings of the eighth annual conference of the cognitive science society*, volume 1, page 12. Amherst, MA, 1986.

Hochreiter, Sepp and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8): 1735–1780, 1997.

Kim, Yoon, Yacine Jernite, David Sontag, and Alexander M Rush. Character-aware neural language models. In *AAAI-16*, pages 2741–2749, Phoenix, Arizona, USA, 2016.

Koehn, Philipp. Statistical Significance Tests for Machine Translation Evaluation. In Lin, Dekang and Dekai Wu, editors, *Proceedings of EMNLP 2004*, pages 388–395, Barcelona, Spain, July 2004. Association for Computational Linguistics.

Koehn, Philipp and Hieu Hoang. Factored Translation Models. In *Conference on Empirical Methods in Natural Language Processing Conference on Computational Natural Language Learning (EMNLP-CoNLL)*, pages 868–876, Prague, Czech Republic, 2007.

Koehn, Philipp, Franz Josef Och, and Daniel Marcu. Statistical phrase-based translation. In *NACL*, pages 48–54, Edmonton, Canada, 2003.

Koehn, Philipp, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, et al. Moses: Open source toolkit for statistical machine translation. In *ACL*, pages 177–180, Prague, Czech Republic, 2007.

Li, Peng, Yang Liu, Maosong Sun, Tatsuya Izuha, and Dakun Zhang. A Neural Reordering Model for Phrase-based Translation. In *COLING*, pages 1897–1907, Dublin, Ireland, 2014.

Mikolov, Tomas, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient Estimation of Word Representations in Vector Space. *CoRR*, abs/1301.3781, 2013.

Papineni, Kishore, Salim Roukos, Todd Ward, and Wei-Jing Zhu. BLEU: a method for automatic evaluation of machine translation. In *ACL*, pages 311–318, Pennsylvania, USA, 2002.

Passban, Peyman, Andy Way, and Qun Liu. Benchmarking SMT Performance for Farsi Using the TEP++ Corpus. In *EAMT-15*, pages 82–89, Antalya, Turkey, 2015.

Passban, Peyman, Chris Hokamp, Andy Way, and Qun Liu. Improving Phrase-Based SMT Using Cross-Granularity Embedding Similarity. In *EAMT*, pages 129–140, Riga, Latvia, 2016a.

Passban, Peyman, Qun Liu, and Andy Way. Boosting Neural POS Tagger for Farsi Using Morphological Information. *ACM Transactions on Asian and Low-Resource Language Information Processing (TALLIP)*, 16(1):4:1–4:15, 2016b. ISSN 2375-4699.

Passban, Peyman, Qun Liu, and Andy Way. Enriching Phrase Tables for Statistical Machine Translation Using Mixed Embeddings. In *COLING*, pages 2582–2591, Osaka, Japan, 2016c.

Pennington, Jeffrey, Richard Socher, and Christopher Manning. Glove: Global Vectors for Word Representation. In *EMNLP*, Doha, Qatar, 2014.

Salton, Gerard, Anita Wong, and Chung-Shu Yang. A vector space model for automatic indexing. *Communications of the ACM*, 18(11):613–620, 1975.

Smit, Peter, Sami Virpioja, Stig-Arne Grönroos, and Mikko Kurimo. Morfessor 2.0: Toolkit for statistical morphological segmentation. In *ACL*, pages 21–24, Gothenburg, Sweden, 2014.

Srivastava, Rupesh Kumar, Klaus Greff, and Jürgen Schmidhuber. Highway networks. In *ICML Deep Learning workshop*, Lille, France, 2015.

Stolcke, Andreas. SRILM - an extensible language modeling toolkit. In *INTERSPEECH*, Denver, Colorado, USA, 2002.

Tamura, Akihiro, Taro Watanabe, and Eiichiro Sumita. Recurrent Neural Networks for Word Alignment Model. In *ACL (Volume 1: Long Papers)*, pages 1470–1480, Baltimore, Maryland, 2014.

Toutanova, Kristina, Dan Klein, Christopher D Manning, and Yoram Singer. Feature-rich part-of-speech tagging with a cyclic dependency network. In *NACL*, pages 173–180, 2003.

Zaremba, Wojciech, Ilya Sutskever, and Oriol Vinyals. Recurrent Neural Network Regularization. *CoRR*, abs/1409.2329, 2014.

Zou, Will Y, Richard Socher, Daniel M Cer, and Christopher D Manning. Bilingual Word Embeddings for Phrase-Based Machine Translation. In *EMNLP*, pages 1393–1398, 2013.

Address for correspondence:
Peyman Passban
peyman.passban@adaptcentre.ie
ADAPT Centre, School of Computing, Dublin City University, Ireland.

PBML

The Prague Bulletin of Mathematical Linguistics
NUMBER 108 JUNE 2017 283–294

Neural Networks Classifier for Data Selection in Statistical Machine Translation

Álvaro Peris, Mara Chinea-Ríos, Francisco Casacuberta

Pattern Recognition and Human Language Technology Research Center, Universitat Politècnica de València

Abstract

Corpora are precious resources, as they allow for a proper estimation of statistical machine translation models. Data selection is a variant of the domain adaptation field, aimed to extract those sentences from an out-of-domain corpus that are the most useful to translate a different target domain. We address the data selection problem in statistical machine translation as a classification task. We present a new method, based on neural networks, able to deal with monolingual and bilingual corpora. Empirical results show that our data selection method provides slightly better translation quality, compared to a state-of-the-art method (cross-entropy), requiring substantially less data. Moreover, the results obtained are coherent across different language pairs, demonstrating the robustness of our proposal.

1. Introduction

The performance of a statistical machine translation (SMT) system is dependent on the quantity and quality of the available training data. Typically, SMT systems are trained with all available data, assuming that the more data used to train the system, the better. Nevertheless, it is critical that such data is related to the task at hand. Translation quality is negatively affected when there is a lack of domain-specific training data (Callison-Burch et al., 2007; Koehn, 2010). In addition, growing the amount of data available is only feasible to a certain extent. The aim of data selection (DS) is to properly select for training a subset of sentence pairs from a large sentence pool, so that the translation quality achieved in the target domain is improved.

DS techniques extract monolingual or bilingual data that are similar to the in-domain corpus based on some criteria, either at monolingual or bilingual level. Such

 Corresponding author: lvapeab@prhlt.upv.es
Cite as: Álvaro Peris, Mara Chinea-Ríos, Francisco Casacuberta. Neural Networks Classifier for Data Selection in Statistical Machine Translation. The Prague Bulletin of Mathematical Linguistics No. 108, 2017, pp. 283–294.
doi: 10.1515/pralin-2017-0027.

selection is incorporated into the training data. The similarity metric varies depending on each technique. Cross-entropy (CE) difference is a typical and well-established ranking function (Moore and Lewis, 2010; Axelrod et al., 2011; Mansour et al., 2011; Schwenk et al., 2012; Rousseau, 2013). CE-based methods train n-gram language models on the in-domain corpus to select similar sentences from the out-of-domain corpus according to their CE difference.

On the other hand, distributed representation of words have proliferated spectacularly during the last years in the research community. Neural networks provide powerful tools for processing text, achieving success in text classification (Kim, 2014), machine translation (Sutskever et al., 2014; Bahdanau et al., 2015) or domain adaptation (Joty et al., 2017). Related to the DS field, Duh et al. (2013) leveraged neural language models to perform DS, reporting substantial gains over conventional n-gram language models.

Recently, convolutional neural networks (CNN) (LeCun et al., 1998) have also been used in the domain adaptation field (Chen and Huang, 2016; Chen et al., 2016). In these works, the authors used a similar strategy to the one proposed in Section 3, but in a different domain adaptation case—close to a transductive learning scenario: they have no in-domain training corpus, only a large out-of-domain pool and small sets of translation instances. Their goal was to select from the out-of-domain corpus, the more suitable samples for translating their in-domain corpora.

This paper tackles DS by taking advantage of neural networks as sentence classifiers, with the ultimate goal of obtaining corpora subsets that improve translation quality. In order to make systems scalable, such subsets should be as reduced as possible. Therefore, our goal is twofold: we want to select sentences subsets with the least size possible that improve translation quality.

The main contributions of this paper are:

- We tackle the DS problem for SMT as a classification task, employing CNNs and bidirectional long short-term memory (BLSTM) networks.
- We conduct a wide experimentation, using monolingual and bilingual corpora. The results show that our method outperforms a state-of-the-art DS technique in terms of translation quality and selection sizes.
- We show that both CNNs and BLSTM networks provide a similar performance for the task at hand.
- In order to make results reproducible, we release the source code of our method. Corpora are also publicly available.

The paper is structured as follows. Section 2, presents our neural DS method. We introduce two architectures, for taking into account a monolingual or a bilingual corpus. Section 3 presents a semi-supervised algorithm for training our classifiers. Next, Section 4 describes the experimental framework, detailing and discussing the results obtained. are detailed and discussed. Finally, Section 5 concludes the work, tracing the future lines of research.

2. Data selection

The goal of DS methods consists in selecting a subset S of sentences from an out-of-domain pool of sentences G, based on an in-domain corpus I. The objective is to enhance the performance of a SMT system trained using this selection. Note that, the lesser the size of S is, the easier is to extend the original SMT system.Therefore, the selection S must represent a trade-off between size and translation improvement.

2.1. Data selection using cross-entropy

As mentioned in Section 1, a well-established DS method consists in scoring the sentences from the out-of-domain corpus (G) by their CE difference (Moore and Lewis, 2010). For selecting S, this technique relates the CE given by a language model trained on the in-domain corpus I, together with an out-of-domain language model, computing a score for a sentence $\mathbf{x}$:

$$c(\mathbf{x}) = H_I(\mathbf{x}) - H_G(\mathbf{x}) \tag{1}$$

where H_I and H_G are the in-domain and out-of-domain CE of sentence $\mathbf{x}$, respectively.

Note that this method is defined in terms of I, as defined by the original authors. Even though it would also be feasible to define this method in terms of S, such redefinition lies beyond the scope of this paper, since our purpose is only to use this method only for comparison purposes.

In Axelrod et al. (2011), the authors propose a extension to this monolingual CE method, so that it is able to deal with bilingual information. To this end, they sum the CE difference for each side of the corpus, both source and target. Let I_s and G_s be the in-domain source corpus and the out-of-domain source corpus respectively, and I_t and G_t be the in-domain and out-of-domain target corpora. Then, the CE difference between a source sentence $\mathbf{x}$ and a target sentence $\mathbf{y}$ is defined as:

$$c(\mathbf{x}, \mathbf{y}) = [H_{I_s}(\mathbf{x}) - H_{G_s}(\mathbf{x})] + [H_{I_t}(\mathbf{y}) - H_{G_t}(\mathbf{y})] \tag{2}$$

2.2. Data selection using neural networks

In this work, we tackle the DS problem as a classification task. Let us consider a classifier model M that assigns a probability $p_M(I \mid \mathbf{x})$ to a given sentence $\mathbf{x}$, depending whether $\mathbf{x}$ belongs to the in-domain corpus I or not.

In this case, to obtain the selection S, one could just apply M to each sentence from the out-of-domain pool G and select the most probable ones.

We propose to use a neural classifier, exploring CNN and BLSTM networks as sentence encoders. As shown in Fig. 1 (left), the input sentence is fed to our system following a one-hot codification scheme and is projected to a continuous space by means of a word-embedding matrix. Next, the sequence of word embeddings is processed either by a CNN or a BLSTM network. After this, we stack one or more fully-connected

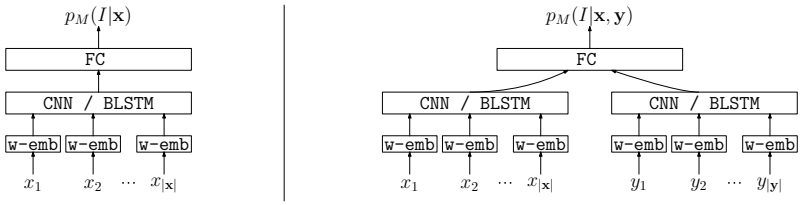


Figure 1: General architecture of the proposed classifiers. The monolingual model is shown at the left while the bilingual model is shown at the right. `w-emb` *stands for word-embedding and* `FC` *for fully-connected layer.*

(FC) layers. Finally, we can apply a softmax function, if we wish to obtain normalized probabilities. All elements can be jointly trained by maximum likelihood.

This reasoning can be extended in order to be applicable to a bilingual corpus. Therefore, if we have the source sentence $\mathbf{x}$ and its corresponding translation $\mathbf{y}$, we can model the probability $\mathrm{p_M(I \mid \mathbf{x}, \mathbf{y})}$. For doing this, we used two networks, one for the source language and another one for the target language. We concatenated their outputs and apply FC layers, as in the previous case, computing an unique score for each bilingual pair. Fig. 1 (right) shows this architecture.

Convolutional neural networks. CNNs have proven their representation capacity, not only in computer vision tasks (Szegedy et al., 2015), but also representing text (Kalchbrenner and Blunsom, 2013; Kim, 2014). In this work, we used the non-static CNN proposed by Kim (2014). This CNN consists in the application of a set of filters to windows of different length. These filters apply a non-linear function (e.g. ReLU). Next, a max-pooling operation is applied to the set of convolutional filters. As result, the CNN obtains a feature vector representing the input sentence.

Recurrent neural networks. In recurrent neural networks, connections form a directed cycle. This allows the network to maintain an internal state and be effective sequence modelers. Moreover, bidirectional networks (Schuster and Paliwal, 1997) have two independent recurrent layers, one processing the input sequence in a forward manner and other processing it a backward manner. Therefore, they allow to exploit the full context at each time-step. Gated units, such as LSTM (Hochreiter and Schmidhuber, 1997; Gers et al., 2000), mitigate the vanishing gradient problem and hence, they are able to properly model long sequences. BLSTM networks can be used for encoding a sentence by concatenating the last hidden state of the forward and backward LSTM layers. This provides a compact representation of the sentence, which accounts for relationships in both time directions.

3. Semi-supervised selection

Properly training these neural classifiers may be a challenging task, since the in-domain data is scarce. Hence, for training them, we follow a semi-supervised iterative protocol (Yarowsky, 1995).

Input: P_0 (positive samples),
N_0 (negative samples),
G_0 (out-of-domain corpus),
l (selection size),
r (training granularity)
Output: P_i (selection of size l)
begin
  $i = 0$
  while $|P_i| \leq l$ **do**
    $M_i \leftarrow$ `Train model on` $\{P_i \cup N_i\}$
    $S_i \leftarrow$ `Classify` G_i `with` M_i
    $P_{i+1} \leftarrow \{P_i \cup$ `get_top`$(S_i, r)\}$
    $N_{i+1} \leftarrow \{N_i \cup$ `get_bottom`$(S_i, r)\}$
    $G_{i+1} \leftarrow \{G_i -$`get_top`$(S_i, r) -$ `get_bottom`$(S_i, r)\}$
    $i++$
  end
  return P_i
end

Algorithm 1: Semi-supervised selection. The functions `get_top` and `get_bottom` select the top-r and the bottom-r scoring sentences from a scored set. The algorithm returns a selection consisting of l sentences.

Algorithm 1 shows this semi-supervised training procedure. Since the data selection is a binary classification problem, we need a set of positive and negative training samples. We start from an initial set of positive samples P_0 and a set of negative samples N_0. At each iteration $i \geq 0$, we train a model with the current sets of data (P_i, N_i). Next, we classify all sentences belonging to the out-of-domain pool (G_i). We extract a number r of top-scoring sentences and include them into the set of positive samples, producing a new set P_{i+1}. Analogously, the r bottom-scoring sentences are included into a new negative samples set N_{i+1}. Hence, at each iteration, we remove $2r$ samples from the out-of-domain set, producing the pool G_{i+1}. Then, a new iteration starts. This is repeated until the selection P_i reaches the desired size (l).

We set our in-domain corpus I as P_0. We randomly extract $|\ I\ |$ sentences from G for constructing N_0. The initial out-of-domain pool G_0 is defined as $\{G - N_0\}$.

4. Experiments in SMT

In this section, we empirically evaluate the DS strategy proposed in Section 2. We conducted experiments on different language pairs for evaluating whether the conclusions drawn from one single language pair hold in further scenarios.

4.1. Corpora

Two corpora are involved within the DS task: an out-of-domain corpus G and an in-domain corpus I. DS selects only a portion of the out-of-domain corpus, and leverages that subset together with the in-domain data to train a, hopefully improved, SMT system. We used the publicly available Europarl (Koehn, 2005) and EMEA (Tiedemann, 2009) corpora as out-of-domain and in-domain data, respectively. As in-domain test sets, we used the Medical-Test and Medical-Mert corpora, partitions established in the 2014 Workshop on Statistical Machine Translation[1]. We focused on the English (En), French (Fr) and German (De) language pairs, conducting experiments in all directions. Table 1 shows the corpora figures.

	EMEA		**Medical-Mert**		**Medical-Test**		**Europarl**	
	$\|S\|$	$\|V\|$	$\|S\|$	$\|V\|$	$\|S\|$	$\|V\|$	$\|S\|$	$\|V\|$
En	1.0M	98k	501	979	1.0k	1.8k	2.0M	157k
Fr		112k		1.0k		26.9k		215k
En	1.1M	99k	500	979	1.0k	1.9k	1.9M	153k
De		141k		874		1.7k		290k

Table 1: Corpora main figures. EMEA is the in-domain corpus, Medical-Test is the evaluation data and Medical-Mert is the development set. Europarl is the out-of-domain corpus. $|S|$ stands for number of sentences and $|V|$ for vocabulary size. M denotes millions of elements and k thousands.

4.2. Experimental setup

All neural models were initialized using word-embedding matrices from word2vec, obtained using the skip-gram model from Mikolov et al. (2013) and trained on part of Google News dataset in the case of English and on Wikipedia in the case of French and German. Word-embedding matrices were fine-tuned during the semi-supervised selection protocol. The size of the word-embeddings was 300.

Following Kim (2014), we used filter windows of lengths 3, 4, 5 with 100 features maps each for the CNN classifier. In order to have a similar number of parameters

[1] http://www.statmt.org/wmt14/medical-task/

than in the CNN (20 million approximately), we used 300 units in each LSTM layer. 2 FC layers of size 200 and 100 were introduced after the CNN and BLSTM (Section 2.2).

For training the CNN classifier, we used Adadelta (Zeiler, 2012) with its default parameters. The BLSTM network was trained with Adam (Kingma and Ba, 2014), with a learning rate of 10^{-4}. During training, we applied Gaussian noise to the weights ($\sigma = 0.01$). All neural models[2] were implemented using the Theano (Theano Development Team, 2016) and Keras libraries. The number of sentences selected at each iteration (r) was chosen trading off speed and granularity ($r = 50,000$).

All SMT experiments were carried out using the open-source phrase-based SMT toolkit Moses (Koehn et al., 2007). The language model used was a 5-gram, with modified Kneser-Ney smoothing (Kneser and Ney, 1995), built with the SRILM toolkit (Stolcke, 2002). The phrase table was generated by means of symmetrised word alignments obtained with GIZA++ (Och and Ney, 2003). The log-lineal combination weights were optimized using MERT (minimum error rate training) (Och, 2003). In order to minimize the random nature of MERT and purposing to provide robustness to the results, every result of this paper constitutes the average of 10 repetitions. In the tables, 95% confidence intervals of these repetitions are shown.

The final translation quality was evaluated by means of BLEU (Papineni et al., 2002). Nevertheless, since we are in a DS scope, the amount of data required for training each system also becomes a fundamental evaluation metric.

The SMT systems were trained using the selection provided by the proposed methods together with the in-domain corpus. We compared the selection methods with two baseline systems. The first one consists in training the SMT system only with in-domain data. We refer to this setup with the name of `bsln-emea`. The second baseline was obtained training with all available data (i.e., in-domain and out-of-domain). We will refer to this setup as `bsln-all`. In addition, we also included results of a purely random sentence selection without replacement.

4.3. Experimental results

Table 2 shows the best results obtained with our DS method using the two neural network architectures proposed (CNN and BLSTM) and the CE method for each language pair.

In En-Fr and En-De, Fr-En, translation quality using DS improves over `bsln-all`, but using significantly less data (20%, 23% and 26% of the total amount of out-of-domain data, respectively). In the case of De-En, translation quality results are similar, but also reducing the amount of data required: only a 23%. According to these results, we can state that our DS strategy is able to deliver similar quality than using all the data, but only with a rough quarter of the data.

[2]Source code available at `https://github.com/lvapeab/sentence-selectioNN`.

	En-Fr		Fr-En	
Strategy	BLEU	#Sentences	BLEU	#Sentences
`bsln-emea`	28.6 ± 0.2	1.0M	29.9 ± 0.2	1.0M
`bsln-all`	29.4 ± 0.1	1.0M+1.5M	32.4 ± 0.1	1.0M+1.5M
Random	29.4 ± 0.4	1.0M+500k	32.3 ± 0.3	1.0M+500k
CE	29.8 ± 0.1	1.0M+450k	31.8 ± 0.1	1.0M+600k
BLSTM	$\mathbf{29.9 \pm 0.3}$	1.0M+**300**k	32.3 ± 0.1	1.0M+500k
CNN	29.8 ± 0.2	1.0M+450k	$\mathbf{32.3 \pm 0.2}$	1.0M+**350**k
	De-En		En-De	
Strategy	BLEU	#Sentences	BLEU	#Sentences
`bsln-emea`	23.7 ± 0.2	1.0M	15.6 ± 0.1	1.0M
`bsln-all`	26.2 ± 0.3	1.0M+1.5M	16.6 ± 0.2	1.0M+1.5M
Random	25.5 ± 0.1	1.0M+600k	16.8 ± 0.1	1.0M+550k
CE	25.5 ± 0.3	1.0M+600k	16.8 ± 0.2	1.0M+500k
BLSTM	25.9 ± 0.1	1.0M+500k	$\mathbf{17.1 \pm 0.2}$	1.0M+**400**k
CNN	$\mathbf{25.9 \pm 0.1}$	1.0M+**400**k	16.9 ± 0.1	1.0M+350k

Table 2: Summary of best results obtained. Columns denote, from left to right: selection strategy, BLEU, number of sentences, given in terms of the in-domain corpus size, and $(+)$ *selected sentences.*

All proposed DS methods are mostly able to improve over random selection but in some cases differences are not significant. It should be noted that beating random is very hard, since all DS methods, including random, will eventually converge to the same point: adding all the data available. The key difference is the amount of data needed for achieving the same translation quality.

Results obtained in terms of BLEU with our DS method are slightly better than the ones obtained with CE difference. However, CE difference requires significantly more sentences to reach comparable translation quality.

Finally, CNN and BLSTM networks seem to perform similarly. Therefore, we conclude that both architectures are good options for this task.

Table 3 shows the best results obtained with our bilingual data selection method using both neural architectures proposed (Bili-CNN and Bili-BLSTM) and bilingual CE (Bili-CE) method for each language pair. Again, the DS selection techniques beat all baselines in terms of BLEU, requiring less data to train the SMT system.

Compared to the monolingual methods, our bilingual DS techniques provide similar results. Nevertheless, in all cases the bilingual methods are able perform better selections at the early stages of the process, as illustrated in Figure 2. As we steadily select more sentences, monolingual and bilingual methods eventually converge to

	En-Fr		Fr-En	
Strategy	BLEU	#Sentences	BLEU	#Sentences
Bili-CE	30.2 ± 0.2	1.0M+350k	32.5 ± 0.1	1.0M+450k
Bili-BLSTM	$\mathbf{30.2 \pm 0.2}$	1.0M+**300k**	32.3 ± 0.1	1.0M+450k
Bili-CNN	30.1 ± 0.3	1.0M+300k	$\mathbf{32.6 \pm 0.2}$	1.0M+**500k**
	De-En		En-De	
Strategy	BLEU	#Sentences	BLEU	#Sentences
Bili-CE	25.9 ± 0.2	1.0M+350k	17.0 ± 0.2	1.0M+500k
Bili-BLSTM	26.0 ± 0.1	1.0M+500k	$\mathbf{17.1 \pm 0.2}$	1.0M+**250k**
Bili-CNN	25.8 ± 0.1	1.0M+200k	17.0 ± 0.1	1.0M+350k

Table 3: Summary of bilingual results obtained. Columns denote, from left to right: selection strategy, BLEU, number of sentences, given in terms of the in-domain corpus size, and $(+)$ selected sentences.

similar results. We can see that adding sentences selected by means of DS techniques improves over the baselines from the very beginning. Selecting at a bilingual level is specially effective in small selections: while the monolingual method requires 150k sentences for beating the `bsln-all` baseline, the bilingual methods only require 50k. Here we show only the En-Fr language pair due to space restriction, but this behavior is consistent across all languages.

