The Prague Bulletin of Mathematical Linguistics
NUMBER 96 OCTOBER 2011

EDITORIAL BOARD

Editor-in-Chief

Eva Hajičová

Editorial staff

Martin Popel
Ondřej Bojar

Editorial board

Nicoletta Calzolari, Pisa
Walther von Hahn, Hamburg
Jan Hajič, Prague
Eva Hajičová, Prague
Erhard Hinrichs, Tübingen
Aravind Joshi, Philadelphia
Philipp Koehn, Edinburgh
Jaroslav Peregrin, Prague
Patrice Pognan, Paris
Alexander Rosen, Prague
Petr Sgall, Prague
Marie Těšitelová, Prague
Hans Uszkoreit, Saarbrücken

Published twice a year by Charles University in Prague

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

© 2011 PBML. All rights reserved.

PBML

The Prague Bulletin of Mathematical Linguistics
NUMBER 96 OCTOBER 2011

CONTENTS

Articles

© 2011 PBML. All rights reserved.

PBML

The Prague Bulletin of Mathematical Linguistics
NUMBER 96 OCTOBER 2011 5–26

An attractive game with the document: (im)possible?

Barbora Hladká, Jiří Mírovský, Jan Kohout

Charles University in Prague, Institute of Formal and Applied Linguistics

Abstract

The annotation experience we have acquired while participating in the Prague treebanking projects provides us with a strong evidence to conclude that the linguistic data annotation by experts is a very intensive and expensive process. No surprise that we care whether we can get the annotated data in a less demanding process. We focus on an alternative way of annotation to generate the data for natural language processing tasks that either have not been implemented yet or have been implemented with a performance lower than human performance. To be more specific, we are interested in ways of annotation gathered mostly under the terms 'crowdsourcing' and 'human computation', i.e. we concentrate on activities that motivate as many non-experts as possible to devote whatever they prefer (effort, time, enthusiasm, responsibility, etc.) to carry out annotation.

In this paper, we review the notion of crowdsourcing, namely we turn our attention to crowdsourcing projects that manipulate textual data. As we are delighted with the games with a purpose