5. Conclusion and future work

We developed a DS method, based on sentence classification techniques. The uses CNNs or BLSTM networks for computing a sentence representation. We thoroughly evaluated it over four language pairs. Our method yielded better translation performance than the cross-entropy DS technique, requiring a minor amount of data. Additionally, we found that both CNN and BLSTM networks performed similarly, thus being both suitable sentence encoders.

At the light of the monolingual results, we expected higher gains of performance when considering the both sides of the corpora. It should be tested if a different combination strategy of the classifiers is able to exploit parallel corpora to their full. Moreover, we should also compare the performance of classical classifiers, such as support vector machines (SVM) or logistical regression. We also noted that the De-En language pair had a different behavior than other language pairs. We should study the DS process when applied to inflected languages.

In this work, we chose the initial set of negative samples (N_0) following a random criterion. In the future, we should investigate if a more informed technique (e.g. per-

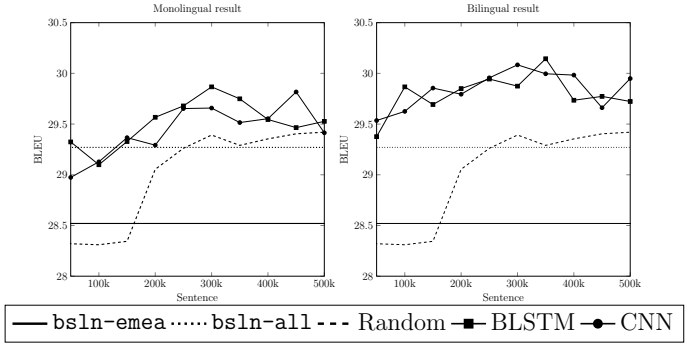


Figure 2: Effect of adding sentences over the BLEU score using the different DS techniques (with monolingual and bilingual form) and random selection techniques for the En-Fr language pair. Horizontal lines represent the scores when using just the in-domain training corpus(`bsln-emea`) and all the data available(`bsln-all`).

plexity or the invitation model from Hoang and Sima'an (2014)) helps the selection system by providing a more suitable N_0.

In addition, we aim to delve into the usage of semi-supervised training strategies for the classifier. Ladder networks (Rasmus et al., 2015) seem a promising tool. We should investigate how to include them in our pipeline. We should also explore one-shot learning strategies in a scenario where only the text to translate is available.

Finally, we should also test our data selection method within the neural machine translation (NMT) technology. NMT systems rely on the usage of large amount of data, but it should be investigated whether the inclusion of in-domain data effectively helps the system. Moreover, as by product of the NMT training, we could use the NMT encoder for pre-initializing our classifier, hoping a boost in the system performance.

Acknowledgements

The research leading to these results has received funding from the Generalitat Valenciana under grant PROMETEOII/2014/030 and the FPI (2014) grant by Universitat Politècnica de València. We also acknowledge NVIDIA for the donation of a GPU used in this work.

Bibliography

Axelrod, Amittai, Xiaodong He, and Jianfeng Gao. Domain adaptation via pseudo in-domain data selection. In *Proc. of EMNLP*, pages 355–362, 2011.

Bahdanau, Dzmitry, Kyunghyun Cho, and Yoshua Bengio. Neural Machine Translation by Jointly Learning to Align and Translate. *arXiv:1409.0473*, 2015.

Callison-Burch, Chris, Cameron Fordyce, Philipp Koehn, Christof Monz, and Josh Schroeder. (Meta-) evaluation of machine translation. In *Proc. of WMT*, pages 136–158, 2007.

Chen, Boxing and Fei Huang. Semi-supervised convolutional networks for translation adaptation with tiny amount of in-domain data. *Proc. of CoNLL*, pages 314–324, 2016.

Chen, Boxing, Roland Kuhn, George Foster, Colin Cherry, and Fei Huang. Bilingual Methods for Adaptive Training Data Selection for Machine Translation. *Proc. of AMTA*, pages 93–103, 2016.

Duh, Kevin, Graham Neubig, Katsuhito Sudoh, and Hajime Tsukada. Adaptation Data Selection using Neural Language Models: Experiments in Machine Translation. In *Proc. of ACL*, pages 678–683, 2013.

Gers, Felix A, Jürgen Schmidhuber, and Fred Cummins. Learning to forget: Continual prediction with LSTM. *Neural Computation*, 12(10):2451–2471, 2000.

Hoang, Cuong and Khalil Sima'an. Latent Domain Translation Models in Mix-of-Domains Haystack, 2014.

Hochreiter, Sepp and Jürgen Schmidhuber. Long Short-Term Memory. *Neural Computation*, 9 (8):1735–1780, 1997.

Joty, Shafiq, Nadir Durrani, Hassan Sajjad, and Ahmed Abdelali. Domain adaptation using neural network joint model. *Computer Speech & Language*, In Press, 2017.

Kalchbrenner, Nal and Phil Blunsom. Recurrent Continuous Translation Models. In *Proc. of EMNLP*, pages 1700–1709, 2013.

Kim, Yoon. Convolutional Neural Networks for Sentence Classification. In *Proc. of EMNLP*, pages 1746–1751, 2014.

Kingma, Diederik P. and Jimmy Ba. Adam: A Method for Stochastic Optimization. *arXiv:1412.6980*, 2014.

Kneser, Reinhard and Hermann Ney. Improved backing-off for m-gram language modeling. In *Proc. of ICASSP*, pages 181–184, 1995.

Koehn, Philipp. Europarl: A parallel corpus for statistical machine translation. In *Proc. of MT Summit*, pages 79–86, 2005.

Koehn, Philipp. *Statistical machine translation*. Cambridge University Press, 2010.

Koehn, Philipp, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, et al. Moses: Open source toolkit for statistical machine translation. In *Proc. of ACL*, pages 177–180, 2007.

LeCun, Y., L. Bottou, Y. Bengio, and P. Haffner. Gradient-Based Learning Applied to Document Recognition. *Proc. of the IEEE*, 86(11):2278–2324, 1998.

Mansour, Saab, Joern Wuebker, and Hermann Ney. Combining translation and language model scoring for domain-specific data filtering. In *Proc. of IWSLT*, pages 222–229, 2011.

Mikolov, Tomas, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Proc. of NIPS*, pages 3111–3119, 2013.

Moore, Robert C and William Lewis. Intelligent selection of language model training data. In *Proc. of ACL*, pages 220–224, 2010.

Och, Franz Josef. Minimum error rate training in statistical machine translation. In *Proc. of ACL*, pages 160–167, 2003.

Och, Franz Josef and Hermann Ney. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51, 2003.

Papineni, Kishore, Salim Roukos, Todd Ward, and Wei-Jing Zhu. BLEU: a method for automatic evaluation of machine translation. In *Proc. of ACL*, pages 311–318, 2002.

Rasmus, Antti, Mathias Berglund, Mikko Honkala, Harri Valpola, and Tapani Raiko. Semi-supervised learning with ladder networks. In *Proc. of NIPS*, pages 3546–3554, 2015.

Rousseau, Anthony. XenC: An Open-Source Tool for Data Selection in Natural Language Processing. *The Prague Bulletin of Mathematical Linguistics*, 100:73–82, 2013.

Schuster, Mike and Kuldip K Paliwal. Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing*, 45(11):2673–2681, 1997.

Schwenk, Holger, Anthony Rousseau, and Mohammed Attik. Large, pruned or continuous space language models on a GPU for statistical machine translation. In *Proc. of NAACL-HLT*, pages 11–19, 2012.

Stolcke, Andreas. SRILM - an Extensible Language Modeling Toolkit. In *Proc. of ICSLP*, pages 901–904, 2002.

Sutskever, Ilya, Oriol Vinyals, and Quoc V. Le. Sequence to Sequence Learning with Neural Networks. In *Proc. of NIPS*, pages 3104–3112, 2014.

Szegedy, Christian, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proc. of CVRP*, pages 1–9, 2015.

Theano Development Team. Theano: A Python framework for fast computation of mathematical expressions. *arXiv:1605.02688*, 2016.

Tiedemann, Jörg. News from OPUS - A collection of multilingual parallel corpora with tools and interfaces. In *Proc. of RANLP*, pages 237–248, 2009.

Yarowsky, David. Unsupervised Word Sense Disambiguation Rivaling Supervised Methods. In *Proc. of ACL*, pages 189–196, 1995.

Zeiler, Matthew D. ADADELTA: An Adaptive Learning Rate Method. *arXiv:1212.5701*, 2012.

Address for correspondence:
Álvaro Peris
lvapeab@prhlt.upv.es
Pattern Recognition and Human Language Technology Research Center,
Universitat Politècnica de València,
Camino de Vera s/n, 46022 Valencia, SPAIN.

PBML

The Prague Bulletin of Mathematical Linguistics
NUMBER 108 JUNE 2017 295–306

Historical Documents Modernization

Miguel Domingo, Mara Chinea-Rios, Francisco Casacuberta

Pattern Recognition and Human Language Technology Research Center
Universitat Politècnica de València - Camino de Vera s/n, 46022 Valencia, Spain

Abstract

Historical documents are mostly accessible to scholars specialized in the period in which the document originated. In order to increase their accessibility to a broader audience and help in the preservation of the cultural heritage, we propose a method to modernized these documents. This method is based in statistical machine translation, and aims at translating historical documents into a modern version of their original language. We tested this method in two different scenarios, obtaining very encouraging results.

1. Introduction

An inherent problem in historical documents is the language in which they are written. Human language evolves with the passage of time, increasing its comprehension for contemporary people. This problem limits the accessibility of historical documents to scholars specialized in the time period in which the document was originated. To break the language barrier, these documents could be translated into a modern version of the language in which they were written.

Most scholars consider a modern version of a historical document to be that version in which words have been updated to match contemporary spelling. This way, the document preserves its original meaning and is easier to read. Fig. 1 shows an example of a historical document with modern spelling. Despite that the new version of the document is easier to read for a person who speaks Spanish, its content is still difficult to comprehend if that person is not specialized in the period in which the document was written.

For this reason, the concept of *modernization* that we propose does not consist only on updating the spelling. We also propose to update the lexicon and grammar to

 Corresponding author: midobal@prhlt.upv.es
Cite as: Miguel Domingo, Mara Chinea-Rios, Francisco Casacuberta. Historical Documents Modernization. The Prague Bulletin of Mathematical Linguistics No. 108, 2017, pp. 295–306. doi: 10.1515/pralin-2017-0028.

En vn lugar de la Mancha, de cuyo nombre no quiero acordarme, no ha mucho tiempo que viuia vn hidalgo de los de lança en astillero, adarga antigua, rozin flaco y galgo corredor. Vna olla de algo mas vaca que carnero, salpicon las mas noches, duelos y quebrantos los sabados, lantejas los viernes, algun palomino de añadidura los domingos, consumian las tres partes de su hazienda. El resto della concluian sayo de velarte, calças de velludo para las fiestas, con sus pantuflos de lo mesmo, y los dias de entre semana se honraua con su vellori de lo mas fino.

En un lugar de la Mancha, de cuyo nombre no quiero acordarme, no ha mucho tiempo que vivía un hidalgo de los de lanza en astillero, adarga antigua, rocín flaco y galgo corredor. Una olla de algo más vaca que carnero, salpicón las más noches, duelos y quebrantos los sábados, lantejas los viernes, algún palomino de añadidura los domingos, consumían las tres partes de su hacienda. El resto de ella concluían sayo de velarte, calzas de velludo para las fiestas, con sus pantuflos de lo mismo, y los días de entre semana se honraba con su vellorí de lo más fino.

Figure 1. Example of a document in which the spelling has been updated to match modern standards. The original text corresponds to the beginning of El Ingenioso Hidalgo Don Qvixote de la Mancha*. The modernized version of the text was obtained from F. Jehle (2001).*

match the current use of the language in which the document was written. Fig. 2 shows Shakespeare's famous Sonnet 18 together with what could be the same sonnet in modern English. The modernized text is not only easier to read but also easier to comprehend. Note that, however, part of the original meaning could be lost in the process. In this case—the original document being a sonnet—part of the rhyme is lost for the sake of clarity. Nonetheless, the goal of increasing the clarity of the document and, thus, its accessibility to a broader audience, is met.

Shall I compare thee to a summer's day?
Thou art more lovely and more temperate:
Rough winds do shake the darling buds of May,
And summer's lease hath all too short a date:
Sometime too hot the eye of heaven shines,
And often is his gold complexion dimm'd;
And every fair from fair sometime declines,
By chance or nature's changing course untrimm'd;
But thy eternal summer shall not fade
Nor lose possession of that fair thou ow'st;
Nor shall Death brag thou wander'st in his shade,
When in eternal lines to time thou grow'st;
So long as men can breathe or eyes can see,
So long lives this, and this gives life to thee.

Shall I compare you to a summer day?
You're lovelier and milder.
Rough winds shake the pretty buds of May,
and summer doesn't last nearly long enough.
Sometimes the sun shines too hot,
and often its golden face is darkened by clouds.
And everything beautiful stops being beautiful,
either by accident or simply in the course of nature.
But your eternal summer will never fade,
nor will you lose possession of your beauty,
nor shall death brag that you are wandering in the underworld,
once you're captured in my eternal verses.
As long as men are alive and have eyes with which to see,
this poem will live and keep you alive.

Figure 2. Example of a document modernization. The original text is Shakespeare Sonnet 18*. The modernized version of the Sonnet was obtained from Crowther (2004).*

Additional problems arise with historical manuscripts. Besides the language barrier, these kind of documents have extra difficulties particular to their author. For instance, they contain a lot of abbreviated words. These abbreviations do not follow any known standard and are usually particular to the time period and writer of the document, with the same writer changing her style during the years. Moreover, in

many occasions, the same word inconsistently appears abbreviated or fully written throughout the same document. Fig. 3 shows an example of a historical manuscript in which this problem is present. The transcription of the manuscript is known as a transliteration, and the version in which abbreviations have been expanded to their corresponding words is known as paleographic version.

al **pmo** capitulo tengo respondido y negado **avr dho** *que* me pesava por no **avr** pecado mas . **ants** he conoscido y conosco pesarme de **coraço** por **avr** pecado en qualquiera tienpo . y a lo ***q*** tengo **dho** ***q*** **pud** ser alguna vez **dzir** ***q*** no me acusava la conciencia de pecado mortal . digo *que* no solo no **teniedome** por justo mas **te**

Al **primero** capitulo tengo respondido y negado **aver dicho** *que* me pesava por no **aver** pecado mas. Antes he conoscido y conosco pesarme de **coraçon** por **aver** pecado en qualquiera tienpo. Y a lo ***que*** tengo **dicho** ***que*** **pudo** ser alguna vez **dezir** ***que*** no me acusava la conciencia de pecado mortal. Digo *que* no solo no **teniendome** por justo mas **teniendome**

Figure 3. Example of a historical manuscript with abbreviations. The left text is a transliteration of the manuscripts, and the right text is known as a paleographic version of the document. Words in ***bold*** *represent abbreviations and their corresponding expansions. Words in* italic *denote words which inconsistently appear abbreviated and fully written throughout the text. Additionally, beginning of sentences have been truecased. The texts from the example belong to the Alcaraz corpus (Villegas et al., 2016).*

In this work, we propose a method to translate historical documents to a contemporary version of the language in which they were written. With this modernized version of a document, we aim at increasing the accessibility of historical documents to a broader audience, as well as helping in the preservation of the cultural heritage: e.g., given a transliteration of a manuscript, this method could be applied to obtain the corresponding paleographic version.

The rest of this paper is structured as follows: Section 2 presents our modernization approach. Then, in Section 3, we describe the experiments conducted in order to assess our proposal. After that, in Section 4, we present the results of those experiments. Finally, conclusions are drawn in Section 5.

2. Modernization

In this section, we present a method to translate a historical document into a contemporaneous version of its language. We also describe two additional techniques to enhance translation quality.

2.1. Statistical Machine Translation

In order to achieve the modernization of historical documents, we propose an approach based on Statistical Machine Translation (SMT). SMT has as a goal to find the best translation $\hat{\mathbf{y}}$ of a given source sentence $\mathbf{x}$ (Brown et al., 1993):

$$\hat{\mathbf{y}} = \arg\max_{\mathbf{y}} \Pr(\mathbf{y} \mid \mathbf{x}) \tag{1}$$

During years, phrase-based models (Koehn, 2010) have been the prevailing approach to compute this expression. These models rely on a log-linear combination of different models (Och and Ney, 2002): namely, phrase-based alignment models, reordering models and language models; among others (Zens et al., 2002; Koehn et al., 2003). However, in the last few years, neural machine translation (Sutskever et al., 2014; Bahdanau et al., 2015) has had a great impact. This novel approach is based on the use of neural networks for carrying out the translation process.

Therefore, considering the document's original language as a source and the modern version of that language as the target, we propose to use phrase-based SMT to obtain a modernized version of the document.

2.2. Data Selection

In order to successfully apply SMT for modernizing a historical document, we need training data as similar as possible as the document to modernize. However, this is not always feasible. To cope with this problem, we propose to use a data selection technique which has been successfully used in SMT to increase the training data with sentences from corpora of different domains than the text to translate, which are as similar as possible to this text.

Infrequent n-grams recovery strategy (Gascó et al., 2012) increases the training corpus by selecting from other corpora the sentences closest to the test set. These sentences contain those n-grams that have been seldom observed in the test set. i.e., the *infrequent n-grams*. An n-gram is considered infrequent when it appears less times than a given infrequency threshold t. Therefore, the idea is to construct a training corpus by selecting from the available corpora those sentences which contain the most infrequent n-grams.

Let X be the set of n-grams that appear in the sentences to be translated; $\mathbf{m}$ one of these n-grams; $R(\mathbf{m})$ the counts of $\mathbf{m}$ in a given source sentence $\mathbf{x}$ from the available corpora; and t a given infrequency threshold. Then, the infrequency score $i(\mathbf{x})$ is defined as:

$$i(\mathbf{x}) = \sum_{\mathbf{m} \in X} \min(1, R(\mathbf{m}))t \tag{2}$$

Therefore, the sentences from the available corpora are scored using Eq. (2). Then, at each iteration, the sentence $\mathbf{x}^*$ with the highest score $i(\mathbf{x}^*)$ is selected and added

to the training corpus. After that, x^* is removed from the available corpora and the counts of the n-grams $R(m)$ are updated within x^*. Consequently, the scores of the corpora are updated. This process is repeated until all the n-grams within X reach frequency t. Once the process is finished, the resulting corpus will be the one used for training the systems.

2.3. Byte Pair Encoding

A common problem in SMT are those rare and unknown words which the system has never seen. This could be a bigger problem when modernizing historical documents due to the constants evolution of the language as well as, in the case of manuscripts, the aforementioned problem with abbreviations (see Section 1). An innovative solution to tackle this problem is Byte Pair Encoding (BPE) (Sennrich et al., 2016).

Based on the intuition that various word classes are translatable via smaller units than words, this technique aims at encoding rare and unknown words as sequences of subwords units. To achieve this, the symbol vocabulary is initialized with the character vocabulary, and each word is represented as a sequence of characters—plus a special end-of-word symbol. After that, all symbol pairs are iteratively counted. Then, each occurrence of the most frequent pair (A, B) is replaced with a new symbol AB. This process is repeated as many times as new symbols to create. Once the encoding is learned, BPE is applied to the training corpora to obtain a representation as sequences of subwords units. Then, the SMT system is trained using the encoded corpora. At the end of the process, the generated text—which has been translated into an encoded version of the target language—is decoded.

3. Experiments

In this section, we describe the experiments conducted in order to assess our proposal. We also present the corpora and metrics, and describe the set up of our framework.

3.1. Corpora

To test our proposal, we selected the corpora distributed at the **CLIN2017 Shared Task on Translating Historical Text**[1]:

Bible: A collection of books from different version of the Dutch bible. Mainly, a version from 1637, another from 1657, another from 1888 and another from 2010. All versions are composed by the same books, except from the 2010's version, which is missing the last part of the content.

[1] https://ifarm.nl/clin2017st/

Dutch Literature: A collection of texts from Dutch literary classics from the 17th century. It contains a small development partition and a test partition. The test partition is composed by a collection of texts from a different decade of the 17th century.

The goal of the shared task was to translate historical documents from 17th to 21st century Dutch. However, the translation they were looking for consisted in *replacing all the words that did not occur in a standard lexicon*. Therefore, the aim of the shared task was to update the spelling to 21st century standards, and not to obtain a version of the documents that matches nowadays Dutch.

While the Dutch literature corpus was created with the aim of updating the spelling, the Bible corpus contains the same books in different versions of Dutch (i.e., the Dutch spoken in the moment they were written). This last corpus was given as a training material for the shared task, and contains a test partition for translating a document from 17th to 19th century Dutch. Therefore, we decided to use this corpus to asses our proposal—considering 19th century Dutch as modern Dutch. Additionally, we make use of the Dutch literature corpus to evaluate our method in the context of only updating the spelling. Table 1 shows the corpora statistics.

		Bible				Dutch literature
		1637–1888	*1637–2010*	*1657–1888*	*1657–2010*	*17th–21st century*
Train	**|S|**	37K	31K	37K	31K	-
	|T|	927/917K	927/786K	934/917K	934/786K	-
	|V|	57/45K	57/37K	57/45K	57/45K	-
Development	**|S|**	-	-	-	-	13
	|T|	-	-	-	-	1260/1265
	|V|	-	-	-	-	505/474
Test	**|S|**	5000	-	-	-	489
	|T|	148/141K	-	-	-	12/12K
	|V|	11/9K	-	-	-	3530/3176

*Table 1. Corpora statistics. **|S|** stands for number of sentences, **|T|** for number of tokens and **|V|** for size of the vocabulary. K denotes thousand. The bible corpus is extracted from different versions of the Dutch bible. The Dutch literature corpus is composed by a collection of texts extracted from various Dutch literary classics.*

For the task of modernizing historical documents, we limited the training corpora to the 1637–1888 partition of the Bible corpus (since we are considering 19th century Dutch to be the contemporary version of Dutch). Additionally, to enrich the language model, we collected all 19th century works available at the *Digitale Bibliotheek voor de Nederlandse letteren*[2] and added them to the training data.

[2]http://dbnl.nl/

The 1637–2010 and 1657–2010 partitions of the Bible corpus were proportionated with a warning about the quality of the 2010's version. Thefore, for the task of updating the spelling to 21st century Dutch, instead of limiting the training data to these two partitions we made use of all the available partitions. More precisely, we selected those sentences from the training corpora which were better suited for the task (see Section 2.2). Additionally, in a similar way as in the previous task, we collected all 21st century works from the *Digitale Bibliotheek voor de Nederlandse letteren* to enrich the language model.

3.2. Metrics

In order to assess our proposal, we made use of the following well know metrics:

BiLingual Evaluation Understudy (BLEU) (Papineni et al., 2002): computes the geometric average of the modified n-gram precision, multiplied by a brevity factor that penalizes short sentences.

Translation Error Rate (TER) (Snover et al., 2006): computes the number of word edit operations (insertion, substitution, deletion and swapping), normalized by the number of words in the final translation.

3.3. SMT Systems

SMT systems were trained with the `Moses` toolkit (Koehn et al., 2007), following the standard procedure: optimizing the weights of the log-lineal model with MERT (Och, 2003), and estimating a 5-gram language model, smoothed with the improved Kneser-Ney method (Chen and Goodman, 1996), with `SRILM` (Stolcke, 2002). Moreover, since source and target have similar linguistic structures—the target language is an evolution of the source language—we used monotonous reordering. The corpora were lowercased and tokenized using the standard scripts, and the translated text was truecased with `Moses`' truecaser.

The systems in which BPE was used (see Section 2.3) were trained in the same way. The only difference is that the corpora were previously encoded using BPE, and the translated text was decoded afterwards. BPE encoding was learned and applied using the scripts kindly provided by Sennrich et al. (2016). In learning the encoding, the default values for the number of symbols to create and the minimum frequency to create a new symbol were used.

4. Results

This section presents the results of the experiments conducted in order to assess our proposal. We first evaluate our method for modernizing a historical document using the Bible corpus (see Section 3.1) and, then, we additionally test our method in a context in which only the spelling needs to be updated, using the Dutch literature

corpus. Confidence intervals ($p = 0.05$) were computed for all metrics by means of bootstrap resampling (Koehn, 2004).

4.1. Document Modernization

The first task consisted in applying our proposed method for obtaining a version of a historical document in modern language. Table 2 shows the results obtained in this task. As a baseline, we compare the quality of the original document with respect to its modern version. Additionally, the shared task from which the corpus was obtained (see Section 3.1) provided an extra baseline. This second baseline was generated by applying some unspecified translation rules to the original document.

System	BLEU	TER
Baseline	13.5 ± 0.3	57.0 ± 0.3
Baseline_2	50.8 ± 0.4	26.5 ± 0.3
SMT	64.8 ± 0.4	17.0 ± 0.3
+ LM_2	65.1 ± 0.4	17.3 ± 0.3
SMT_{BPE}	64.8 ± 0.4	17.4 ± 0.3
+ LM_2	$\mathbf{66.7 \pm 0.4}$	$\mathbf{16.2 \pm 0.3}$

Table 2. Experimental results for the document modernization task using the Bible corpus. Baseline system corresponds to considering the original document as the modernized document. Baseline_2 was proportionated as part of the shared task and was obtained by applying certain translation rules to the original document. SMT is the standard SMT system. SMT + LM_2 is the SMT system trained with an additional language model. SMT_{BPE} is the standard system in which the training corpus has been encoded using BPE. SMT_{BPE} + LM_2 is the system in which the training corpus has been encoded using BPE and an additional language model is used during the training process. Best results are denoted in ***bold****.*

The proposed standard SMT system greatly improves this first baseline, both in terms of BLEU (around 51 points of improvement) and TER (around 40 points of improvement). Moreover, it also improves significantly the second baseline (around 14 points of BLEU and 9 points of TER). Finally, enriching the system by adding an additional language model does not significantly improve the results of the standard system. Most likely, this is due to the training data being very similar to the document we are modernizing (they all belong to the same version of the Bible). For this reason, the language model obtained from the training data is robust enough to do the modernization without additional help.

Encoding the training corpora with BPE (see Section 2.3) to reduce the number of unknown words brings similar results to just using the standard system. Once more,

the similarity between training and test reduces the vocabulary problem. Nonetheless, combining the use of BPE with the additional language model obtains a significant improve over the standard system (around 2 points of BLEU and 1 points of TER). Most likely, this is due to BPE taking profit from the additional language model to better learn how to generate subword units.

4.2. Standard Spelling

The second task consisted in updating the spelling of a historical document to match current standards. Although our proposed method aims at obtaining a version of the document with modern language, we wanted to assess how the method would work in this context. Similarly as in the previous task, we considered as baseline the quality of the original document in comparison to the document with the updated spelling.

System	Original corpora		Data selection	
	BLEU	TER	BLEU	TER
Baseline	29.9 ± 1.8	32.4 ± 1.1	-	-
SMT	48.1 ± 1.8	22.0 ± 0.8	**49.9 ± 1.8**	**20.2 ± 0.8**
+ LM_2	49.4 ± 1.8	21.2 ± 0.8	49.8 ± 1.8	20.9 ± 0.8
SMT_{BPE}	48.6 ± 1.6	24.2 ± 0.9	49.2 ± 1.6	23.7 ± 0.8
+ LM_2	47.9 ± 1.7	25.5 ± 0.9	49.9 ± 1.7	23.7 ± 0.8

Table 3. Experimental results for the standard spelling task using the Dutch literature corpus. Baseline system correspond to considering the original document as the document with the updated spelling. SMT is the standard SMT system. SMT + LM_2 is the SMT system trained with an additional language model. SMT_{BPE} is the standard system in which the training corpus has been encoded using BPE. SMT_{BPE} + LM_2 is the system in which the training corpus has been encoded using BPE and an additional language model is used during the training process. Best results are denoted in ***bold****.*

Our standard SMT system greatly improves the baseline, obtaining increases of around 18 points of BLEU and 10 points of TER. Similarly as in the previous task, enriching the system with an additional language model does not obtain significant improvements. This is probably due to the nature of the task: only non-standard words should be change, independently of semantic correctness. The language model, however, has only been trained with sentences which are semantically correct.

In this case, encoding the training corpus with BPE (see Section 2.3) to mitigate the number of unknown words does not improve results. Not even when using an additional language model. Most likely, the nature of the task makes more difficult for BPE to learn to create subword units.

When using data selection to create a new training corpus formed only by those sentences which are more similar to the document (see Section 2.2), we obtain a significant improve in terms of TER. Results for BLEU, however, are not significantly different to training with all the available corpora. Similarly to what happened before, enriching the system with an additional language model does not obtain significant improvements.

As in the previous case, encoding the training corpus with BPE to reduce the number of unknown words does not improve results. BLEU values are more or less within the same confidence interval, while TER significantly increases around 3 points. Enriching the system with an additional language model also obtains similar results.

Finally, in comparison to the results of the shared task from which this corpus was obtained (see Section 3.1), our approach would have placed 6^{th} out of 9. It is worth noting, however, that while the aim of the shared task was to update the spelling to modern standards without aiming for semantic correctness, our method aimed at obtaining modern semantic, lexicon and grammar.

5. Conclusions and Future Work

In this work, we have presented a method, based on SMT, to translate a historical document to a modern version of its original language. With this method, we aim at increasing the accessibility of historical documents to a broader audience as well as helping in the preservation of the cultural heritage.

Experimental results show that the proposed method significantly increases the quality of the document—with respect to the modern language. However, due to the lack of available corpora, we tested our proposal on a corpus in which the training data is very similar to the document to modernize. This is not often the case with historical documents. Therefore, we should test our method in a framework in which the document to translate has few similarities with the training data.

We also proposed two alternatives for solving two common problems in SMT which also affect to the modernization task. The first of these alternatives, to find training data as similar to the document as possible, was not tested due to the training data already being similar to the document. The second alternative, to tackle rare and unseen words, significantly improves the results achieved by the basic method.

Additionally to the modernization of historical documents, we have tested our method for updating the spelling of a historical document according to modern standards. Experimental results show that our proposal succeeds at standardizing the spelling. However, when comparing to other approaches to this problem, our method still needs some improvements. Nonetheless, this task searches for updating the spelling without aiming for semantic correctness, while our proposal aims at obtaining a modern version of the language—including its spelling and semantic.

We also tested the previously mentioned alternatives. Using data selection techniques to find training data as similar as possible to the document significantly im-

proves results. However, due to the nature of the task, the second alternative does not improve results.

As a future work, besides obtaining more corpora to being able to work in a more common framework, we want to assess our proposal with historical manuscripts to see how it behaves with the additional difficulties inherent in the manuscripts. Additionally, it would be interesting to use our method to generate the paleographic version of a transliterated transcript.

Acknowledgements

The research leading to these results has received funding from the Ministerio de Economía y Competitividad (MINECO) under project CoMUN-HaT (grant agreement TIN2015-70924-C2-1-R), and Generalitat Valenciana under project ALMAMATER (grant agreement PROMETEOII/2014/030).

Bibliography

Bahdanau, Dzmitry, Kyunghyun Cho, and Yoshua Bengio. Neural Machine Translation by Jointly Learning to Align and Translate. In *Proceedings of the International Conference on Learning Representations (arXiv:1409.0473)*, 2015.

Brown, Peter F., Vincent J. Della Pietra, Stephen A. Della Pietra, and Robert L. Mercer. The Mathematics of Statistical Machine Translation: Parameter Estimation. *Computational Linguistics*, 19(2):263–311, 1993.

Chen, Stanley F. and Joshua Goodman. An Empirical Study of Smoothing Techniques for Language Modeling. In *Proceedings of the Annual Meeting on Association for Computational Linguistics*, pages 310–318, 1996.

Crowther, John. *No Fear Shakespeare: Sonnets*. SparkNotes, 2004.

F. Jehle, Fred. *Works of Miguel de Cervantes in Old- and Modern-spelling*. Indiana University Purdue University Fort Wayne, 2001.

Gascó, Guillem, Martha-Alicia Rocha, Germán Sanchis-Trilles, Jesús Andrés-Ferrer, and Francisco Casacuberta. Does more data always yield better translations? In *Proccendings of the European Chapter of the Association for Computational Linguistics*, pages 152–161, 2012.

Koehn, Philipp. Statistical Significance Tests for Machine Translation Evaluation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 388–395, 2004.

Koehn, Philipp. *Statistical Machine Translation*. Cambridge University Press, 2010.

Koehn, Philipp, Franz Josef Och, and Daniel Marcu. Statistical Phrase-Based Translation. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology*, pages 48–54, 2003.

Koehn, Philipp, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondřej Bojar, Alexandra Constantin, and Evan Herbst. Moses: Open Source Toolkit for Statistical

Machine Translation. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, pages 177–180, 2007.

Och, Franz Josef. Minimum Error Rate Training in Statistical Machine Translation. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, pages 160–167, 2003.

Och, Franz Josef and Hermann Ney. Discriminative Training and Maximum Entropy Models for Statistical Machine Translation. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, pages 295–302, 2002.

Papineni, Kishore, Salim Roukos, Todd Ward, and Wei-Jing Zhu. BLEU: a Method for Automatic Evaluation of Machine Translation. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, pages 311–318, 2002.

Sennrich, Rico, Barry Haddow, and Alexandra Birch. Neural Machine Translation of Rare Words with Subword Units. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, pages 1715–1725, 2016.

Snover, Matthew, Bonnie Dorr, Richard Schwartz, Linnea Micciulla, and John Makhoul. A Study of Translation Edit Rate with Targeted Human Annotation. In *Proceedings of the Association for Machine Translation in the Americas*, pages 223–231, 2006.

Stolcke, Andreas. SRILM - An extensible language modeling toolkit. In *Proceedings of the International Conference on Spoken Language Processing*, pages 257–286, 2002.

Sutskever, Ilya, Oriol Vinyals, and Quoc V Le. Sequence to Sequence Learning with Neural Networks, 2014.

Villegas, Mauricio, Alejandro H. Toselli, Verónica Romero, and Enrique Vidal. Exploiting Existing Modern Transcripts for Historical Handwritten Text Recognition. In *Proceedings of the International Conference on Frontiers in Handwriting Recognition*, pages 66–71, 2016.

Zens, Richard, Franz Josef Och, and Hermann Ney. Phrase-Based Statistical Machine Translation. In *Proceedings of the Annual German Conference on Advances in Artificial Intelligence*, volume 2479, pages 18–32, 2002.

Address for correspondence:
Miguel Domingo
midobal@prhlt.upv.es
Universitat Politècnica de València
PRHLT Research Center
Camino de Vera s/n, 46022 Valencia, Spain

PBML

The Prague Bulletin of Mathematical Linguistics
NUMBER 108 JUNE 2017 307–318

Comparative Quality Estimation for Machine Translation Observations on Machine Learning and Features

Eleftherios Avramidis

German Research Center for Artificial Intelligence (DFKI Berlin), Language Technology Lab

Abstract

A deeper analysis on Comparative Quality Estimation is presented by extending the state-of-the-art methods with adequacy and grammatical features from other Quality Estimation tasks. The previously used linear method, unable to cope with the augmented features, is replaced with a boosting classifier assisted by feature selection. The methods indicated show improved performance for 6 language pairs, when applied on the output from MT systems developed over 7 years. The improved models compete better with reference-aware metrics.

Notable conclusions are reached through the examination of the contribution of the features in the models, whereas it is possible to identify common MT errors that are captured by the features. Many grammatical/fluency features have a good contribution, few adequacy features have some contribution, whereas source complexity features are of no use. The importance of many fluency and adequacy features is language-specific.

1. Introduction

The need for automatically predicting the quality of Machine Translation (MT) output has lead into the development of Quality Estimation (QE; Specia et al., 2009). Whereas most QE tasks aim at a single judgment, there have been concerns on how confident one can be in quantifying quality. Humans seem to have difficulty in scoring the quality of translations, particularly in defining the distinction between the level of quality each score represents (Callison-Burch et al., 2007). A solution would be to reduce the requirements for the ground truth, by favouring ordinality against cardinality. This can be done by eliciting judgments of relative quality, through direct comparisons between two or more translation items (Duh, 2008). For problems that

 Corresponding author: eleftherios.avramidis@gmail.com
Cite as: Eleftherios Avramidis. Comparative Quality Estimation for Machine Translation Observations on Machine Learning and Features. The Prague Bulletin of Mathematical Linguistics No. 108, 2017, pp. 307–318.
doi: 10.1515/pralin-2017-0029.

require comparisons of performance, it may be beneficial to neglect qualitative observations that are irrelevant to the comparison and may interfere with the decision.

Following this idea, we are focusing on Comparative QE as the automatic process of analyzing two or more translations produced by various MT systems and employing machine learning (ML) to express a judgment about how they compare in terms of quality. Although a considerable amount of research has employed this concept for various applications, such as system combination, statistical MT tuning and evaluation, there has been little analysis of the very concept of Comparative QE per se.

In this paper we attempt to extend the relatively limited state-of-the-art work and investigate the factors that play an important role for the task. In particular we will:

- bring features from other QE tasks to Comparative QE: introduce adequacy features, augment the grammatical ones with CFG rules and position indicators,
- observe whether linear methods in this problem can cope with the amount and the type of the advanced features and suggest instead an ensemble classifier
- improve on previous work regarding the competition with reference-aware metrics, confirming that elaborate features and ML may provide more information about relative translation quality than the comparison with the references,
- show which quality indicators are important for comparing MT outputs by investigating their contribution in the produced models, identify the MT errors that make these features useful for the automatic comparison of the translations,
- use feature selection methods to select an optimal number of features in order to improve the performance of the learning method or to achieve the same performance with a smaller amount of features,
- indicate the importance of grammatical features and confirm that the contribution of specific grammatical features is language-specific
- empirically confirm that source complexity features are not useful for predicting a comparison between automatic translations.

2. Related Work

The concept of Comparative QE, although not explicitly defined, has been used in many MT related tasks. In particular, previous works perform it as they:

(a) predict a continuous score independently for each system output and then they rank the outputs based on their individual score (e.g. Specia et al., 2009),
(b) use binary classification or regression with a cut-off value, to accept/reject a basic system and then back-off to another system without judging it (Quirk, 2004),
(c) use binary classification to compare two systems (Yasuda et al., 2002) or
(d) use an ordinal ranking (Herbrich et al., 1999) to compare an undefined number of systems (Hopkins and May, 2011; Avramidis et al., 2011; Formiga et al., 2013).

In this paper we are going to follow on the latter work. It essentially extends the binary classification (b), with the difference that the underlying classifier is system-agnostic and that it decides on comparisons for all possible pairs. Contrary to the

continuous regression approach, the ordinal model only learns a relative notion of the translation quality, by having quality indicators from all compared outputs.

Formiga et al. (2013) confirm that ordinal regression makes better predictions as compared to ordering MT outputs, based on separate regression models over absolute scores of adequacy. When it comes to learning from ordinal rankings, Avramidis and Popović (2013) set the state-of-the-art performance for German-English, in the frame of a WMT shared task in QE (Bojar et al., 2013),

Previous work has motivated the use of grammatical features focusing in specific structures (eg. Mutton et al., 2007), feature selection was motivated by Specia et al. (2009), whereas an analysis of features was done by Felice and Specia (2012); nevertheless all the above work is limited to non-Comparative QE.

As compared to previous work, here we extend the state-of-the-art on Comparative QE by increasing the human correlation through the use of a Gradient Boosting classifier. We add additional linguistically-informed features inspired from other tasks. We also present a detailed analysis of the contribution of (a) the individual features, (b) the feature selection and (c) the learning methods. Our models exceed all previous experiments in coverage, as they expand into 6 language directions and are learned on outputs from heterogeneous MT systems developed within a period of 7 years.

3. Methods

3.1. Problem definition

This work aims at developing an empirical system which is able to order multiple translation outputs in the same way humans would do. In particular, the system is given one source sentence and several translations which have been produced for this sentence. The goal is to *rank* them, i.e. to order the translations based on their quality after deriving several qualitative criteria over the translations.

We define a ranking $R = \{s, \mathbf{t}, \mathbf{r}\}$ where a source sentence s is associated with a set of translations $\mathbf{t} = (t_1, t_2, \ldots, t_m)$, as t_j is the j-th translation of s and m the number of the translations. Each set of translations $\mathbf{t}$ is associated with a list of ordinal judgments (ranks) $\mathbf{r} = (r_1, r_2, \ldots, r_n)$, where r_j is the judgment on translation t_j, as compared to the other translations in $\mathbf{t}$. This kind of qualitative ordering does not imply any absolute or generic measure of quality. Ranking takes place on a sentence level, which means that the inherent mechanism focuses on only one sentence at a time, considers the available translation options and makes a decision. Any assigned rank has therefore a meaning only for the sentence-in-focus and given the particular alternative translation candidates. Each source sentence $s^{(i)}$ is associated with a set of translations $\mathbf{t}^{(i)} = (t_1^{(i)}, t_2^{(i)}, \ldots, t_m^{(i)})$ where $t_j^{(i)}$ is the j-th translation of the i-th source sentence and m the number of the translations. Each list of translations is associated with a list containing relative judgments (ranks) $\mathbf{r}^{(i)} = (r_1^{(i)}, r_2^{(i)}, \ldots, r_n^{(i)})$ where $r_j^{(i)}$ is the judgment on the j-th translation of the i-th source sentence.

Counts: number of tokens and unknown words, number of occurrences of the target word within the target hypothesis (type/token ratio), number of commas and dots,
Parsing: PCFG parsing for both source and target side: the sentence log-likelihood, the number of n-best trees, the number of VPs in the best parse tree
Source complexity features: average source token length, average number of translations per source word in the sentence, percentage of unigrams/bigrams/trigrams in frequency quartiles 1 (lower frequency words) and 4 (high frequency words) in a corpus of the source language, percentage of source sentence unigrams seen in a corpus
Contrastive scoring: the METEOR score using the competing translations as references

Counts: avg. chars per word, count of nums and of tokens with non-alphabetic characters
Language model: smoothed probability from 3-gram and 5-gram LM, 3-gram perplexity
IBM Model 1: scores on both directions
Contrastive scoring: smoothed BLEU; precision, recall, frag. penalty of METEOR
Unknown words: first and last position of unknown words (absolute and normalized to the length of the sentence), average and standard dev. of the positions of unknown words
Rule-based correction: total errors, comma/parenthesis+space, uppercase sentence start

Table 1. Upper: Features for the baseline feature set. Lower: Features for the augmented feature set, added to the baseline features and the grammatical features of Section 3.2

A *feature vector* is defined as $\mathbf{x}^{(i)} = G(s^{(i)}, \mathbf{t}^{(i)})$ and it is created from every pair of source and its translations $(s^{(i)}, \mathbf{t}^{(i)})$, where $i = 1, 2, \ldots n$. The function G that produces the feature vector given a source and its translations is referred to as *feature generation*. Each feature vector $\mathbf{x}^{(i)}$ derived from the i-th source sentence and the corresponding list of ranks define an *instance* $I^{(i)} = (\mathbf{x}^{(i)}, \mathbf{r}^{(i)})$ and a *training set* of n instances is consequently defined as $T = \{(\mathbf{x}^{(i)}, \mathbf{r}^{(i)})\}_{i=1}^{n}$. A *ranker* is a function which given a feature vector $\mathbf{x}^{(i)}$ produces a list of *predicted* ranks $\hat{\mathbf{r}}^{(i)}$. The goal of the *learning process* is therefore given the training set T to define a ranker that minimizes the total error between the predicted list of ranks and the golden list of ranks: $\sum_{i=1}^{m} \mathcal{E}(\mathbf{r}^{(i)}, \hat{\mathbf{r}}^{(i)})$.

3.2. Feature generation

The **baseline feature set** (upper Table 1) consists of features that had the optimal performance as reported in previous work, i.e. the baseline and the best performing ranking QE features of WMT (Bojar et al., 2013). The **augmented feature set** extends the baseline set with features from non-Comparative QE (lower Table 1). Additionally more fluency features are added, as deemed helpful in the baseline, and adequacy features are introduced, as they were absent. These features are described below:

We count the **node labels of the parse tree**, namely NPs, VPs, PPs, verbs, nouns and for every node label we get the minimum, maximum and average depth/height of its positions in the tree and the average and standard deviation of its position. Every

parse tree is decomposed into **Context-Free Grammar (CFG) rules** and for every rule, we get the number of occurrences and statistics about its height and depth in the tree. For the rules that contain a VP or a verb, two additional features indicate their distance from the beginning and the end of the sentences. This is of particular interest for translations into German, where the position of the VPs in the sentence is important.

A set of **alignment features** is produced as the nodes between the source and the target trees are aligned based on the scores of the lexical IBM-1 model (Zhechev, 2009). For every node alignment, we get the count of the aligned nodes in the sentences, the count of occurrences of the target CFG rules whose heads are aligned to the similar rules in the source, the depth of the source node in the source tree and the distance of the aligned nodes (if related to verbs) from the beginning and the end of the sentence.

This process got all possible alignments of node labels, resulting into 154,657 features. Nevertheless, many of these features are sparse, since they depend on the appearance of grammatical phenomena, so we used some sparsity heuristics resulting into 139 features: the monolingual CFG features including VPs and NPs with more than 20k occurrences (5+5 features), CFG alignment features including VPs with more than 10k occurrences (5) and NPs with more than 30k occurrences (5), CFG position features with more than 24k occurrences (5), rule-based corrections with more than 1k occurrences (4) and from the rest of the features, the ones with more than 51k occurrences (110 features). This selection aims at making the experiments computationally feasible, although there is no evidence that the reduced set is optimal.

3.3. Learning Methods and Evaluation

The ranker performs pairwise classification (Avramidis and Popović, 2013). The baseline uses Logistic Regression with the Newton-Raphson algorithm including Stepwise Feature Set Selection. As an advanced method, after preliminary experiments[1], we chose a Gradient Boosting of 100 decision trees and 100 boosting stages, limiting the maximum depth of the individual estimators to 3 and presorting data in order to find splits faster. Feature selection is done with Recursive Feature Elimination with cross-validation (RFECV) using SVM (Herbrich et al., 1999) with a linear kernel.

The predicted ranking is evaluated based on its correlation with human rankings, using Cross Validation with 10 folds over the entire dataset. The correlation metric is Kendall's tau as per WMT12: ties and cases of equal disagreement are removed from the test sets, whereas predicted ties are counted as discordant pairs, occasionally leading to negative taus.[2] Significance tests are based on the theoretical two-tailed t-

[1] including Decision Trees, Gaussian Naïve Bayes, kNN, LDA, Log. Regression with L2 Regularisation, Adaboost, Bagging, ExtRa Trees, and Random Forest. The boosting was tested with both 50 and 100 trees

[2] The evaluation setup differs from that of Bojar et al. (2013) to allow more robust testing, so here we rerun and evaluate their best methods as our baseline. Under our evaluation setup they result into slightly different scores

test of tau and confidence intervals by bootstrap resampling ($n = 1000$, $\alpha = 0.05$). NDCG is considered as an additional ranking metric (Järvelin and Kekäläinen, 2002).

4. Experiments

The experiments are performed on MT output from WMT annotated with human rankings (WMT2008-2014; e.g. Bojar et al., 2013) for English to German, French, Spanish and vice-versa, but advanced feature engineering is done only for German due to the increased MT errors for this language. A separate model is trained for every language direction. Per language pair, there are about 7k sentences from the news domain translated by about 100 systems. Translations of each sentence are grouped randomly into batches of 5 and ranked by various annotators. This provides 13k-25k batches, resulting into 64k to 100k pairwise comparisons. The vast majority of the systems are phrase-based and variations, whereas only 5% are rule-based.

Feature generation and learning are run with QUALITATIVE (Avramidis, 2016), PCFG is run with the Berkeley Parser pre-trained on the TIGER, TueBaD/Z, AncoRa and FTB treebanks (Petrov et al., 2006) and rule-based correction is run with LANGUAGETOOL[3].

4.1. Ranking performance

In this experiment (a) we test whether the predicted rankings have any correlation with human rankings, (b) we compare the augmented ranking mechanism against the baseline and a random ranking and (c) we compare the augmented ranking mechanism against state-of-the-art reference-aware metrics. The metrics compared are: BLEU with sentence-level smoothing (Papineni et al., 2001), METEOR, (Denkowski and Lavie, 2014), rgbF (Popović, 2012), WER and TER (Snover et al., 2006).

Results The results (Table 2) indicate that (a) the predicted rankings have significant correlation with human rankings with a t-test p-value almost zero, (b) the predicted rankings are significantly better than random ones. The augmented ranking mechanism has achieved improved correlation against the baseline ranking mechanism.

A notable improvement over the baseline is that (c) the augmented ranking mechanism performs significantly better than the state-of-the-art reference-aware automatic metrics on a sentence level for the language pairs involving German, where focused feature engineering took place. It also outperforms other metrics in language pairs where the feature engineering from other language pairs was adopted, apart from one metric, METEOR, which is on par with the ranking mechanism. This confirms that elaborate features and ML may provide more information about relative translation quality than direct comparison with references.

[3]http://languagetool.org

lang.	basel.	augm.	random	BLEU	METEOR	rgbF	TER	WER
de-en	0.26*	0.28*	-0.14	-0.22 ‡	0.23 ‡	0.16 ‡	-0.02 ‡	0.15 ‡
en-de	0.15*	0.17*	-0.17	-0.42 ‡	0.13 ‡	0.10 ‡	-0.09 ‡	-0.15 ‡
es-en	0.11*	0.22*	-0.18	-0.19 ‡	0.22 ◇	0.16 ‡	-0.02 ‡	0.13 ‡
en-es	0.11*	0.12*	-0.17	-0.21 ‡	0.12 ◇	0.09 ◇	-0.10 ‡	0.08 ‡
fr-en	0.18*	0.19*	-0.18	-0.18 ‡	0.20 ◇	0.15 ‡	-0.02 ‡	0.16 ‡
en-fr	0.20*	0.21*	-0.15	-0.12 ‡	0.18 ◇	0.15 ‡	-0.03 ‡	0.15 ‡

‡: augmented ranking mechanism is significantly better than metric
◇: augmented ranking mechanism is significantly as good as metric
*: correlation with humans is significant, with a measured $p < 4 \cdot 10^{-20}$

Table 2. Basic vs. augmented ranking mechanism with random ranking and automatic metrics, concerning correlation with human judgments (tau) on segment-level

4.2. Observations on the baseline features

Useful conclusions concerning the contributions of various features can be drawn by examining the estimated beta coefficients of the logistic regression model of the baseline. For every coefficient, the null hypothesis of it being equal to zero has been rejected with a χ-test. The sign (positive/negative) of the coefficient indicates whether the feature has a positive or a negative contribution to the selection of the translation by the humans. Also, since the feature values are normalized with their mean and variance, the coefficient may provide indications for the importance of the features on the final decision. Some observations on the beta coefficients (Table 3) are:

Number of unknown words: Although OOVs are not necessarily untranslated words, when two translations of the same source have a different amount of unknown words, it is more likely that the one with the most of them has failed to translate some.

Overall amount of tokens: Statistical systems often omit the translation of some source words. This occurs when words suggested by the translation model reduce dramatically the overall score during the decoding process. Manual evaluation indicates that this occurs with long-distance re-ordering of German verbs, not scored properly by the language model. Therefore, when a translation has less words than its competitor, it may be the case that a useful word was omitted. *Additional words* also occur as a translation error, e.g. when phrases chosen during the decoding of a phrase-based system overlap partially. A special case of this, when the same word is repeated in the generated translation (type/token ratio) is given a negative coefficient.

Contrastive scoring: When more than one systems perform the same translation, they often convey more correct information collectively than each of them. Therefore, a system output that agrees more with the majority of the other systems is more likely to be preferred as the best translation.

The number of verb phrases (VPs) is connected with the fluency, as a result of the parser having tried to analyze the sentence and identify the VPs. Among translation errors, it is more likely that a VP is not formed properly, than having superfluous VPs formed by mistake. Therefore, it is observed that if a translation has more VPs than its competitor, it is more likely to be chosen. Similarly, when the parser analyses a translation, it creates **n-best lists with trees** with all possible grammatical analyses. The size of the list can indicate how ambiguous the parse is and therefore a translation with fewer n-best trees is more preferable for comparing translations. The **parse log-likelihood** also has a positive contribution, as an indication of grammaticality.

Punctuation count indicates that translation systems often make mistakes with punctuation and it is more likely to select a translation when it has fewer commas, or when it has more dots. Systems erroneously create too many commas or omit dots.

Finally, there is little explanation of the low, albeit negative contribution of the **tri-gram LM probability**, since one would expect that a higher probability would be preferable. One could assume that this is interacting with some other features, e.g. to favour grammatical features over the LM, or that some MT systems overvalue the LM score, which is also the reason for the omission of German verbs, mentioned earlier.

There can also be conclusions about the features which were assigned a zero coefficient. Using this, we can see that out of the **non-comparative QE features** only the punctuation features, the type/token ratio and the tri-gram probability helped, added to the target sentence length, which already existed as a feature. **Source complexity features** have been also assigned zero coefficients, so we can confirm that they play no role in the comparison between translations and that they do not introduce any useful knowledge about the *relative ability* of the systems to translate these sentences.

4.3. Machine Learning method and Feature Selection

Here, we investigate (a) the effect of adding the augmented feature set on the baseline model with Logistic Regression (b) the possibility to reduce the amount of features by performing Feature Selection (c) the improvements by using an ensemble instead of a linear classifier and finally (d) the effect of adding/removing features.

Feature Selection is applied only for German-English and English-German on a sub-set of the full-dataset. Since RFECV does not scale well, it is run on a stratified sample resulting into the 2.5% of the original sentences of a single fold for German-English and the 5% for English-German[4]. The selected feature set was used to train and evaluate the ranking model with 10-folded cross-validation, as above.

Results The results of using RFECV and Gradient Boosting can be seen in Table 4. Simply adding the augmented feature set on the baseline model with the Logistic

[4]Although this small sample is not guaranteed to be enough for feature selection, we will show that it is enough for reducing the feature size without harming the overall performance

feature name (target sentence)	β
number of unknown words	-0.58
number of tokens	0.50
contrastive METEOR	0.29
number of VPs	0.17
number of n-best trees	-0.17
type/token radio	-0.14
number of commas	-0.11
sentence parse log-likelihood	0.08
3-gram probability	-0.05
number of dots	0.04
…other features of Table 1	0.00

Table 3. Logistic Regression coefficients for the baseline, in descending order of absolute values

lang.	method	set	tau	NDCG
de-en	LogReg	basic	0.261	0.730
		full	0.110	0.680
		RFECV	0.181	0.716
	GradBoost	basic	0.265	0.736
		full	**0.280**	**0.742**
		RFECV	**0.276**	**0.739**
en-de	LogReg	basic	0.151	0.725
		full	0.034	0.703
		RFECV	0.020	0.696
	GradBoost	basic	0.138	0.723
		full	**0.170**	**0.733**
		RFECV	**0.174**	**0.731**

Table 4. Performance of the basic, the full feature set and the result of the RFECV with Logistic Regression and Gradient Boosting

Regression causes a significant drop, indicating that this method is not capable of handling such an amount and type of features, possibly because it cannot handle non-linear indicators. RFECV improves significantly the performance of Logistic Regression on the augmented feature set for German-English, but it still does not reach the performance of the same algorithm with the baseline set. For English-German, both the full set and the RFECV lead to almost zero correlation.

When it comes to using the advanced feature set, Gradient Boosting achieves significantly better performance than Logistic Regression. Using RFECV to reduce the full set has a negligible effect on the model trained with Gradient Boosting. Although the usage of RFECV did not improve the performance, it is interesting that the number of features (139) was reduced to less than the half, but the correlation remained the same. Reducing the amount of features can be of interest in an application environment, since it also reduces the computation. The above observation can also be seen in Figure 1, which depicts the increase in the classification quality, as features are added in the model. The optimal set for German-English contains 41 features, whereas the English-German one contains 56 features. The performance reaches already high levels with an amount of about 25 features and after a few fluctuations it enters a plateau where more features do not have a significant implication to the model.

4.4. Observations on the advanced features

Whereas 139 features were passed to Feature Selection, the latter favoured a significantly smaller number of features, nevertheless leading to the same performance. We can use the results of the selection to (a) identify important differences between the baseline and the augmented set and (b) compare between the two language directions. Some observations on the selection (Table 5) are:

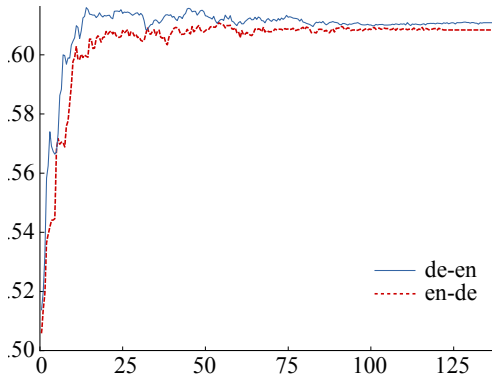


Figure 1. Number of features selected by RFECV vs. classification accuracy

language pair	de-en	en-de
Tree nodes		
nouns (count)		+
nouns (average position)	+	+
nouns (std of positions)	+	
NPs (count)	+	+
NPs (average position)	+	
NPs (std of positions)	+	+
VPs (std of positions)		+
VPs (avg, max tree height)	+	+
PPs (count, std of positions)		+
CFG rules		
NP→DT-NN (count)	+	
PP→IN-NP (count)	+	
VP→TO-VP (count)	+	
S→VP (position from end)		+
VP→VP (position from end)		+
Aligned CFG rules and nodes		
S→NP-VP (count/depth/pos.)	+	
NP (count)		+

Table 5. Grammatical features selected by RFECV

Augmented vs. baseline feature set: Although source complexity features were ruled out during Logistic Regression, Feature Selection for the augmented set favours few features that do include source information through the alignment of grammatical structures between source and target. For German-English, these are the statistics of the alignment of the simplest CFG sentence rule (S→NP-VP), whereas for English-German the aligned NPs. The contribution of these alignments is reasonable, given their grammatical operation and density. Additionally, this indicates that although simple features based on source information may be of little use, targeted features that capture translation adequacy on particular structures can still be of high relevance for comparing translations. Finally, it is worth noting that single features from the basic ranking mechanism have been replaced by a multitude of more specific features with similar functionality (e.g. the count of VPs has been replaced with counts of VPs within more fine-grained rules). This can be attributed to the advanced learning method which can handle better a larger amount of partially overlapping features.

Comparison between language pairs: Language-specific differences are shown by the grammatical that were automatically selected. The ones selected for English-German indicate the importance of the *position of the VPs and the PPs* in the sentence, obviously justified by the German positional requirements. This is in contrast to German-English, which get no features referring to the position of VPs or PPs. For the direction into English we can note the CFG rules that relate with grammatical phenomena which may be often mistranslated, such as the NPs with a determiner and a noun, the VPs containing a gerund and the PPs with the preposition “in”.

5. Conclusion and further work

We have built on top of previous state-of-the-art work on Comparative Quality Estimation by introducing adequacy features and severely augmenting the grammatical/fluency features with CFG rules and position indicators. Logistic Regression used previously cannot handle properly the advanced features, possibly because they are non-linearly separable, so we introduced a Gradient Boosting classifier that could cope better with the problem and improve the performance of the ranking.

We tested the methods with 6 language directions by training on the output of systems spanning 7 years of development. The models can compete better against state-of-the-art reference-aware metrics on the segment-level, particularly when language-specific feature engineering took place, confirming previous observations that elaborate features with ML can compete direct scoring against references. The contribution of grammatical features is notable and it is possible to identify common MT errors that justify the empirically estimated contribution of particular indicators. The use of most grammatical features strongly depends on the target language, e.g. position of VPs is important for German. The majority of the features indicate fluency, few features indicate adequacy, whereas source complexity features are of no importance.

Although these experiments are based on empirical analysis on the output of a broad set of MT systems, we are aware that we are missing some significant representation of Neural MT, which has changed considerably the quality and the error types of MT. Investigations to this direction will be inevitably part of further work.

Acknowledgment This work has received funding from the European Union's Horizon 2020 research and innovation program under grant agreement N° 645452 (QT21)

Bibliography

Avramidis, Eleftherios. Qualitative: Python Tool for MT Quality Estimation Supporting Server Mode and Hybrid MT. *The Prague Bulletin of Mathematical Linguistics*, 106:147–158, 2016.

Avramidis, Eleftherios and Maja Popović. Machine learning methods for comparative and time-oriented Quality Estimation of Machine Translation output. In *Proceedings of the Eighth Workshop on Statistical Machine Translation*, pages 329–336, Sofia, Bulgaria, 2013.

Avramidis, Eleftherios, Maja Popović, David Vilar, and Aljoscha Burchardt. Evaluate with Confidence Estimation: Machine ranking of translation outputs using grammatical features. In *Proceedings of WMT*, pages 65–70, Edinburgh, Scotland, 2011.

Bojar, Ondřej, Christian Buck, Chris Callison-Burch, Christian Federmann, Barry Haddow, Philipp Koehn, Christof Monz, Matt Post, Radu Soricut, and Lucia Specia. Findings of the 2013 Workshop on Statistical Machine Translation. In *Proceedings of the Eighth Workshop on Statistical Machine Translation*, pages 12–58, Sofia, Bulgaria, 2013.

Callison-Burch, Chris, Cameron Fordyce, Philipp Koehn, Christof Monz, and Josh Schroeder. (Meta-) evaluation of machine translation. In *Proceedings of the Second Workshop on Statistical Machine Translation*, pages 136–158, Prague, Czech Republic, 2007.

Denkowski, Michael and Alon Lavie. Meteor Universal: Language Specific Translation Evaluation for Any Target Language. In *Proceedings of the Ninth Workshop on Statistical Machine Translation*, pages 376–380, Baltimore, Maryland, USA, 2014.

Duh, Kevin. Ranking vs. regression in machine translation evaluation. *Proceedings of the Third Workshop on Statistical Machine Translation*, pages 191–194, 2008.

Felice, Mariano and Lucia Specia. Linguistic Features for Quality Estimation. In *Proceedings of the 7th Workshop on Statistical Machine Translation*, pages 96–103, Montréal, Canada, 2012.

Formiga, Lluís, Lluís Màrquez, and Jaume Pujantel. Real-life Translation Quality Estimation for MT System Selection. In *Proceedings of MT Summit XIV*, pages 69–76, Nice, France, 2013.

Herbrich, Ralf, Thore Graepel, and Klaus Obermayer. Support Vector Learning for Ordinal Regression. In *International Conference on Artificial Neural Networks*, pages 97–102, 1999.

Hopkins, Mark and Jonathan May. Tuning as ranking. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1352–1362, Edinburgh, Scotland, 2011.

Järvelin, Kalervo and Jaana Kekäläinen. Cumulated gain-based evaluation of IR techniques. *ACM Transactions on Information Systems*, 20(4):422–446, 2002.

Mutton, Andrew, Mark Dras, Stephen Wan, and Robert Dale. GLEU: Automatic Evaluation of Sentence-Level Fluency. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 344–351, 2007.

Papineni, Kishore, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a Method for Automatic Evaluation of Machine Translation. IBM Research Report RC22176(W0109-022), IBM, 2001.

Petrov, Slav, Leon Barrett, Romain Thibaux, and Dan Klein. Learning Accurate, Compact, and Interpretable Tree Annotation. In *Proc. of ACL*, pages 433–440, Sydney, Australia, 2006.

Popović, Maja. rgbF: An Open Source Tool for n-gram Based Automatic Evaluation of Machine Translation Output. *The Prague Bulletin of Mathematical Linguistics*, 98(98):99–108, 2012.

Quirk, Chris. Training a Sentence-Level Machine Translation Confidence Measure. In *Proceedings of LREC2004*, volume 4, pages 825–828, Lisbon, Portugal, 2004.

Snover, Matthew, Bonnie Dorr, Richard Schwartz, Linnea Micciulla, and Ralph Weischedel. A Study of Translation Error Rate with Targeted Human Annotation. In *In Proceedings of the Association for Machine Transaltion in the Americas*, 2006.

Specia, Lucia, M. Turchi, N. Cancedda, M. Dymetman, and N. Cristianini. Estimating the Sentence-Level Quality of Machine Translation Systems. In *13th Annual Meeting of the European Association for Machine Translation*, pages 28–35, Barcelona, Spain., 2009.

Yasuda, Keiji, Fumiaki Sugaya, Toshiyuki Takezawa, Seiichi Yamamoto, and Masuzo Yanagida. Automatic machine translation selection scheme to output the best result. In *Proceedings of LREC2002*, pages 525–528, Las Palmas, Spain, 2002.

Zhechev, Ventsislav. Unsupervised Generation of Parallel Treebank through Sub-Tree Alignment. *Prague Bulletin of Mathematical Linguistics*, 91:89–98, 2009.

Address for correspondence:
Eleftherios Avramidis
eleftherios.avramidis@gmail.com
Alt Moabit 91c, 10559 Berlin, Germany

PBML

The Prague Bulletin of Mathematical Linguistics
NUMBER 108 JUNE 2017 319–329

Finite-State Back-Transliteration for Marathi

Vinit Ravishankar[a b]

[a] University of Malta, Faculty of Information and Communication Technology, Malta
[b] Charles University in Prague, Faculty of Mathematics and Physics, Institute of Formal and Applied Linguistics, Czech Republic

Abstract

In this paper, we describe the creation of an open-source, finite-state based system for back-transliteration of Latin text in the Indian language Marathi. We outline the advantages of our system and compare it to other existing systems, evaluate its recall, and evaluate the coverage of an open-source morphological analyser on our back-transliterated corpus.

1. Introduction

Numerous transliteration standards that transliterate Indian languages from their various native scripts into the Latin script have existed for centuries, such as the Hunterian standard, or the International Alphabet of Sanskrit Transliteration (IAST). These standards are applied consistently within academic or formal contexts, where transliteration is necessary.

Despite the growth of Internet penetration in India, adoption of input method editors (IMEs) for Indian languages has been relatively slow. Monojit (2011) have provided a description of the potential challenges involved in creating IMEs with Latin-based keyboards. To compensate for this absence, there has been a tendency to represent Indian languages with the English variant of the Latin script on social media, forums, and over private messaging protocols like text messages or internet relay chat (IRC). Despite the existence of numerous formal transliteration standards, there is a strong tendency towards the use of an unofficial, "organic" transliteration standard, informally dubbed *Romanagari* (for languages that use the Devanagari script). Romanagari is largely based on the English variant of the Latin script, with no diacritics. (1) is an example of a Romanagari sentence, along with the formal ISO 15919 and Devanagari equivalents.

 Corresponding author: vinit.ravishankar@gmail.com
Cite as: Vinit Ravishankar. Finite-State Back-Transliteration for Marathi. The Prague Bulletin of Mathematical Linguistics No. 108, 2017, pp. 319–329. doi: 10.1515/pralin-2017-0030.

(1) mi tyanna marathi shikvaycho
mī tyāṁnā marāṭhī śikavāyacō
मी त्यांना मराठी शिकवायचो

"I used to teach them Marathi"

Being able to convert this text into Devanagari is fairly essential for any further processing, like machine translation, to even be attempted.

In this paper, we describe the implementation of a finite-state transducer (FST) based system to "back-transliterate" the southern Indo-Aryan language Marathi; i.e. to transliterate Romanagari Marathi to formal Devanagari. Whilst there has been no significant research on back-transliterating Marathi, we compare our results to back-transliteration results of other systems designed for Hindi, Bengali and Gujarati. We also outline the advantages of such a system over more statistical ones.

In section 2, we describe the language Marathi, including relevant grammatical details. Section 3 is a literature review of prior work in this domain. Section 4 describes the frameworks we used for implementing our system, and section 5 describes our methodology. Section 6 describes several interesting challenges we faced, and our (potential) solutions. In section 7, we describe our corpus, and our evaluation, and provide an analysis of the results. Finally, we discuss our results and our system in section 8, and conclude in section 9.

2. Marathi

Marathi, the fourth-most widely spoken language in India, is an Indo-Aryan language primarily spoken in the western Indian state of Maharashtra. Whilst historically, the Modi script was more widespread, modern Marathi is primarily written in the Balbodh script, which is an abugida, similar to Devanagari. Letters can be full vowels or full consonants; consonants are marked with diacritics to indicate associate vowels. The absence of a diacritic indicates a schwa, although not universally: Marathi, similar to other Indo-Aryan languages, displays the schwa deletion phenomenon (Choudhury et al., 2004), where inherent schwas associated with consonants are sometimes suppressed. Outside these environements, consonant clusters without schwas are often represented using ligatures, or with a "combining" diacritic (◌्).

Balbodh is very similar to Devanagari, apart from the addition of the retroflex lateral approximate (ळ), and an additional diacritic for consonant clusters beginning with an alveolar tap/trill in syllable onsets. Over the course of this paper, therefore, we refer to the script as Devanagari.

Grammatically, Marathi is more agglutinating than many other Indo-Aryan languages, likely owing to Maharashtra's geographical proximity to the Dravidian lan-

guage family: postpositions and cases are often orthographically joined to their heads, and enclitics are common. For instance, compare (2) and (3):

(2) Hindi:

baiṭh-n-ē vālē kō hī
sit-GER-OBL AGT to FOC

(3) Marathi:

bas-ṇār-yā-lā-c
sit-AGT-OBL-DAT-FOC

'To the person that is sitting *(and no one else)*'

3. Prior work

Research on south Asian[1] languages within the context of social media is quite abundant, with several studies focusing on code-switching, a fairly common phenomenon in the subcontinent. Transliteration — or, more accurately, *back*-transliteration — has been less of a research focus. A shared task that involved, amongst other challenges, back-transliteration of Hindi, Gujarati and Bengali was run in 2013 (Roy et al., 2013). The best-performing system (Gella et al., 2013) attempted to back-transliterate test using multi-view hashing.

Outside the south Asian context, there has been significant research on back transliteration. Knight and Graehl (1998) present an algorithm using weighted finite-state transducers, applied to Japanese; Kang and Choi (2000) present a decision tree-based system for Korean. Most of these systems, however, describe back-transliteration *to* the Roman script, rather than away from it.

There has also been some research (albeit not significant amounts) on the actual utility of using Romanagari: Rao et al. (2013) attempted to quantify the cognitive load of processing Romanagari Hindi and concluded that it was significantly higher than the load of processing both Devanagari Hindi and English. This has not, however, hindered the proliferation of Romanagari over social media.

4. Implementation

4.1. *hfst*

The Helsinki Finite-State Technology (*hfst*) library (Lindén et al., 2011) is a front-end for various open source finite-state library back-ends. It allows for data exchange

[1] "South Asian" in this context refers to languages spoken in the Indian subcontinent, including India, Pakistan, Bangladesh and Sri Lanka

between finite state tools implemented in multiple different formalisms; relevant to us, it covers the Xerox LexC and TwolC formalisms (Lindén et al., 2009). We used *hfst* to implement two-level rules; in our approach, this helped eliminate several problematic transcriptions that arose after mere orthographic transfer. Whilst traditionally used for morphological analysis, we view our problem in a very similar fashion: we obtain a set of multiple back-transliterations ("analysis") from which we choose the appropriate one ("disambiguation").

4.2. *lttoolbox*

For actual morphological analysis, we use the *lttoolbox* formalism, a morphological analysis framework used within the open-source machine translation framework, Apertium (Forcada et al., 2011). There exists an *lttoolbox*-based morphological analyser for Marathi[2], with a coverage of 80% on the Marathi Wikipedia. We measure the coverage of this analyser on the Devanagari generated by our system. Morphological analysis is an important prerequisite to many NLP tasks, including rule-based machine translation; measuring coverage of an open-source analyser is, therefore, a useful metric.

5. Methodology

As Romanagari → Devangari back-transliteration can be considered a many-to-many mapping, purely finite-state methods are not sufficient for transliteration, as our *hfst* output is a set of possible transliterations. To quote Knight and Graehl (1998), however, back-transliteration is less "forgiving" than transliteration: there can only be one correct equivalent to a word. We therefore compare three further "filters" to prune these lists: a frequency list, a 2-gram language model, and a morphological analyser.

Our *hfst* rules consisted of two layers: the first being a paradigm-based mapping from Devanagari characters to Latin; this layer relied on finite-state transducers to enforce appropriate transliteration in several domains, based on empirical rules we determined through observation, like inserting combining diacritics when moving from a consonant to another consonant. Our second layer consisted of a series of *twol*-style replace rules, applied synchronically. The most important rule here was to fix schwa deletion, i.e. to remove the combining diacritic wherever necessary. Other rules included, for instance, rules replacing certain digraphs with nasalisation diacritics.

6. Challenges

Whilst building this system, we faced several challenges that were rather interesting from a linguistic perspective.

[2]Available at `https://svn.code.sf.net/p/apertium/svn/languages/apertium-mar/`

6.1. Copular cliticisation

Spoken Marathi displays significant cliticisation of the copular verb असणे *asṇe* ("to be") in the present tense. These clitics, interestingly, carry more inflectional information than the formal copular verb would. For instance, contrast (4) and (5):

(4) tī basat āhē
3FSG sit-PTCP COP.3SG
"She is sitting"

(5) tī bast=iyē
3FSG sit-PTCP=COP.3FSG
"She is sitting"

This phenomenon is represented orthographically in Romanagari, where it is significantly more widespread than non-cliticised copulas are. Whether this cliticisation is valid from a prescriptive perspective in *formal* Devanagari or not is debatable; most formal grammars, such as Dhongade and Wali (2009) do not address this question, despite providing glosses (in Latin transcription) that include cliticised copulas. A brief analysis of several corpora with formal language seems to indicate that these forms are extremely infrequent; therefore, our default solution is to separate the copula from the participle. Our system does, however, allow this separation to be suppressed, based on relevant command-line parameters.

6.2. Word-final *a*

Unlike Hindi, Marathi does not always suppress word-final schwas, particularly for words with Sanskrit etymologies. Schwas are represented with the letter *a* in Romanagari. Further, masculine agreement for verbs and adjectives, often represented by the vowel आ (ā), is also represented with the letter *a*. Finally, neuter agreement - whilst represented with the letter *e* in formal contexts - is often reduced to a schwa in both spoken Marathi, marked with a nasalisation diacritic in Devanagari, and represented with the letter *a* in Romanagari. This leads, essentially, to a three-way back-transliteration ambiguity with word-final *a*s. This sort of ambiguity is impossible for a frequency-list based model to deal with; theoretically, our bigram model ought to be able to fix the agreement issues.

7. Evaluation

A serious problem with evaluating our system was the complete lack of corpora; previous shared tasks on similar themes did not include corpora in Marathi. To fix this, we created our own corpus for evaluating our analyser: a combination of three "mini" corpora (described in table 1), including:

1. Sections of the Marathi Wikipedia transliterated to Romanagari by three annotators[3]
2. Romanagari Twitter feeds (primarily viral "memes"), manually transliterated to Devanagari
3. Romanagari lyrics to Marathi songs, available on the internet, along with their formal Devanagari equivalents

Corpus	Tokens	Types	Letters
Wikipedia	666	450	12,922
Twitter	440	307	6,080
Lyrics	352	180	4,914
Total	1458	889	23,916

Table 1. Corpus statistics

We also generated our frequency list and language models using a Marathi corpus provided by the university, IIT Bombay[4] (3.8m tokens). We did not use Wikipedia to generate language models, as part of our evaluation was on Wikipedia-based text. We used the open-source *kenlm* (Heafield, 2011) to generate our bigram model.

Having assembled our corpus, we proceeded to evaluate our systems on it. Whilst Gella et al. (2013) evaluated their system using the F_1-score, this was not really a valid measure for our system: their precision metric measured the number of correct transliterations their system generated, divided by the number of transliterations their system generated, whilst our system was guaranteed to generate only one transliteration per word. A more valid metric for comparison would be the *recall* of our systems, which measured the number of correct transliterations generated, divided by the number of reference transliterations. We used this measure, along with the mean ambiguity: the mean number of candidates per word, generated by *hfst* before filtering.

7.1. Quantitative

We manually tokenised and lower-cased all our text, both Romanagari and Devanagari. Punctuation was stripped from both. We then ran our *hfst* system on our corpus and post-processed the output. For post-processing, we compared three models: unigram, bigram, and unigram with substring backoff, where we backed off to substrings until we received a match in our frequency list.

[3]All urban Maharashtrian native speakers of standard Marathi

[4]Available at `http://www.cfilt.iitb.ac.in/marathi_Corpus/`

	uni	bi	uni w/ substr
Recall (%)	68.74 (74.91)	68.10 (74.84)	70.37 (76.28)
Coverage (%)	72.72	71.83	72.65
Mean ambiguity	42.42		

Table 2. Evaluation with three filtering methods (figures in parentheses indicate recall on tokens)

Corpus	Recall (%)	Coverage (%)
Wikipedia	75.27	75.35
Twitter	69.58	76.36
Lyrics	64.28	70.82

Table 3. Per-corpus evaluation

For each, we measured recall and morphological analyser coverage. Our results have been described in table 2. A more fine-grained evaluation on each sub-corpus for our unigram with substring backoff model is described in table 3.

An interesting problem that we faced was the ambiguity/recall trade-off on adding or removing certain rules. The most obvious example of this was the schwa addition rule: despite the rule covering a significant chunk of relevant terms, there were exceptions to the rule, and issues with applying the rule at morpheme boundaries. Removing the rule, obviously, resulted in a significant drop in recall. Forcing the system to generate *both* possible forms for every valid context, however, increased the recall of our system. The mean ambiguity of our system, however, simultaneously increased massively. Two other rules include forcing long vowels at word-final positions, and short vowels at word-initial positions, and the inclusion of two non-native vowels used primarily in loanwords. The effects of the addition/removal of several such rules are outlined in table 4, with our unigram with substring matching model as the baseline.

Our evaluation shows that our system performs better than any of the systems outlined by Roy et al. (2013); their best systems obtained transliteration recalls of 50.90% and 47.50% for Bengali and Gujarati respectively, even ignoring proper nouns in their evaluation. Whilst obviously not directly comparable to Marathi, the (relative) linguistic similarity between Gujarati and Marathi provides at least some grounds for comparison.

	Baseline	Schwa	Vowel length	Foreign vowel
Recall (%)	70.37	72.57	70.15	70.37
Mean ambiguity (%)	42.42	250.25	40.12	169.48

Table 4. Changes in recall and ambiguity based on the inclusion of certain constraints (unigram w/ substr matching)

7.2. Qualitative

Interestingly, our results show no significant difference between the bigram model and the unigram one; the bigram model had the same gender agreement errors that the unigram ones did. A plausible explanation for this is that the frequency of specific determiner and noun combinations was low enough to be offset by the more frequent determiner, multiplied by a non-zero number after smoothing. We also evaluated models with higher-order *n*-grams; these showed no improvement.

We performed an analysis of 100 randomly sampled errors (table 5, page 327) from our set of back-transliteration failures. There were several interesting observations. First, most failures were foreign-language words. While many of these were English loanwords that were spelt the same as in English, and not phonetically (eg. *friend*, *perfume*), there were also several proper nouns from other Indian languages (primarily Hindi). Ambiguous back-transliterations were also an issue: often, multiple Devanagari words could be represented by a single Romanagari representation. This issue was particularly visible in agreement, and in word-final ambiguities between schwas and the vowel /a:/, where resolving a word-final *a* to either option would result in a valid word. Post-evaluation, we realised that it was possible to prevent some of this ambiguity: we introduced a post-processing measure that checked whether eliminating the word-final long vowel would result in a valid word. If it did, we included the long vowel. The justification for this was that whilst a word-final *a* could either represent a schwa or a long vowel, it would likely represent a long vowel where necessary to resolve ambiguity.

Our next source of failures was our schwa rule. These were of two different kinds: failures because the schwa rule inserted a schwa where none was necessary, and failures where it failed to insert a schwa where necessary. This was closely followed by words absent from our frequency list corpus. Next, we had 5 errors due to "impossible" to generate words, where the Romanagari was completely lossy: a character present in the source Devanagari was absent in its transliteration (eg. जेंव्हा *jeṃvhā* ("when"): Rom. *jevha*). Finally, two errors were due to problems with our rules, which we proceeded to fix post-evaluation.

Error type	Count
Foreign	30
Ambiguous input	29
Schwa rule fail	18
Unseen in corpus	16
"Incomplete" input	5
Rule absent	2

Table 5. Error analysis

8. Discussion

8.1. Analysis

The biggest issue with our system, at the moment, is our evaluation corpus itself. Whilst we did manage to assemble a reasonably diverse corpus, a larger corpus would allow for much better evaluation. More rigorous annotation control that accounted for variances in annotation style would also come in useful; Choudhury et al. (2010) describe several corpus bootstrapping techniques that we could use for more extensive future studies.

The most significant advantage of our finite-state system over statistical ones is the ability to easily model exceptions without having to retrain models. This makes it extremely trivial for us to add foreign words and proper nouns into our system. Further, due to the paradigm-oriented nature of *hfst*, adding the root of an OOV term would immediately allow for all concatenatively inflected forms of the noun to be (potentially) recognised. Setting up a finite-state system for back-transliteration, given linguistic knowledge - or even just native speaker intuition and a good grammar book - is also quite effortless. Based on our results, comparing similar systems for other south Asian languages, particularly parallel corpora-sparse ones, would be an interesting future project.

8.2. Improvements

Several of our ambiguity-related problems could be solved by taking context into account. Our bigram model was, unfortunately, not very successful: it would, however, be possible to integrate a morphological analyser into our pipeline. It could, for instance, determine the gender of the nearest noun and then force agreement on adjectives and verbs with ambiguous endings.

Whilst we currently rely on probabilities of complete generated strings, we could also integrated weighted FSTs into our system, where transitions are given certain weights based on their probabilities. Pereira and Riley (1997) have proposed a similar

system for speech recognition; Knight and Graehl (1998) adapted it to back-transliteration of Japanese katakana, implementing shortest-path algorithms to extract the most likely sequence.

9. Conclusions

There are several issues with schwas that need to be sorted out for our system to be truly "deployment" quality. However, it is important, again, to stress how "easy" it is to model exceptions with a finite-state based system. When attempting to gather social media text for analysis, our system could be used fairly trivially to obtain Devanagari equivalents to surface forms, which could then be rapidly post-edited. Common exceptions or loanwords could be obtained from a frequency list and added to the model prior to conversion.

Our system's recall on tokens - 76.28%, as described in table 2 on page 325 - is higher than our recall on terms, indicating that our system does perform better on more frequent words. This is quite encouraging, from a perspective of social media, where Romanagari is most likely to occur. Finally, our system is free and open-source, licensed under the GPL v3.0. This makes adoption for other Indian languages quite trivial, including for languages not included amongst the 22 "scheduled" languages of India, that are consequently (relatively) more underfunded and understudied.

Acknowledgements

We would like to thank Vaijayanti Ravishankar, Yamini Chitale and Anuja Phadke for their help with back-transliteration, to create part of our evaluation corpus. We also thank Francis Tyers for his comments on an earlier draft of this manuscript, Tommi Pirinen for his assistance with *hfst*, and the anonymous reviewers, whose insights and suggestions have been taken into account. This project has been funded by a stipend from the Erasmus Mundus Language and Communication Technology program, whom we are grateful to.

Bibliography

Choudhury, Monojit, Anupam Basu, and Sudeshna Sarkar. A diachronic approach for schwa deletion in Indo Aryan languages. In *Proceedings of the 7th Meeting of the ACL Special Interest Group in Computational Phonology: Current Themes in Computational Phonology and Morphology*, pages 20–26. Association for Computational Linguistics, 2004.

Choudhury, Monojit, Kalika Bali, Tirthankar Dasgupta, and Anupam Basu. Resource creation for training and testing of transliteration systems for indian languages. 2010.

Dhongade, R. and K. Wali. *Marathi*. London Oriental and African language library. John Benjamins Publishing Company, 2009. ISBN 9789027238139. URL `https://books.google.com.mt/books?id=zVV0vi5C8uIC`.

Forcada, Mikel L., Mireia Ginestí-Rosell, Jacob Nordfalk, Jim O'Regan, Sergio Ortiz-Rojas, Juan Antonio Pérez-Ortiz, Felipe Sánchez-Martínez, Gema Ramírez-Sánchez, and Francis M. Tyers. Apertium: a free/open-source platform for rule-based machine translation. *Machine Translation*, 25(2):127–144, June 2011. ISSN 0922-6567, 1573-0573. doi: 10.1007/s10590-011-9090-0. URL http://link.springer.com/10.1007/s10590-011-9090-0.

Gella, Spandana, Jatin Sharma, and Kalika Bali. Query word labeling and back transliteration for indian languages: Shared task system description. *FIRE Working Notes*, 3, 2013.

Heafield, Kenneth. KenLM: Faster and smaller language model queries. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, pages 187–197. Association for Computational Linguistics, 2011.

Kang, Byung-Ju and Key-Sun Choi. Automatic Transliteration and Back-transliteration by Decision Tree Learning. Citeseer, 2000.

Knight, Kevin and Jonathan Graehl. Machine transliteration. *Computational Linguistics*, 24(4): 599–612, 1998.

Lindén, Krister, Miikka Silfverberg, and Tommi Pirinen. HFST tools for morphology–an efficient open-source package for construction of morphological analyzers. In *International Workshop on Systems and Frameworks for Computational Morphology*, pages 28–47. Springer, 2009.

Lindén, Krister, Erik Axelson, Sam Hardwick, Tommi A Pirinen, and Miikka Silfverberg. Hfst—framework for compiling and applying morphologies. In *International Workshop on Systems and Frameworks for Computational Morphology*, pages 67–85. Springer, 2011.

Monojit, Umair Z Ahmed Kalika Bali. Challenges in designing input method editors for Indian languages: The role of word-origin and context. *Advances in Text Input Methods (WTIM 2011)*, page 1, 2011.

Pereira, FC and Michael D Riley. 15 Speech Recognition by Composition of Weighted Finite Automata. *Finite-state language processing*, page 431, 1997.

Rao, Chaitra, Avantika Mathur, and Nandini C Singh. 'Cost in Transliteration': The neurocognitive processing of Romanized writing. *Brain and language*, 124(3):205–212, 2013.

Roy, Rishiraj Saha, Monojit Choudhury, Prasenjit Majumder, and Komal Agarwal. Overview of the fire 2013 track on transliterated search. In *Post-Proceedings of the 4th and 5th Workshops of the Forum for Information Retrieval Evaluation*, page 4. ACM, 2013.

Address for correspondence:
Vinit Ravishankar
vinit.ravishankar@gmail.com
Faculty of Information and Communication Technology,
University of Malta,
Msida MSD 2080,
MALTA

PBML

The Prague Bulletin of Mathematical Linguistics
NUMBER 108 JUNE 2017 331–342

Linguistically Motivated Vocabulary Reduction for Neural Machine Translation from Turkish to English

Duygu Ataman,[a b] Matteo Negri,[b] Marco Turchi,[b] Marcello Federico[b]

[a] Università degli Studi di Trento, Trento, Italy
[b] Fondazione Bruno Kessler, Trento, Italy

Abstract

The necessity of using a fixed-size word vocabulary in order to control the model complexity in state-of-the-art neural machine translation (NMT) systems is an important bottleneck on performance, especially for morphologically rich languages. Conventional methods that aim to overcome this problem by using sub-word or character-level representations solely rely on statistics and disregard the linguistic properties of words, which leads to interruptions in the word structure and causes semantic and syntactic losses. In this paper, we propose a new vocabulary reduction method for NMT, which can reduce the vocabulary of a given input corpus at any rate while also considering the morphological properties of the language. Our method is based on unsupervised morphology learning and can be, in principle, used for pre-processing any language pair. We also present an alternative word segmentation method based on supervised morphological analysis, which aids us in measuring the accuracy of our model. We evaluate our method in Turkish-to-English NMT task where the input language is morphologically rich and agglutinative. We analyze different representation methods in terms of translation accuracy as well as the semantic and syntactic properties of the generated output. Our method obtains a significant improvement of 2.3 BLEU points over the conventional vocabulary reduction technique, showing that it can provide better accuracy in open vocabulary translation of morphologically rich languages.

1. Introduction

Neural machine translation (NMT) is a recent approach to machine translation (MT), which exploits deep learning to directly model the translation probability of

 Corresponding author: ataman@fbk.eu
Cite as: Duygu Ataman, Matteo Negri, Marco Turchi, Marcello Federico. Linguistically Motivated Vocabulary Reduction for Neural Machine Translation from Turkish to English. The Prague Bulletin of Mathematical Linguistics No. 108, 2017, pp. 331–342. doi: 10.1515/pralin-2017-0031.

Turkish	English
duy(-mak)	*(to) sense*
duygu	*sensation*
duygusal	*sensitive*
duygusallaş(-mak)	*(to) become sensitive*
duygusallaştırıl(-mak)	*(to) be made sensitive*
duygusallaştırılmış	*the one who has been made sensitive*
duygusallaştırılamamış	*the one who could not have been made sensitive*
duygusallaştırılamamışlardan	*from the ones who could not have been made sensitive*

Table 1. Turkish-to-English translation

texts in two different languages. Although the first models (Sutskever et al., 2014; Bahdanau et al., 2014) are only few years old, today NMT has already become the new state-of-the-art. Similar to other statistical approaches to MT, NMT is an instance of supervised learning, where a probabilistic model learns to predict an output given the input, based on an history of translation examples. The accuracy of the model is limited by the ability of the system to generalize to unseen examples, which is still an open issue in NMT due to computational restrictions. Current implementations of the model are computationally expensive; they require huge amounts of training time and memory space due to the large number of parameters to optimize. The translation engine uses a word vocabulary whose size is limited in order to control the complexity of the model. However, a text can only be translated if an exact match of the given source word can be found in the vocabulary.

Data sparseness, especially due to rare content words or infrequent inflected word forms, is one of the main reasons that limits the current performance of NMT in low-resourced and morphologically rich languages. For instance, Turkish, the language we focus on in this paper, is an agglutinative language where morphological inflections occur through attachment of suffixes to a given stem. Most syntactic forms in English, such as prepositions, negation, person or copula, are achieved solely through morphological inflections in Turkish. Table 1 illustrates the distance from Turkish to English in terms of the required translations to be generated by an ideal MT system. There are about 30,000 root words and 150 distinct suffixes in Turkish, which can experience agglutinative concatenations and internal changes through fusion to achieve vowel harmony, and cause the morphological tags to grow exponentially (Oflazer and El-Kahlout, 2007). Hence, the search for alternative word representation techniques that can solve the sparsity problem in Turkish is extremely important and can allow better handling of the input complexity.

Recent studies have tried implicitly extending the vocabulary by segmenting the words in the corpus into smaller units such as characters (Ling et al., 2015; Lee et al., 2016), sub-words (Sennrich et al., 2016; Wu et al., 2016) or hybrid (Luong and Manning,

2016) units. The problem with these approaches is that they disregard any notion of morphology during estimation of the sub-word units, which may lead to loss of semantic and syntactic information preserved in the word structure. In this paper, we propose to overcome this problem by developing a linguistically motivated segmentation method for open vocabulary translation of morphologically rich languages. We present a novel method that can perform segmentation to fit any desired vocabulary size for NMT while also considering the morphological properties of words. Being unsupervised, the proposed method can be fundamentally used with any language pair and direction in MT. We evaluate the benefit of our approach in a Turkish-to-English (TR-EN) NMT task against a conventional vocabulary reduction method that relies solely on statistics, and a supervised method that applies segmentation based on morphological analysis. The results show that our linguistically motivated vocabulary reduction method achieves significantly better translation accuracy compared to the conventional method and maintains its performance at different rates of vocabulary reduction.

2. Neural Machine Translation

The NMT model we use in this paper is based on the encoder-decoder and attention models described in (Bahdanau et al., 2014). First, a bi-directional RNN (the encoder) maps the sparse one-hot representation of an input sentence $X = (x_1, x_2, \dots x_m)$ into corresponding dense vectors called encoder hidden states. Then, a unidirectional RNN (the decoder) step-wisely predicts the target sequence $Y = (y_1, y_2, \dots y_j \dots y_l)$ as follows. The i^{th} target word is predicted by sampling from a word distribution computed from the previous target word y_{i-1}, the previous hidden state of the decoder, and a convex combination of the encoder hidden states (*i.e.* context vector). In particular, each weight of the combination is predicted by the attention model, on the basis of the previous target word, the previous decoder hidden state and the corresponding encoder hidden state. Both the encoder and decoder RNNs are implemented with GRU gates (Cho et al., 2014). The dimensions of the embeddings and hidden layers are proportional to the vocabulary size. Large vocabularies hence imply more parameters and higher computational costs.

3. Related Work

In general, two approaches have been proposed to cope with the limited vocabulary problem in NMT. The first one includes purely statistical methods, which aim to predict a set of sub-words that can optimally fit a given vocabulary size. These methods achieved state-of-the-art results for many morphologically rich languages (*e.g.* German, Russian, Czech and Finnish).

One such method is Byte-Pair Encoding (BPE), a likelihood-based sub-word unit generation method. BPE is originally a data compression algorithm (Gage, 1994), and

Corpus Frequency	Vocabulary Entry	English Translation
1011	hapishane	*jailhouse*
793	hapishan@@	-
587	hapishanede	*in the jailhouse*
245	hapishaneden	*from the jailhouse*
229	hapishanesinde	*at the jailhouse of (him/her/it)*
181	hapishanenin	*of the jailhouse*
100	hapishanesine	*to the jailhouse of (him/her/it)*

Table 2. Turkish vocabulary entries obtained with BPE

Source	Segmentation	NMT Output	Reference
kanunda	kan@@ unda	in **your blood**	in **the law**
sigortalılar	sigor@@ talı@@ lar	the insurer**s**	the insured **ones**

Table 3. Translation examples obtained when BPE is applied on Turkish words

has been recently modified by Sennrich et al. (2016) for vocabulary reduction, where the most frequent character sequences are iteratively merged to find the optimal description of the corpus vocabulary. Open vocabulary translation using this method is based on the assumption that many types of words can be translated when segmented into smaller units, such as named entities, compound words, and loanwords (Sennrich et al., 2016). Nevertheless, in cases of common morphological paradigms such as the derivational or inflectional transformations which are typically observed in Turkish, the method lacks a linguistic notion which would allow it to better generalize syntactic patterns among the data and use the vocabulary space more effectively. Table 2 lists some of the entries found in the NMT dictionary after the segmentation of the corpus with BPE, which stores many repetitions of the same lemma in different surface forms, indicating an inefficacy in capturing a compact representation of the data. Another crucial problem is related to the semantic losses which occur due to segmenting words at positions which breaks the morphological structure. Table 3 presents some of the typical mistakes observed in the NMT output when BPE is applied for segmentation. In the first example, the Turkish word *kanunda* (translation: **in the law**), the lemma of which is *kanun* (translation: **law**), is segmented in the middle of the root, which causes a semantic shift. The segmented word now becomes a completely different word, *kan* (translation: **blood**). In the second example, segmentation of the suffixes leads to generate the wrong inflected form in English.

Another set of purely statistical methods that attempted to cope with the vocabulary problem in NMT are based on the idea of constructing the translation model directly at the character-level (Ling et al., 2015; Lee et al., 2016). These models use

deep neural networks as compositional functions to predict representations of characters and new morphological forms. However, these models also assume that, by solely relying on statistics we might be able to capture the morphological rules that form the basics of semantics and syntax of language. Moreover, these models are known to generate spurious words that do not exist in the language (Lee et al., 2016).

The second family of approaches includes methods that also consider the morphological properties of words but can only reduce the vocabulary to a limited extent, usually by applying cut-off thresholds on the vocabulary and reducing the coverage of the long tail of less frequent words. For instance, Sánchez-Cartagena and Toral (2016) have used a morphological analyzer to separate words into root and inflection boundaries to achieve vocabulary reduction for NMT. However, in addition to failing to capture a full morphological description of words (*i.e.* generating the complete set of affixes existent in a word), their method cannot reduce the vocabulary of a given text to fit any vocabulary size. Another study tried to overcome this limitation by using the *Baseline* variant of Morfessor (Creutz and Lagus, 2005b), which allows to reach a vocabulary size set prior to segmentation (Bradbury and Socher, 2016). Although providing a sense of morphology into the segmentation process, this tool neglects the morphological varieties between sub-word units, which might result in sub-word units that are semantically ambiguous (*i.e.* either stems or suffixes).

In conclusion, to our knowledge, there is no vocabulary reduction method for NMT that can both reduce the vocabulary size at any given rate while also considering the individual morphological properties of the generated sub-word units. We aim to solve this problem with the segmentation method described in the next section.

4. Linguistically Motivated Vocabulary Reduction

We present a linguistically motivated segmentation method that achieves open vocabulary translation while considering the morphological properties of individual sub-word units. First, we propose using a supervised segmentation method based on morphological analysis, which helps us to evaluate our vocabulary reduction technique in terms of its ability to generalize the morphology of language from input data. This method aims to represent words in a less sparse way while preserving the complete morphological information. Later, we describe the method proposed in this paper, an unsupervised morphology learning algorithm that predicts the sub-word units in a corpus by a prior morphology model while reducing the vocabulary size to fit a given constraint.

4.1. Supervised Morphological Segmentation

As a supervised approach to linguistically motivated segmentation, we use a method which can reduce the word vocabulary of the Turkish corpus to only the root words along with a set of suffix units that are represented in terms of their inflec-

tional roles. This representation maintains a full description of the morphological properties of sub-word units in a word while minimizing the sparseness caused by inflection and allomorphy. We adopt the pre-processing approach of Bisazza and Federico (2009), who used the suffix combinatory finite-state analyzer of Oflazer (1994) to tag each sub-word unit in a Turkish word, and a morphological disambiguation tool (Sak et al., 2007) to decrease the sparseness caused by suffix allomorphy. After the pre-processing, we separate all roots and suffix tags into separate tokens and add an end-of-word (EOW) symbol for each analyzed word.

4.2. Unsupervised morphological segmentation

Supervised methods can provide the best accuracy in analysis, although, an ideal approach for MT should not require language-specific resources. Therefore, in this paper, we suggest to extend the unsupervised morphology induction framework Morfessor to develop a novel linguistically motivated vocabulary reduction method in NMT, which optimizes the complexity of the segmentation model with a constraint on the vocabulary size. The analysis of Creutz and Lagus (2005a) shows that Morfessor models optimized with the Maximum A-Posteriori (MAP) criterion generally achieve the best results. Our model is based on Morfessor *Flatcat* (Grönroos et al., 2014), a variant of this model family that uses a category-based Hidden Markov Model (HMM) and a flat lexicon structure. The category-based model is essential for a linguistically motivated segmentation as words would only be split considering the possible categories of their sub-words, preventing to split the words at random positions when a frequent sub-word is observed.

The aim of MAP optimization is to avoid overfitting by finding a balance between model accuracy and complexity. The model consists of two parts, a morpheme lexicon and a grammar that combines the language units together and generates new words. The MAP estimate of the overall system is given as:

$$M^* = \mathrm{argmax}_M P(D|M)P(M) \tag{1}$$

where the two factors represent the likelihood of the training corpus D and the prior probability of the model M. The former is estimated by an HMM which considers transitions between different morpheme categories (*e.g.* stem to suffix) when a word is constructed. The latter is modeled considering individual properties of the generated morphemes μ_i:

$$P(M) \approx m! \prod_i^m P(\mathrm{usage}(\mu_i))P(\mathrm{form}(\mu_i)) \tag{2}$$

where m is the number of distinct morphemes in the lexicon (Creutz and Lagus, 2007). The *usage* of a morpheme is related to its meaning and is modeled with its frequency, length, and the left and rightward perplexities. The *form* of a morpheme is the set of physical properties that distinguish it from the others in the lexicon.

Using the a-posteriori probability, one can train a segmentation model considering both the model complexity and the maximum-likelihood of the corpus, without any control on the size of the output lexicon. In order to use the model to achieve controlled vocabulary reduction for NMT, we insert a constraint on the desired lexicon size into the MAP optimization by applying a regularization weight over the lexicon cost and giving more favor in a reduction of the model complexity during optimization. The cost function is then estimated by the general formula:

$$L(D, M) = -\log P(D|M) - \alpha \log P(M) \quad (3)$$

where a higher α would force the algorithm to generate a smaller lexicon size and a higher amount of segmentation. Considering the tendency of the flat lexicon models to keep the frequent words unsegmented in the corpus (Grönroos et al., 2014), in order to achieve a more accurate segmentation model we disregard the frequency distribution $P(\mu_i)$ from the weighted part of the cost function. In fact, the value of the term is generally too small to affect the model complexity, but has an important role in determining the characteristics of the discovered morphemes.

For a given NMT vocabulary size limit, by setting the regularization weight α as $\frac{m_1}{m_2}$, where m_1 is the initial vocabulary size of the corpus, and m_2 is the desired vocabulary size, we achieve the right amount of regularization and the output lexicon size. The modified model has a new input parameter, *output lexicon size*, which sets the amount of regularization that reduces the vocabulary to the desired size. By using the parameter as a convergence limit we also minimize the model convergence time.

5. Experimental Set-up

We design two sets of experiments in order to evaluate our method. In the first experiment, we evaluate its ability to capture the morphological properties of sub-word units. As an indicator of vocabulary reduction that maintains the full morphological description and semantics of the original word, we deploy the supervised segmentation described in Section 4.1. However, the supervised method can only reduce the vocabulary to an extent. Hence, to eliminate the effect of out-of-vocabulary (OOV) words in test set to the accuracy, we set-up a controlled environment where we segment the data using the supervised method and sample the training, development and test sets so that they do not contain any OOVs. We also compare the performance of the method presented in Section 4.2, and BPE-based segmentation on the same data sets, and the case without segmentation. In order to achieve a fair comparison between the two vocabulary reduction methods, we train the splitting rules of our method and BPE only on the source side of the parallel data. In the second set of experiments, we evaluate our method in a real case scenario. We do not include the supervised method in this phase as its performance would be highly affected by the amount of OOVs in the training and test sets. In Experiment 2.a, we use data sets of

Data set	Experiment	#sentences (K)	#tokens (M)	#types (K)
TED	(1)	115	1.6 (TR) - 2.2 (EN)	141 (TR) - 44 (EN)
TED	(2.a)	133	1.9 (TR) - 2.7 (EN)	169 (TR) - 53 (EN)
TED + Generic	(2.b)	283	4.1 (TR) - 5.6 (EN)	268 (TR) - 96K (EN)

Table 4. Data sets used in each experiment. K - thousand, M - million.

similar distribution, whereas in Experiment 2.b, we increase data sparsity by adding generic data to the training set. We segment the source side of parallel corpora using different methods while we segment the target side with BPE. We measure the performance in either experiment (2.a and 2.b) on the same test set.

We use two sets of data for training our NMT systems. The first data set is the Turkish-English portion of TED Talks (Cettolo et al., 2012) from IWSLT (Paul et al., 2010) and is used in Experiment 1 and 2.a. The second data set is a combination of TED Talks and a collection of generic data from EU Bookshop (Skadiņš et al., 2014), Global Voices, Gnome, Tatoeba, Ubuntu (Tiedemann, 2012), KDE4 (Tiedemann, 2009), Open Subtitles (Lison and Tiedemann, 2016) and SETIMES (Tyers and Alperen, 2010), filtered using the invitation model of Cuong and Simaan (2014) to reduce the size. The generic data is used in Experiment 2.b. In all the experiments, we use development and test sets of 1,000 sentences and use the remaining data for training the models. The statistics of all the data sets used in each experiment are given in Table 4.

The NMT models used in the evaluation are based on the Nematus toolkit (Sennrich et al., 2017). They have a hidden layer and embedding dimension of 1024, a mini-batch size of 100 and a learning rate of 0.01. The dictionary size is 40,000 (*src* & *trg*) in the 1^{st}, and 30,000 (*src*) - 40,000 (*trg*) in the 2^{nd} experiment. We train the models using the Adagrad (Duchi et al., 2011) optimizer with a dropout rate of 0.1 (*src* & *trg*) and 0.2 (*embeddings and hidden layers*). We shuffle the data at each epoch. BPE merge rules are of equal size to the dictionary. We train the models for 50 epochs and choose the best model on the development set for translating the test set.

The modified Morfessor *FlatCat* models (Grönroos et al., 2014) are trained with a perplexity threshold of 10, a length threshold of 5, and an *output lexicon size* of 40,000 (*Experiment 1 & 2.a*) and 30,000 (*Experiment 2.b*), which is a new input parameter added to the model implementation. Training time is 20 minutes (using an Intel Xeon E3-1240 v5 CPU), while segmentation time varies from 10 to 30 minutes, depending on the corpus size. Performance is measured using the BLEU (Papineni et al., 2002), TER (Snover et al., 2006) and CHRF3 (Popovic, 2015) scores and significance tests are computed with Multeval (Clark et al., 2011).

1. TED corpus, no-OOV case, voc=40K			
Method	**BLEU↑**	**TER↓**	**CHRF3↑**
No Segmentation	17.77	68.07	38.94
BPE	19.52	66.23	42.33
Supervised	21.61▲	61.76▲	**44.01**
LMVR	**21.71▲**	**61.41▲**	43.90

Input (*Reference*)	**Method**	**Segmentation**	**Output**
ağlarını	BPE	ağ@@ larını	the cry
(*the nets*)	LMVR	ağ +larını	the nets
	Supervised	ağ +Noun + A3pl <EOW>	networks
ağlamayacak	BPE	ağ@@ lamayacak	will not survive
(*would not be crying*)	LMVR	ağlama +yacak	will not cry
	Supervised	ağla +Neg +Fut +A3sg <EOW>	will not cry

Table 5. Results of Experiment 1 - TED corpus and no-OOV case. Top: Output accuracies, where ▲indicates statistically significant improvement over the BPE baseline (p-value < 0.05). Bottom: Translation examples.

6. Results and Discussion

Table 5 shows the performance of different segmentation methods in Experiment 1. Our linguistically motivated vocabulary reduction (LMVR) method achieves the best performance on average, proving our hypothesis that a correct morphological representation generates more accurate translations. Our method outperforms the strong baseline of BPE-based segmentation by **2.2** BLEU, **4.8** TER and **1.6** CHR3F points. The performance is slightly higher than the supervised method, which is related to the ambiguity caused by loss of information during the morphological analysis. The predicted vocabularies also indicate the significant difference between LVMR and BPE, where 73% of the sub-word units in the vocabulary are completely different. In order to better illustrate the properties of the generated sub-word units, we present example translations of two words from the test set. The two words have different roots, the first one is *ağ* (translation: **net**), and the second one is *ağla* (translation: **(to) cry**). BPE segments both words to the same root *ağ*, a character sequence frequently observed in root words in Turkish. In the first case, both unsupervised methods segment the word into the same sub-word units, while the embedding of the sub-word unit segmented with BPE is semantically ambiguous and generates unreliable translations. On the other hand, our method can preserve the correct meaning in both cases.

In Experiment 2, we evaluate our method at different rates of vocabulary reduction according to the vocabulary sizes given in Table 4. All metrics confirm that our method achieves better performance than the baseline in both experiments. In Experiment 2.a, at a vocabulary reduction rate of 4.25 (140K -> 40K), we obtain an improvement of **2.3** BLEU points over the baseline. In the most challenging case, Experiment

	2.a TED corpus, OOV case, voc=40K			2.b Large corpus, OOV case, voc=30K		
Method	**BLEU↑**	**TER↓**	**CHRF3↑**	**BLEU↑**	**TER↓**	**CHRF3↑**
BPE	20.45	64.50	42.65	24.42	60.14	47.05
LMVR	**22.76**▲	**62.94**▲	**45.36**	**25.42**▲	**58.88**▲	**47.71**

Table 6. Results of Experiment 2 - OOV presence and different rates of vocabulary reduction. ▲indicates statistically significant improvement over the BPE baseline (p-value < 0.05).

2.b, we increase the training set using data coming from varying domains, which maximizes the sparseness due to rare word forms in the corpus. Furthermore, we decrease the source vocabulary limit to 30,000, requiring a vocabulary reduction rate of 9 (270K -> 30K). As given in Table 6, our method can still outperform the baseline by **1.0** BLEU point. The results and the computational efficiency of our method prove that it can be deployed in practical NMT systems trained with generic corpora.

7. Conclusion

In this paper we have addressed the vocabulary limitation in NMT, which has been an open issue in the translation of morphologically rich languages. For this purpose, we have proposed a novel linguistically motivated vocabulary reduction method that can achieve open vocabulary translation while, unlike previous approaches, maintaining a linguistic notion at the sub-word level. The method is completely unsupervised and can estimate a fixed size dictionary of sub-word units considering their individual morphological properties. We have evaluated our method against a statistical vocabulary reduction method and showed that our method obtains significantly better performance due to bringing a linguistic notion into the segmentation process.

Acknowledgements

This work has been partially supported by the EC-funded H2020 projects QT21 (grant no. 645452) and ModernMT (grant no. 645487). The authors would like to thank Arianna Bisazza and Prashant Mathur for their contributions to this study.

Bibliography

Bahdanau, Dzmitry, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *CoRR*, abs/1409.0473, 2014.

Bisazza, Arianna and Marcello Federico. Morphological pre-processing for Turkish to English statistical machine translation. In *IWSLT*, pages 129–135, 2009.

Bradbury, James and Richard Socher. MetaMind neural machine translation system for WMT 2016. In *Proceedings of the 1st Conference on Machine Translation. ACL*, 2016.

Cettolo, Mauro, Christian Girardi, and Marcello Federico. WIT3: Web Inventory of Transcribed and Translated Talks. In *Proceedings of EAMT*, pages 261–268, 2012.

Cho, Kyunghyun, Bart van Merrienboer, Dzmitry Bahdanau, and Yoshua Bengio. On the Properties of Neural Machine Translation: Encoder-Decoder Approaches. In *Proceedings of the 8th Workshop on Syntax, Semantics and Structure in Statistical Translation*, pages 103–111, 2014.

Clark, Jonathan H., Chris Dyer, Alon Lavie, and Noah A. Smith. Better Hypothesis Testing for Statistical Machine Translation: Controlling for Optimizer Instability. In *Proceedings of the 49th Annual Meeting of ACL*, pages 176–181. ACL, 2011.

Creutz, Mathias and Krista Lagus. Inducing the morphological lexicon of a natural language from unannotated text. In *Proceedings of the International and Interdisciplinary Conference on Adaptive Knowledge Representation and Reasoning*, pages 51–59, 2005a.

Creutz, Mathias and Krista Lagus. *Unsupervised morpheme segmentation and morphology induction from text corpora using Morfessor 1.0*. Helsinki University of Technology, 2005b.

Creutz, Mathias and Krista Lagus. Unsupervised models for morpheme segmentation and morphology learning. *Transactions on Speech and Language Processing*, 4(1):3, 2007.

Cuong, Hoang and Khalil Simaan. Latent domain translation models in mix-of-domains haystack. In *Proceedings of COLING*, pages 1928–1939, 2014.

Duchi, John, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12(Jul):2121–2159, 2011.

Gage, Philip. A new algorithm for data compression. *The C Users Journal*, 12(2):23–38, 1994.

Grönroos, Stig-Arne, Sami Virpioja, Peter Smit, and Mikko Kurimo. Morfessor FlatCat: An HMM-Based Method for Unsupervised and Semi-Supervised Learning of Morphology. In *COLING*, pages 1177–1185, 2014.

Lee, Jason, Kyunghyun Cho, and Thomas Hofmann. Fully Character-Level Neural Machine Translation without Explicit Segmentation. *CoRR*, abs/1610.03017, 2016.

Ling, Wang, Isabel Trancoso, Chris Dyer, and Alan W Black. Character-based neural machine translation. *CoRR*, abs/1511.04586, 2015.

Lison, Pierre and Jörg Tiedemann. Opensubtitles2016: Extracting large parallel corpora from movie and tv subtitles. In *Proceedings of LREC*, 2016.

Luong, Minh-Thang and Christopher D Manning. Achieving open vocabulary neural machine translation with hybrid word-character models. In *Proceedings of the 54th Annual Meeting of ACL*. ACL, 2016.

Oflazer, Kemal. Two-level description of Turkish morphology. *Literary and linguistic computing*, 9(2):137–148, 1994.

Oflazer, Kemal and Ilknur Durgar El-Kahlout. Exploring different representational units in English-to-Turkish statistical machine translation. In *Proceedings of the 2nd Workshop on Statistical Machine Translation*, pages 25–32. ACL, 2007.

Papineni, Kishore, Salim Roukos, Todd Ward, and Wei-Jing Zhu. BLEU: A Method for Automatic Evaluation of Machine Translation. In *Proceedings of the 40th Annual Meeting of ACL*, pages 311–318. ACL, 2002.

Paul, Michael, Marcello Federico, and Sebastian Stücker. Overview of the IWSLT 2010 Evaluation Campaign. In *Proceedings of IWSLT*, pages 3–27, 2010.

Popovic, Maja. chrF: character n-gram F-score for automatic MT evaluation. 2015.

Sak, Haşim, Tunga Güngör, and Murat Saraçlar. Morphological disambiguation of Turkish text with perceptron algorithm. In *International Conference on Intelligent Text Processing and Computational Linguistics*, pages 107–118. Springer, 2007.

Sánchez-Cartagena, Vıctor M and Antonio Toral. Abu-MaTran at WMT 2016 Translation Task: Deep Learning, Morphological Segmentation and Tuning on Character Sequences. In *Proceedings of the 1st Conference on Machine Translation. ACL*, 2016.

Sennrich, Rico, Barry Haddow, and Alexandra Birch. Neural machine translation of rare words with subword units. *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, 2016.

Sennrich, Rico, Orhan Firat, Kyunghyun Cho, Alexandra Birch, Barry Haddow, Julian Hitschler, Marcin Junczys-Dowmunt, Samuel L"aubli, Antonio Valerio Miceli Barone, Jozef Mokry, and Maria Nadejde. Nematus: a Toolkit for Neural Machine Translation. In *Proceedings of EACL*, 2017.

Skadiņš, Raivis, Jörg Tiedemann, Roberts Rozis, and Daiga Deksne. Billions of Parallel Words for Free: Building and Using the EU Bookshop Corpus. In *Proceedings of LREC*. European Language Resources Association, 2014.

Snover, Matthew, Bonnie Dorr, Richard Schwartz, Linnea Micciulla, and Ralph Weischedel. A Study of Translation Error Rate with Targeted Human Annotation. In *Proceedings of AMTA*, 2006.

Sutskever, Ilya, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112, 2014.

Tiedemann, Jörg. News from OPUS-A collection of multilingual parallel corpora with tools and interfaces. In *Recent advances in natural language processing*, volume 5, pages 237–248, 2009.

Tiedemann, Jörg. Parallel Data, Tools and Interfaces in OPUS. In *Proceedings of LREC*. European Language Resources Association, 2012.

Tyers, Francis M and Murat Serdar Alperen. South-east European Eimes: A parallel corpus of balkan languages. In *Proceedings of the LREC Workshop on Exploitation of Multilingual Resources and Tools for Central and (South-) Eastern European Languages*, pages 49–53, 2010.

Wu, Yonghui, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation. *CoRR*, abs/1609.08144, 2016.

Address for correspondence:
Duygu Ataman
ataman@fbk.eu
Via Sommarive 18, Povo, 38123 Trento, Italy

PBML

The Prague Bulletin of Mathematical Linguistics
NUMBER 108 JUNE 2017 343–354

Questing for Quality Estimation A User Study

Carla Parra Escartín,[a] Hanna Béchara,[b] Constantin Orăsan[b]

[a] ADAPT Centre, SALIS, Dublin City University, Ireland
[b] RGCL, University of Wolverhampton, UK

Abstract

Post-Editing of Machine Translation (MT) has become a reality in professional translation workflows. In order to optimize the management of projects that use post-editing and avoid underpayments and mistrust from professional translators, effective tools to assess the quality of Machine Translation (MT) systems need to be put in place. One field of study that could address this problem is Machine Translation Quality Estimation (MTQE), which aims to determine the quality of MT without an existing reference. Accurate and reliable MTQE can help project managers and translators alike, as it would allow estimating more precisely the cost of post-editing projects in terms of time and adequate fares by discarding those segments that are not worth post-editing (PE) and have to be translated from scratch.

In this paper, we report on the results of an impact study which engages professional translators in PE tasks using MTQE. We measured translators' productivity in different scenarios: translating from scratch, post-editing without using MTQE, and post-editing using MTQE. Our results show that QE information, when accurate, improves post-editing efficiency.

1. Introduction

Machine Translation Post-Editing (MTPE) has become a reality in industrial translation settings. This has impacted on translation workflows as translators are imposed shorter deadlines and lower rates for these tasks than when translating from scratch. However, the quality of Machine Translation (MT) still remains an issue, particularly for post-editors, who usually complain that they spend more time assessing the MT output quality and fixing the translations, than when translating the same text from scratch. Many professional translators acknowledge that after a few segments un-

 Corresponding author: carla.parra@adaptcentre.ie
Cite as: Carla Parra Escartín, Hanna Béchara, Constantin Orăsan. Questing for Quality Estimation A User Study. The Prague Bulletin of Mathematical Linguistics No. 108, 2017, pp. 343–354. doi: 10.1515/pralin-2017-0032.

dergoing MTPE, they delete the remaining segments and translate everything from scratch if they deem that it will take them less time. This suggests that in some cases the translations suggested are not good enough. MT Quality Estimation (MTQE) can address this issue by assessing the quality of an automatically translated segment and proposing for post-editing only those that are good enough.

Quality estimation in MT aims to predict the quality of the MT output without using a reference translation (Blatz et al., 2004; Specia et al., 2011). This field has received extensive interest from the research community in recent years, resulting in the proposal of a number of machine learning methods that estimate the quality of a translation on well defined data sets, but which do not necessarily reflect the reality of professional translators. In order to integrate MTQE successfully in translation workflows it is necessary to know when a segment is good enough for a translator. However, and as pointed out by Turchi et al. (2015), "QE research has not been followed by conclusive results that demonstrate whether the use of quality labels can actually lead to noticeable productivity gains in the CAT framework".

In this paper, we present a user study which aims to understand better how quality estimation should be used in order to improve the productivity of professional translators. To achieve this, we use the English to Spanish part of the Autodesk Post-Editing data corpus (ISLRN 290-859-676-529-5) and 4 professional translators. As this data set comprises real data used by Autodesk in past translation projects, it constitutes a valid and publicly open dataset for our experiments to validate the usability of MTQE in real translation scenarios.

The remainder of this paper is structured as follows: in the next Section 2, we briefly discuss previous relevant work. Section 3 reports on the experimental setting on the study that we carried out, followed by analysis and discussion of the results in Section 4. Finally, Section 5 wraps up our work and discusses future paths to be explored.

2. Related Work

Although MTQE has not been widely tested in real translation workflows, a few researchers, particularly in the field of translation studies, have attempted to cover this gap and assess to which extent MTQE could be useful for professional translators. In their work, Turchi et al. (2015) assess whether the use of quality labels can actually lead to noticeable productivity gains. They do so by first establishing the conditions to carry out on-field evaluation and then carrying out an experiment providing translators with binary quality labels (green and red, depending on the MTQE obtained for the segment). They observed a non-significant productivity increase in translators' productivity though. When dividing the test data according to segment length and quality, they concluded that "the higher percentage of wins is statistically significant only for medium-length suggestions with HTER>0.1". Their data set accounted for 1389 segments (542 were used in training the QE engine, and 847 in testing) and

their experiment was carried out by four professional translators. In total, they gathered two instances of each segment, one for the scenario in which the translator was shown the MT output QE, and one in which the translator did not have a QE of the MT output.

Moorkens et al. (2015) researched whether human estimates of post-editing effort accurately predict actual post-editing effort and whether the display of confidence scores (MTQE) influences post-editing behaviour. In their study, they used two different groups of participants. One consisting of six members of staff, postdoctoral researchers and PhD students of a Brazilian University, and a second one consisting of 33 undergraduate and Masters translation students. The first group of participants were asked to assess the quality of a set of 80 segments of two Wikipedia articles describing Paraguay and Bolivia and Machine Translated into Portuguese using Microsoft's Bing Translator. They were asked to classify the MT output according to a 3-grade scale:

1. Segments requiring a complete retranslation;
2. Segments requiring some post-editing but for which PE is still quicker than retranslation; and
3. Segments requiring little or no post-editing.

Secondly, and after a break of at least 2 weeks to avoid the participants remembering their ratings, the same participants were asked to post-edit the segments without any type of MTQE being shown. Finally, the second group of participants (undergraduate and masters students), were asked to post-edit the same sample but in this case MTQE was used. Using the average ratings of the first phase of their research, Moorkens et al. (2015) colour-coded each segment in red (better to retranslate), amber (MT could be useful but requires post-editing), and green (MT requires little or no post-editing). Although their study is based in a rather small sample, their findings suggest that "the presentation of post-editing effort indicators in the user interface appears not to impact on actual post-editing effort".

Moorkens and Way (2016) researched the acceptability of translation memory (TM) compared to that of MT among translators. They engaged 7 translators and asked them to rate 60 segments translated from English into German. The text was taken from the documentation of the an open-source computer-aided design program called FreeCAD and from the Wikipedia entry describing what computer-aid design is. They conclude that when low- and mid-ranking fuzzy matches are presented to translators without scores, translators find the suggestions irritating, and for over 36% of such instances, useless for their purposes. In contrast, in their experiment all of the MT matches suggested were rated as having some utility to post-editors. Moorkens and Way (2016) conclude that their findings suggest that "MT confidence measures need to be developed as a matter of urgency, which can be used by post-editors to wrest control over what MT outputs they wish to see, and perhaps more importantly still, which ones should be withheld".

Finally, in a recent study aiming at determining the user interface needs for post-editors of MT, Moorkens and O'Brien (2017) report that of the respondents to a survey aiming at determining the features that translators wished post-editing interfaces had, 81% expressed the wish of having a feature showing confidence scores for each target text segment from the MT engine. This finding makes the impact study reported here of utmost relevance, as we precisely aimed at investigating the impact of showing MTQE to translators when undergoing MTQE tasks. This will be explained in the next section 3.

3. Experimental Setup

3.1. Data

We decided to use the Autodesk Post-Editing Data corpus in our experiments in order to simulate a real translating experience. This corpus consists of parallel data with English as the source language and 12 different target languages. The size per language pair varies from 30,000 to 410,000 segments, and each segment is labelled with information as to whether it comes from a Translation Memory (TM) match or it is MT output. The post-edited target sentences are also included in the dataset, along with a raw MT score and a Fuzzy Match Score. The data belongs to a technical domain, and the segments come predominantly from software manuals.

We selected our sentences from the English to Spanish part of the corpus. We then used a semantically enriched version of QuEst++ (cf. Section 3.2) in order to predict the target-side Fuzzy Match Score (FMS) of the machine translation output. We decided to use the FMS as translators are more used to working with Translation Memory leveraging and fuzzy matches (Parra Escartín and Arcedillo, 2015a,b) than to more traditional MT evaluation metrics such as BLEU (Papineni et al., 2002) or HTER (Snover et al., 2006). While FMS is usually computed on the source side of a text, in our case, and similarly to what is proposed in Parra Escartín and Arcedillo (2015a,b), we use the FMS as a MT evaluation metric and thus aim at predicting a target-side FMS. Following the findings in Parra Escartín and Arcedillo (2015b), we established a threshold of 75% FMS or higher to consider a segment worth to be post-edited.

The sentences used in this experiment were selected in such a way that a quarter were chosen from sentences where the QE system performed well ("Good QE"). In these cases the predicted FMS for each sentence is close enough to the observed FMS score to give the translator a correct idea of its quality (within 5%). Another quarter were chosen from sentences where the QE system performed badly ("Bad QE"). In this case the observed score is more than 10% off compared to the observed score.[1] Another quarter of the sentences do not include MTQE information, and the final

[1]Given the difference between the predicted QE and the observed score some of the segments are being mislabelled as "worth post-editing"/"not-worth postediting". We took this into consideration during our evaluation.

quarter have no MT suggestion at all ("Translate From Scratch"). For the purpose of this study, our total number of sentences is 300 (about 3000 words) equally distributed among the four categories above (i.e. each category contained a total of 75 segments).

3.2. The Quality Estimation System

In our experiments, we use QuEst++ (Specia et al., 2015) enhanced with the semantically motivated features we described in (Béchara et al., 2016). QuEst++ is considered to be the state-of-the-art framework for MTQE tasks and is used as a baseline in the most recent MTQE shared tasks, such as the ones in 2014 (Bojar et al., 2014), 2015 (Bojar et al., 2015), and 2016 (Bojar et al., 2016). It includes a feature extraction framework and also provides with the machine learning algorithms necessary to build the MTQE prediction models. The 17 baseline features are language independent and include shallow surface features (e.g. number of punctuation marks, average length of words, number of words, etc.). They also include n-gram frequencies and language model probabilities.

We tuned QuEst++ with in-domain data, building our own language models and n-gram counts from the Autodesk Dataset. As estated above, we also added a number of additional features to the system. More concretely, we extracted a variety of linguistically motivated features inspired by deep semantics such as distributional Similarity Measures, Conceptual Similarity Measures, Semantic Similarity Measures and Corpus Pattern Analysis (Béchara et al., 2016). We integrated these Semantic Textual Similarity (STS) features into the QE pipeline and noticed an improvement over the baseline. By replicating the experiments in Béchara et al. (2016) for the Autodesk data, we observe similar results as demonstrated in Table 1.

System Description	MAE
QuEst++ – out of the box	9.82
QuEst++ – tuned for in-domain data	9.78
QuEst++ – with STS features	9.52

Table 1: MAE predicting the FMS for Autodesk

3.3. PET: Post-Editing Tool

For our study we use PET (Aziz et al., 2012) as our post-editing tool. Like other CAT tools, PET provides an easy to use user interface which facilitates both translating and post-editing. In addition, the tool records a number of statistics such as the keystrokes pressed and the time needed to perform the translation, which are very relevant for this research. Even though PET is unlikely to be used in a real-world post-editing

situation, it is ideal for our research. The tool is open-source and written in Java, which allowed us to easily modify the code to incorporate the traffic light system described in section 3.4. While other tools such as SDL Trados Studio[2] or MemoQ[3] would have been preferred by the translators due to both familiarity and ease of use, these tools did not allow the same kind of malleability and customisation as PET, which allowed us access to the source code in order to edit in our traffic lights.

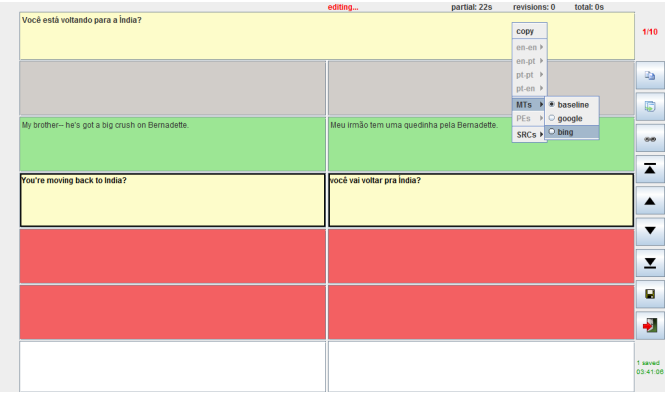


Figure 1: A Screenshot of PET out of the box

3.4. The User Study

Inspired by the work reported by Turchi et al. (2015), we modified PET to present translators with a traffic light system which suggests the type of task they were facing in each case:

Light yellow (referred to in the evaluation as *Translate*) indicated that a translator had to translate the given sentence from scratch (in this case, the translator was not given an MT translated sentence to post-edit).

Light blue (*Post-edit*) indicated that a machine translation of the source segment is available, however, no MTQE information is provided, and therefore the translators must decide for themselves whether to translate from scratch, or to post-edit.

Light green (*QE Post-edit*) indicated that the MTQE system strongly suggests that the translator should post-edit the sentence produced by the MT engine. As indi-

[2] http://www.sdl.com/solution/language/translation-productivity/trados-studio/

[3] https://www.memoq.com/en/

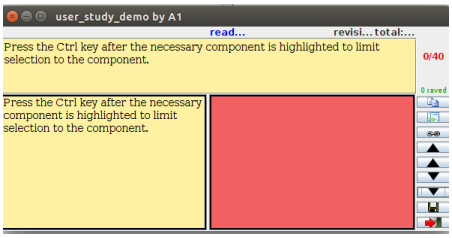


Figure 2: Translate from scratch

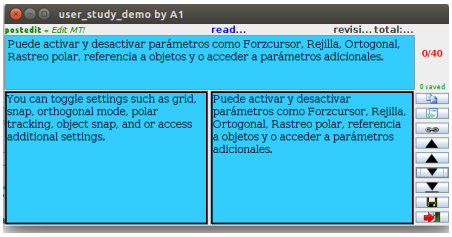


Figure 3: Post-edit without MTQE

cated earlier, this means that the MTQE system has predicted a fuzzy match score of 75% or more.

Light red (*QE Translate*) indicated that the MTQE system strongly suggests that the translator translates the sentence from scratch. This means that the MTQE system has predicted a fuzzy match score of less than 75%.

Figures 2 and 3 show how the colour coding system was displayed to the translators.

In order to refine our experiment, we performed a pilot study engaging 4 non-professional translators who are native speakers of Spanish. These translators were asked to look at a subset of 40 sentences from the full dataset. While the results of this study remained inconclusive in terms of linking productivity to MTQE, we learned a lot about the needs of the translators and the presentation of the task. For the full study, we enlisted the help of 4 Spanish professional translators with several years' translating experience. The years of experience varied greatly, between 3 and 14 years experience. All 4 translators had some experience with Computer-Assisted Translation tools and Post-Editing tasks. All 4 translators are native speakers of Spanish with a working proficiency of English and were asked to fill out questionnaires before and after completing the tasks with the aim of gathering information about their background and their experience while performing the task. Table 2 summarises the translator details.

While all translators had some experience with post-editing tools, none of the translators were familiar with PET before participating in the experiment. To overcome this issue, together with the instructions to carry out the task for the experiment, we also provided them with a short user manual of the tool with screenshots aiming at familiarising them with the interface prior to the task itself. All translators were paid for their time and were asked to complete the task over the course of a day, in order to simulate the real-world experience.

Translator	C	M	V	S
Experience in technical domains (years)	14	6	3	6
Experience as a professional translator (years)	14	6	3	9
Experience with post-editing tools (years)	2	4	3	1
Opinion of Computer-Assisted Translation tools	Pos	Pos	Pos	Pos
Opinion of post-editing tasks	Neg	Pos	Pos	Pos

Table 2: Overview of the professional translators engaged in the experiment

4. Results and Discussion

We extracted the post-editing times and keystrokes for all 4 translators. We then normalised these results by dividing each by the number of tokens in the final post-edited target sentence in order to compare sentences of different lengths. We also discarded one sentence, because the post-editing time exceeded 9000 seconds. In cases where a translator skipped a sentence, we discarded their statistics as well. In both cases we discarded the sentence data for all translators, in order to ensure the results remained comparable. In total, we discarded 4 sentences this way. In this section, we summarise the results on the remaining data.

Figure 4 shows the time, measured in seconds per word, that each translator spent on a given type of task (raw post-editing, translating from scratch, QE Postedit and QE translate). Each translator is identified by a letter. In addition, we provide the average for all four translators. As we expected, the sentences that needed to be translated from scratch took the most time across all translators, even without taking into account the quality of the QE. This seems to suggest that MT can considerably boost translator efficiency.

In Figure 5 we look closer at the time spent post-editing, separating out the good and bad QE. Here we can see that good quality estimation results, on the other hand, seem to consistently enhance performance across all translators. The average time spent per token drops from 1.62 seconds for no QE to 1.15 seconds for good QE.

In order to gain more insight, we also take a look at the number of keystrokes by type of task and by MTQE quality. Figures 6 and 7 take a closer look based on the type of task and the quality of MTQE respectively. The number of keystrokes used in post-editing is clearly lower than the number of keystrokes used when translating from scratch. Strangely enough, translators used less keystrokes in the cases where, despite being given a translation, they were instructed to translate from scratch, than when they were asked to post-edit the translation. This is an unexpected result and it will have to be investigated further. One possible explanation could be that they used the arrow keys a lot to navigate in the segment. The average number of keystrokes per segment drops from 81 for no QE to 46 with bad QE and 33 with good QE. Here we can see that even bad QE seems to be a better aid than no QE at all, at least in terms of

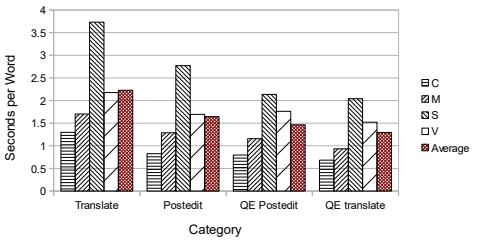


Figure 4: Number of seconds per word spent translating/ post-editing per type of task

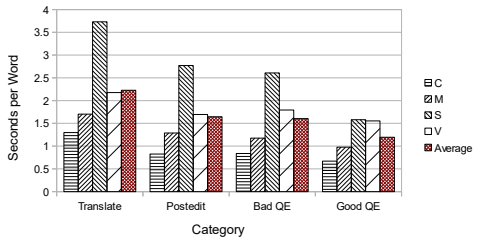


Figure 5: Number of seconds per word spent translating/ post-editing by QE quality

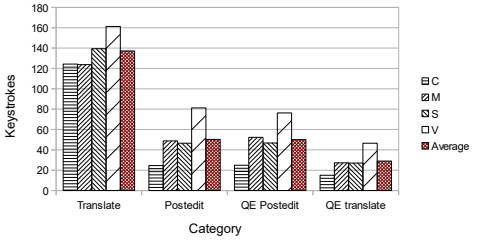


Figure 6: Number of keystrokes per segment spent translating/ post-editing per type of task

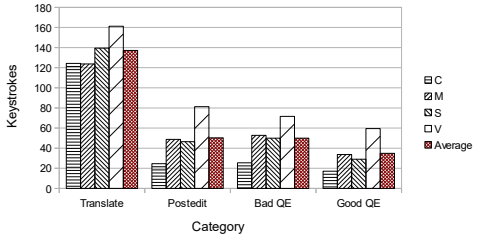


Figure 7: Number of keystrokes per segment spent translating/ post-editing by QE quality

post-editing effort as measured by keystrokes, and in this experimental setting. This might be because even the segments which are marked as "translate from scratch" provide a MT output which gives translators at least something to work with rather than starting from nothing.

As part of the experiment, we also asked all the translators to fill out questionnaires before and after the task in order to gain a more first-hand perspective of translators and post-editing tools. Responses suggest that while all four translators approved of the MT suggestions, all found the post-editing tool difficult to navigate, which may have affected both their results and opinions of MTQE. Despite our findings, three of the translators answered that they did not find MTQE helpful. However, as the translators had no way of distinguishing which was good and which was bad QE, this could have influenced their opinions of the usefulness of it. One translator disagreed,

saying that they liked getting a first impression via the traffic lights system. Three out of the four translators claimed that the MT suggestions were helpful, while one insisted that they were better off translating from scratch, despite the high increase in efficiency shown by the results above.

5. Conclusion and Future Work

In this paper, we have reported on the results of a user study we conducted to investigate the impact of using the MTQE information in the post-editing workflow. We engaged 4 professional Spanish translators to take part in a post-editing/translation task, using a traffic lights system to provide MTQE information. We ran a study using 300 sentences from the Autodesk post-editing parallel corpus, annotated for Fuzzy Match Scores (FMS) using a semantically enhanced version of QuEst++.

Despite our rather small sample, our results seem to indicate that MTQE, especially good and accurate MTQE, is vital to the efficiency of the translation workflow, and can cut translating time and effort significantly. Translator feedback still seems quite negative in spite of this improvement, which suggests a better post-editing tool might be required to win over the translators.

In future work, we plan to analyse the results of this user study further. The data compiled through this experiment will also be released to allow other researchers to replicate our work or carry out further studies and/or experiments. We would also like to test whether the results reported here replicate for other language pairs and domains. Similar findings in such experiments would demonstrate the need for accurate and reliable MTQE, as well as the need to integrate it in professional translation workflows to improve post-editing efficiency. Our results, despite preliminary, seem to indicate this.

Acknowledgements

The authors wish to thank the anonymous reviewers for their valuable feedback. This research is supported by the People Programme (Marie Curie Actions) of the European Union's Framework Programme (FP7/2007-2013) under REA grant agreement nº 317471, the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement Nº 713567, and Science Foundation Ireland in the ADAPT Centre (Grant 13/RC/2106) (www.adaptcentre.ie) at Dublin City University.

Bibliography

Aziz, Wilker, Sheila Castilho, and Lucia Specia. PET: a Tool for Post-editing and Assessing Machine Translation. In *LREC*, pages 3982–3987, 2012.

Béchara, Hanna, Carla Parra Escartín, Constantin Orăsan, and Lucia Specia. Semantic Textual Similarity in Quality Estimation. *Baltic Journal of Modern Computing*, 4(2):256, 2016.

Blatz, John, Erin Fitzgerald, George Foster, Simona Gandrabur, Cyril Goutte, Alex Kulesza, Alberto Sanchis, and Nicola Ueffing. Confidence estimation for machine translation. In *Proceedings of the 20th International Conference on Computational Linguistics (CoLing-2004)*, pages 315–321, 2004.

Bojar, Ondřej, Christian Buck, Christian Federmann, Barry Haddow, Philipp Koehn, Johannes Leveling, Christof Monz, Pavel Pecina, Matt Post, Herve Saint-Amand, Radu Soricut, Lucia Specia, and Aleš Tamchyna. Findings of the 2014 Workshop on Statistical Machine Translation. In *Proceedings of the Ninth Workshop on Statistical Machine Translation*, pages 12–58, Baltimore, Maryland, USA, June 2014. Association for Computational Lingustics.

Bojar, Ondřej, Rajan Chatterjee, Christian Federmann, Barry Haddow, Chris Hokamp, Matthias Huck, Varvara Logacheva, and Pavel Pecina, editors. *Proceedings of the Tenth Workshop on Statistical Machine Translation*. Association for Computational Linguistics, Lisbon, Portugal, September 2015.

Bojar, Ondřej, Rajen Chatterjee, Christian Federmann, Yvette Graham, Barry Haddow, Matthias Huck, Antonio Jimeno Yepes, Philipp Koehn, Varvara Logacheva, Christof Monz, Matteo Negri, Aurelie Neveol, Mariana Neves, Martin Popel, Matt Post, Raphael Rubino, Carolina Scarton, Lucia Specia, Marco Turchi, Karin Verspoor, and Marcos Zampieri. Findings of the 2016 Conference on Machine Translation. In *Proceedings of the First Conference on Machine Translation*, pages 131–198, Berlin, Germany, August 2016. Association for Computational Linguistics.

Moorkens, Joss and Sharon O'Brien. *Human Issues in Translation Technology: The IATIS Yearbook*, chapter Assessing User Interface Needs of Post-Editors of Machine Translation, pages 109–130. Routledge, Oxford, UK, 2017.

Moorkens, Joss and Andy Way. Comparing Translator Acceptability of TM and SMT outputs. *Baltic Journal of Modern Computing*, 4(2):141–151, 2016.

Moorkens, Joss, Sharon O'Brien, Igor A.L. da Silva, Norma B. de Lima Fonseca, and Fabio Alves. Correlations of perceived post-editing effort with measurements of actual effort. *Machine Translation*, 29(3–4):267–284, 2015.

Papineni, Kishore, Salim Roukos, Todd Ward, and Wei-Jing Zhu. BLEU: a Method for Automatic Evaluation of Machine Translation. In *Proceedings of the Association for Computational Linguistics (ACL)*, pages 311–318, 2002.

Parra Escartín, Carla and Manuel Arcedillo. Machine translation evaluation made fuzzier: A study on post-editing productivity and evaluation metrics in commercial settings. In *Proceedings of the MT Summit XV*, Miami (Florida), October 2015a. International Association for Machine Translation (IAMT).

Parra Escartín, Carla and Manuel Arcedillo. Living on the edge: productivity gain thresholds in machine translation evaluation metrics. In *Proceedings of the Fourth Workshop on Post-editing Technology and Practice*, pages 46–56, Miami, Florida (USA), November 2015b. Association for Machine Translation in the Americas (AMTA).

Snover, Matthew, Bonnie Dorr, Richard Schwartz, Linnea Micciulla, and Makhoul John. A Study of Translation Edit Rate with Targeted Human Annotation. In *Proceedings of Association for Machine Translation in the Americas (AMTA)*, pages 223–231, 2006.

Specia, Lucia, Najeh Hajlaoui, Catalina Hallett, and Wilker Aziz. Predicting Machine Translation Adequacy. In *Proceedings of the 13th Machine Translation Summit*, pages 513–520, Xiamen, China, September 2011.

Specia, Lucia, Gustavo Paetzold, and Carolina Scarton. Multi-level Translation Quality Prediction with QuEst++. In *Proceedings of ACL-IJCNLP 2015 System Demonstrations*, pages 115–120, Beijing, China, July 2015. Association for Computational Linguistics and The Asian Federation of Natural Language Processing.

Turchi, Marco, Matteo Negri, and Marcello Federico. MT Quality Estimation for Computer-assisted Translation: Does it Really Help? In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Short Papers)*, pages 530–535, Beijing, China, July 26–31 2015. Association for Computational Linguistics.

Address for correspondence:
Carla Parra Escartín
carla.parra@adaptcentre.ie
ADAPT Centre, School of Applied Language and Intercultural Studies
Dublin City University
Glasnevin
Dublin 9, Ireland

PBML

The Prague Bulletin of Mathematical Linguistics
NUMBER 108 JUNE 2017 355–366

Improving Machine Translation through Linked Data

Ankit Srivastava, Georg Rehm, Felix Sasaki

German Research Center for Artificial Intelligence (DFKI),
Language Technology Lab, Berlin, Germany

Abstract

With the ever increasing availability of linked multilingual lexical resources, there is a renewed interest in extending Natural Language Processing (NLP) applications so that they can make use of the vast set of lexical knowledge bases available in the Semantic Web. In the case of Machine Translation, MT systems can potentially benefit from such a resource. Unknown words and ambiguous translations are among the most common sources of error. In this paper, we attempt to minimise these types of errors by interfacing Statistical Machine Translation (SMT) models with Linked Open Data (LOD) resources such as DBpedia and BabelNet. We perform several experiments based on the SMT system Moses and evaluate multiple strategies for exploiting knowledge from multilingual linked data in automatically translating named entities. We conclude with an analysis of best practices for multilingual linked data sets in order to optimise their benefit to multilingual and cross-lingual applications.

1. Introduction

Statistical Natural Language Processing (NLP) technologies rely on large volumes of data from which models can be constructed to leverage patterns and knowledge from these data sets. Typically, these resources are in the form of annotated (structured, labeled) or unstructured natural language text such as aligned input and output language paired sentences for Machine Translation (MT) or parsed treebanks for parsing. However, we can observe a certain shortage of NLP systems (Nebhi et al., 2013; Hokamp, 2014) which exploit knowledge from structured or semi-structured resources such as the Linked Open Data (LOD) lexical resources created for and maintained as part of the Semantic Web and its Linked Data Cloud. This shortage is most likely due to the fact that the MT community is primarily focused upon con-

 Corresponding author: ankit.srivastava@dfki.de
Cite as: Ankit Srivastava, Georg Rehm, Felix Sasaki. Improving Machine Translation through Linked Data. The Prague Bulletin of Mathematical Linguistics No. 108, 2017, pp. 355–366. doi: 10.1515/pralin-2017-0033.

tinuously improving their respective rule-based, statistical or neural algorithms and approaches, while the LOD community is focused upon representing, providing and linking data sets. Our contribution is an approach at building a bridge between the two communities.

In this paper, using Statistical Machine Translation (SMT) as a case-study, we explore three strategies for leveraging knowledge from a variety of LOD resources. In addition to analysing the impact of linked data on MT, we briefly discuss considerations for creating and linking multilingual lexical resources on the web so that NLP systems can benefit from them.

This paper is structured as follows. We briefly overview the background technologies (Semantic Web, Resource Description Format, Linked Open Data, SMT workflow) leveraged in this research in Section 2. In Section 3, we outline three strategies for integrating linked data in a SMT system followed by a summary of previous works in Section 4. In Sections 5 and 6 we describe our experimental results and analysis after which we conclude in Section 7.

2. Technologies

In this section, we briefly summarise the technologies used, i. e., Statistical Machine Translation (SMT), Linked Open Data (LOD) resources, and Semantic Web technologies facilitating the integration of SMT with LOD.

2.1. Semantic Web Technologies

With regard to the Semantic Web, several key technologies can be exploited in NLP systems.[1]

RDF (Resource Description Framework) is a formalism to represent data on the web as a labelled graph of triples (subject, predicate, object, or, to put it another way, objects and their relations). URIs (Uniform Resource Identifiers) are compact sequences of characters used to identify resources – including objects – on the web. Ontologies are hierarchical vocabularies of types and relations, allowing more efficient storage and use of data by encoding generic facts about objects. RDF Schema (RDFS) is one such formalism or knowledge representation language, OWL (Web Ontology Language) can be used to represent more complex knowledge structures. RDF and RDFS are the underlying syntax and ontology as well as vocabulary languages, used to represent machine readable data and define relevant properties such as rdfs:label for language name. SPARQL[2] is the query language used to retrieve information from RDF-encoded data including NIF.

[1]The basic technologies, data formats and approaches that constitute the technological building blocks of the Semantic Web and Linked Data are developed and standardised by the World Wide Web Consortium (W3C).

[2]http://www.w3.org/TR/rdf-sparql-query/

The knowledge sources employed in our experiments are structured as Linked Data, stored in RDF (subject-predicate-object triples). In order to access or retrieve information (translations) from the RDF datasets for integration in a MT system, we need to query the database using SPARQL. The example below illustrates a sample SPARQL query for retrieving the German (de) translation of the term "Microsoft Paint."

Listing 1. An example SPARQL query

```
PREFIX dbpedia: <http://dbpedia.org/resource/>

SELECT distinct *
WHERE {
    <http://dbpedia.org/resource/Paint_(software)>
    rdfs:label ?label
    filter langMatches( lang(?label), "de" )
}
```

NIF 2.0[3] (Natural Language Processing Interchange Format) is an RDF-based format that aims to achieve interoperability between NLP tools such as parsers, SMT engines and annotated language resources such as DBpedia. Its joint application with technologies like ITS 2.0[4] (Internationalization Tag Set) and the OntoLex lemon model[5] makes it an ideal candidate to implement multilingual applications. The primary use case of this standard is to serve as an input and output format for web services, that enable seamless pipelining or combination of various language and linked data processing web services in sequence. With regard to NLP, an important characteristic of NIF is that its atomic unit is a character rather than a word. Thus, if a sentence has 23 characters (including spaces between words), the resource or sentence spans from 0 to 22. In this way, NLP pipelines can create fine grained annotations relying on the graph based model of RDF. In order to evaluate effectiveness of LOD in SMT (primary aim of this paper), we integrated our SMT system with NIF input / output wrappers using methodology described in (Srivastava et al., 2016).

In a nutshell, if the data such as a multilingual lexicon is stored as a linked data (NIF / RDF), then SPARQL is a tool to retrieve information from the linked data such as translations in the required target language.

[3]http://persistence.uni-leipzig.org/nlp2rdf/

[4]http://www.w3.org/TR/its20/

[5]https://www.w3.org/2016/05/ontolex/

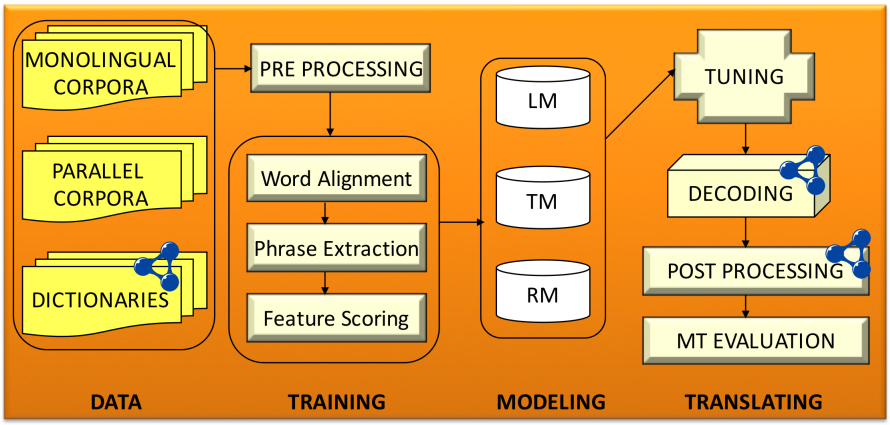


Figure 1. Workflow of the SMT modules

2.2. Machine Translation

There has been an ever increasing interest in Machine Translation, one of the earliest non-numeric applications of computers (Hutchins, 2000), since the enormous increase of multilingual user-generated content on the web. There are a number of approaches to implementing MT (rule-based, example-based, statistical, syntax-based, semantics-based, hybrid, and neural). Statistical MT is the most widely researched paradigm and represents, along with neural MT, the current state of the art.

In this paper, we conduct our experiments using the widely deployed open source SMT system Moses[6] (Koehn et al., 2007). We use the phrase-based SMT system with standard configurations, as specified in Section 5. Several similar approaches exist such as an attempt on integrating bilingual dictionaries in SMT (Arcan et al., 2014).

Figure 1 shows the workflow of a typical SMT system. Data in the form of bilingual (including dictionaries extracted from LOD) and monolingual resources (typically collections of documents tokenised into sentences) is fed into the training network which creates the language model (LM), translation model (TM), and reordering model (RM). These models are then tuned, followed by decoding (the actual translation step), followed by post-processing (such as linked data translation edits).

As shall be described in Section 3, we integrate linked data (illustrated with the blue triangular structure in Figure 1) into the SMT system at three points:

- As dictionaries during training (before Word Alignment)
- As an alternate resource (translation rules) during decoding

[6]http://www.statmt.org/moses/

- As rules in the form of post-editing process

2.3. Linked Data Resources

For the MT improvement, we are going to use three linked data resources: DBpedia, BabelNet, and JRC-names. These three resources are part of the Linguistic Linked Open Data Cloud[7], an interconnected set of linguistic resources represented as linked data. The LLOD cloud helps to address problems in various research and application areas, such as interoperability of linguistic annotations, graph-based annotations based on the linked data graph model without the need for special purpose tools, or fast increase of multilingual resources via ease of linkage. In addition, based on LLOD principles, new formats like OntoLex (Fiorelli et al., 2015) have been put forward.

DBpedia[8] is a linked open dataset (extracted from Wikipedia) consisting of 4.58 million entities in up to 125 languages and 29.8 million links to external web pages. DBpedia has been used in many linked data applications. For the improvement of MT it is useful because of the high number of multilingual labels, and the high number of cross-lingual links between DBpedia instances. DBpedia Spotlight[9] is an open-source tool for automatically annotating mentions of DBpedia resources in text. Note that the translations may be prone to error on account of being user generated.

BabelNet (Navigli and Ponzetto, 2012) is a a multilingual resource created by linking Wikipedia to WordNet and other semantic networks, filling gaps with MT. BabelNet is highly multilingual and, since it encompasses, e. g., DBpedia, we expect an additional improvement of MT compared to using DBpedia only.

JRC-names[10] (Steinberger et al., 2011) is a freely available multilingual named entity resource for person and organisation names that have been compiled from over seven years of analysing multilingual news articles. Since March 2016, JRC-Names has also been available as linked data, including additional information such as frequencies per language, titles found with the entities, and date ranges.

Table 1 gives a comparative evaluation of the languages and sizes of these three resources.

3. Integrating LOD into SMT – Three Approaches

As regards integrating Linked Open Data resource into Machine Translation workflows, we implemented three different strategies (illustrated in Figure 1).

[7]http://linguistic-lod.org/llod-cloud

[8]http://wiki.dbpedia.org

[9]https://github.com/dbpedia-spotlight/

[10]https://ec.europa.eu/jrc/en/language-technologies/jrc-names

Resource	# Entries	# Languages
DBpedia	23.8 million	125
BabelNet	14 million	270
JRC-Names	205 thousand	22

Table 1. Comparison of Linked Data resources

- **Dictionaries:** Transform LD resources into a dictionary for word alignment such that the models will contain knowledge from the Linked Data resource and let the Moses decoder decide which translation knowledge (linked data or parallel corpora) to retrieve.
- **Pre-decoding:** Forced decoding by first named entity linking via SPARQL query (using Moses xml-input exclusive feature).
- **Post-processing:** Automatic post-editing or correcting of untranslated words, i.e. translations which are not present in the translation model.

Note that each of the three algorithms are applied individually to each of the three LOD resources (DBpedia, BabelNet, JRC-named), described in Section 5.

3.1. Algorithm 1: Dictionaries

Each of our LOD resources (DBpedia, BabelNet, JRC-names) is available as a bilingual dictionary on their respective websites. For the dictionary approach, we treat these dictionaries as an additional bilingual terminology dataset and integrate them into the SMT system using well-known methods of adding bilingual terms to the training data before the word and phrase alignment step of training (Bouamor et al., 2012).

3.2. Algorithm 2: Pre-decoding

The term pre-decoding alludes to the fact that the LOD resource is gathered right before calling the SMT decoder. In reality, the linked data resource provides additional translation rules for specific words and phrases (mainly named entities) during decoding. The pre-decoding algorithm inspired by the approach in (Srivastava et al., 2016) is described below:

1. Take as input a source sentence
2. Tag the named entities using an off-the-shelf Named Entity Recogniser
3. For each of the named entities invoke a SPARQL query for the appropriate resource (DBpedia, JRC-names, BabelNet) to retrieve the translation in the target language

4. Integrate these translations in the Moses decoder. Encode the named entity and its translation in a format compatible with the Moses decoder (enabled with the xml-input feature)

Note that all the procedures above are carried out by freely available web service API calls, the source code for which can be found at `https://github.com/freme-project` for FREME web services[11] and at `https://github.com/dkt-projekt` for DKT web services.[12]

3.3. Algorithm 3: Post-processing

As mentioned previously, a major source of error in MT is the presence of unknown words, i.e. entries which do not have a valid translation in the training data. This is particularly true when the SMT system is trained in a domain different from the domain of the test data, as is typical of large-scale evaluations such as the WMT Shared Tasks (Bojar et al., 2016). Our third algorithm identifies the untranslated words[13] and calls a SPARQL query to retrieve the translation (if available) from each of the three LOD resources. The SPARQL Query endpoints are available at:

- DBpedia: `http://de.dbpedia.org/sparql`
- BabelNet: `http://babelnet.org/sparql/`
- JRC-names: `http://data.europa.eu/euodp/en/linked-data`

4. Related Work

We use multiple linked data resources using three different strategies. There have been previous attempts at integrating LOD into SMT, however, to the best of our knowledge, none of these demonstrated all approaches on one dataset like we do in this subsmission. (McCrae and Cimiano, 2013) primarily integrated the dictionary of translations extracted from LOD resources during decoding and created a new feature for linked data. They essentially let the Moses decoder decide when to choose translations from LOD and when to translate from its phrase tables. In contrast to our approach on encoding documents in NIF (while entity linking via SPARQL queries), they employ another ontology called Lemon (Lexicon Model for Ontologies[14]) to translate unknown words, i. e., translations not found by the decoder. Our Algorithm 1 (Dictionaries) is most similar to their approach while we employ an alternative approach to handling unknown words (Algorithm 3 [Post-processing]).

(Du et al., 2016) extracted translations from BabelNet dictionaries using both (McCrae and Cimiano, 2013)'s methods as well as the post processing (Algorithm 3)

[11] Of particular interest is the web service named e-entity/dbpedia-spotlight.

[12] Of particular interest are the services DKTBrokerStandalone/nifTools, e-NLP/Sparqler, and e-SMT.

[13] Moses allows special annotation to highlight the presence of unknown words in the translated output

[14] http://lemon-model.net

Category	Training	Development	Test
Dataset	Europarl v7	newstest 2011	newstest 2012
German–English	1,920,209	3,003	3,003
Spanish–English	1,965,374	3,003	3,003

Table 2. Statistics of parallel corpus used in baseline SMT training experiments

System	BLEU	TER
Baseline	12.30	0.788
DBpedia	12.33	0.776
BabelNet	12.25	0.780
JRC-Names	12.39	0.762

Table 3. Evaluation results on English–German

employed in this contribution to demonstrate modest improvements in translating English-Polish and English-Chinese data.

The pre-decoding approach of locating named entities and forcing their translations from LOD resources (retrieved via SPARQL queries) on to the decoder was inspired by methodology described in (Srivastava et al., 2016).

5. Experiments

We trained the SMT system to translate from English to German and Spanish. The set of parallel sentences for training, and the development and test sets for tuning and testing respectively were sourced from the data provided for the WMT 2012 shared task on MT[15]. This was done mainly to make our experiments comparable to that of (McCrae and Cimiano, 2013). Table 2 gives an overview of the data sizes our models are trained on.

Tables 3 and 4 show the evaluation results of our MT experiments. The Baseline system did not use any linked data of any sort. The two evaluation metrics used are BLEU (Papineni et al., 2002) and TER (Snover et al., 2006).

Contrary to our expectation, BabelNet did not perform as well as other linked data resources. While JRC-Names gave the best performance, probably owing to their data being from the same domain as the test data (news domain). We also believe that

[15]http://www.statmt.org/wmt12/

System	BLEU	TER
Baseline	31.70	0.577
DBpedia	31.03	0.550
BabelNet	30.99	0.558
JRC-Names	31.91	0.540

Table 4. Evaluation results on English–Spanish

BabelNet being the largest resource in terms of size also contained more noise and it was often difficult to disambiguate translations.

6. Analysis of Multilingual Linked Data Sets

Compared to previous approaches, see (McCrae and Cimiano, 2013), our experiments do not provide a high improvement of MT quality. However, we can draw useful conclusions in light of best practices for creating linguistic LOD. The forum for the best practices is the BPMLOD Community Group, see[16]. We examined guidelines in the realm of BPMLOD, for linguistic linked data resources such as BabelNet[17] and bilingual dictionaries[18]. Based on the experiments we conducted, there are a few features which are of importance for applying linguistic LOD in MT.

- Domain Identifier: When a specific term has multiple translations in another language, properties such as the domain would help in disambiguating the context.
- Morphology: When translating into a morphologically richer language, information about the form of a noun changes based on the case can help to improve the translation quality.

In conducting a manual evaluation of the results i.e. having a bilingual German speaker eye a randomly selected subset of the translated outputs, we also discovered that while our systems are useful in disambiguating erroneous translations, the automatic MT evaluation metrics are deficient in counting them such that they do not account for variability in translations. For example the reference translation "MS Paint" only matches partially with the LOD system translation "Microsoft Paint." Algorithm 2 (Pre-decoding) identified and correctly translated 15% more terms (named entities) than the baseline SMT system.

[16]https://www.w3.org/community/bpmlod/

[17]http://www.w3.org/2015/09/bpmlod-reports/multilingual-dictionaries/

[18]http://www.w3.org/2015/09/bpmlod-reports/bilingual-dictionaries/

Source (en):	**MS Paint** is a good option.
Baseline (de):	**Frau Farbe** ist eine gute wahl.
Linked (de):	**Microsoft Paint** ist eine gute wahl.
Reference (de):	**MS Paint** ist eine gute Möglichkeit.

7. Conclusion and future work

In this paper, we demonstrated employing knowledge from three semantic web resources which show modest improvement in English-German and English-Spanish translations. We leave for future work exploiting several more features such as word senses from the knowledge-rich semantic network in MT.

While deep learning-based neural approaches to MT (i. e., NMT: Neural Machine Translation (Sennrich et al., 2016)) have been the state of the art since WMT 2016, we decided to demonstrate our Linked Data-focused approach using SMT due to the lower complexity of the integration task. Future work will include experiments with NMT using our Linked Data-focused approach at improving MT systems. Note that the post-process (Algorithm 3) approach can be theoretically applied to a neural MT system as-is.

It is our belief that the use of Linked Open Data in combination with Named Entity Recognition (Algorithm 2 [Pre-decoding] in our approach) helps reduce the long tail of difficult to translate names. This is similar to word sense disambiguation in MT (Carpuat, 2008). Employing world knowledge for disambiguating terms other than named entities is another potential direction for future research.

This paper is a step towards making MT semantic web-aware and it is our hope that more MT researchers undertake integration of this fertile knowledge source.

Acknowledgements

We would like to thank the anonymous reviewers for their insightful and helpful comments. The project Digitale Kuratierungstechnologien (DKT) is supported by the German Federal Ministry of Education and Research (BMBF), Unternehmen Region, instrument Wachstumskern-Potenzial (No. 03WKP45). More information on the project can be found online at `http://www.digitale-kuratierung.de`.

Bibliography

Arcan, Mihael, Marco Turchi, Sara Tonelli, and Paul Buitelaar. Enhancing Statistical Machine Translation with bilingual terminology in a CAT environment. In *11th Conference of the Association for Machine Translation in the Americas*, pages 54–68, 2014.

Bojar, Ondrej, Christian Buck, Rajen Chatterjee, Christian Federmann, Liane Guillou, Barry Haddow, Matthias Huck, Antonio Jimeno Yepes, Auralie Navaol, Mariana Neves, Pavel Pecina, Martin Popel, Philipp Koehn, Christof Monz, Matteo Negri, Matt Post, Lucia Specia, Karin Verspoor, Jorg Tiedemann, and Marco Turchi, editors. *Proceedings of the First Confer-*

ence on Machine Translation. Association for Computational Linguistics, Berlin, Germany, August 2016. URL http://www.aclweb.org/anthology/W/W16/W16-2200.

Bouamor, Dhouha, Nasredine Semmar, and Pierre Zweigenbaum. Identifying bilingual Multi-Word Expressions for Statistical Machine Translation. In Calzolari, Nicoletta, Khalid Choukri, Thierry Declerck, Mehmet Ugur Dogan, Bente Maegaard, Joseph Mariani, Jan Odijk, and Stelios Piperidis, editors, *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC-2012)*, pages 674–679, Istanbul, Turkey, May 2012. European Language Resources Association (ELRA). ISBN 978-2-9517408-7-7. URL http://www.lrec-conf.org/proceedings/lrec2012/pdf/886_Paper.pdf. ACL Anthology Identifier: L12-1527.

Carpuat, Marine Jacinthe. *Word Sense Disambiguation for Statistical Machine Translation*. PhD thesis, 2008. AAI3350676.

Du, Jinhua, Andy Way, and Andrzej Zydron. Using BabelNet to Improve OOV Coverage in SMT. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016)*, 2016.

Fiorelli, Manuel, Armando Stellato, John P. Mccrae, Philipp Cimiano, and Maria Teresa Pazienza. LIME: The Metadata Module for OntoLex. In *Proceedings of the 12th European Semantic Web Conference on The Semantic Web. Latest Advances and New Domains - Volume 9088*, pages 321–336, New York, NY, USA, 2015. Springer-Verlag New York, Inc. ISBN 978-3-319-18817-1. doi: 10.1007/978-3-319-18818-8_20. URL http://dx.doi.org/10.1007/978-3-319-18818-8_20.

Hokamp, Chris. Leveraging NLP Technologies and Linked Open Data to Create Better CAT Tools. In *International Journal of Localisation, Vol 14*, pages 14–18, 2014.

Hutchins, John. *John W. Hutchins (Eds.), Early Years in Machine Translation*, chapter The first decades of Machine Translation: overview, chronology, sources, pages 1–16. John Benjamins B. V., 2000.

Koehn, Philipp, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondrej Bojar, Alexandra Constantin, and Evan Herbst. Moses: Open Source Toolkit for Statistical Machine Translation. In *Proceedings of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions*, ACL '07, pages 177–180, Stroudsburg, PA, USA, 2007. Association for Computational Linguistics. URL http://dl.acm.org/citation.cfm?id=1557769.1557821.

McCrae, John and Philipp Cimiano. Mining Translations from the Web of Open Linked Data. In *Proceedings of the Joint Workshop on NLP, LOD and SWAIE*, pages 8–11, 2013.

Navigli, Roberto and Simone Paolo Ponzetto. BabelNet: The Automatic Construction, Evaluation and Application of a Wide-Coverage Multilingual Semantic Network. In *Artificial Intelligence*, pages 217–250, 2012.

Nebhi, Kamel, Luka Nerima, and Eric Wehrli. NERTIS - A Machine Translation Mashup System using Wikimeta and DBpedia. In *Semantic Web (ESWC) 2013 Satellite Events*, pages 312–318, 2013.

Papineni, Kishore, Salim Roukos, Todd Ward, and Wei-Jung Zhu. BLEU: A Method for Automatic Evaluation of Machine Translation. In *40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, 2002.

Sennrich, Rico, Barry Haddow, and Alexandra Birch. Edinburgh Neural Machine Translation Systems for WMT 16. In *Proceedings of the First Conference on Machine Translation*, pages 371–376, Berlin, Germany, August 2016. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/W/W16/W16-2323.

Snover, Matthew, Bonnie Dorr, Richard Schwartz, Linnea Micciula, and John Makhoul. A Study of Translation Edit Rate with targeted Human Annotation. In *7th Conference of the Association for Machine Translation in the Americas*, pages 223–231, 2006.

Srivastava, Ankit, F. Sasaki, P. Bourgonje, J. Moreno-Schneider, J. Nehring, and G. Rehm. How To Configure Statistical Machine Translation with Linked Open Data Resources. In *Proceedings of the 38th Annual Translating and Computer Conference*, TC 38, 2016.

Steinberger, Ralf, Bruno Pouliquen, Mijail Kabadjov, Jenya Belyaeva, and Erik van der Goot. JRC-NAMES: A Freely Available, Highly Multilingual Named Entity Resource. In *Proceedings of the International Conference Recent Advances in Natural Language Processing 2011*, pages 104–110. Association for Computational Linguistics, 2011. URL http://aclweb.org/anthology/R11-1015.

Address for correspondence:
Ankit Srivastava
ankit.srivastava@dfki.de
DFKI GmbH
Alt-Moabit 91c
10559 Berlin, Germany

PBML

The Prague Bulletin of Mathematical Linguistics
NUMBER 108 JUNE 2017

INSTRUCTIONS FOR AUTHORS

Manuscripts are welcome provided that they have not yet been published elsewhere and that they bring some interesting and new insights contributing to the broad field of computational linguistics in any of its aspects, or of linguistic theory. The submitted articles may be:

- long articles with completed, wide-impact research results both theoretical and practical, and/or new formalisms for linguistic analysis and their implementation and application on linguistic data sets, or
- short or long articles that are abstracts or extracts of Master's and PhD thesis, with the most interesting and/or promising results described. Also
- short or long articles looking forward that base their views on proper and deep analysis of the current situation in various subjects within the field are invited, as well as
- short articles about current advanced research of both theoretical and applied nature, with very specific (and perhaps narrow, but well-defined) target goal in all areas of language and speech processing, to give the opportunity to junior researchers to publish as soon as possible;
- short articles that contain contraversing, polemic or otherwise unusual views, supported by some experimental evidence but not necessarily evaluated in the usual sense are also welcome.

The recommended length of long article is 12–30 pages and of short paper is 6–15 pages.

The copyright of papers accepted for publication remains with the author. The editors reserve the right to make editorial revisions but these revisions and changes have to be approved by the author(s). Book reviews and short book notices are also appreciated.

The manuscripts are reviewed by 2 independent reviewers, at least one of them being a member of the international Editorial Board.

Authors receive a printed copy of the relevant issue of the PBML together with the original pdf files.

The guidelines for the technical shape of the contributions are found on the web site `http://ufal.mff.cuni.cz/pbml`. If there are any technical problems, please contact the editorial staff at `pbml@ufal.mff.cuni.cz`.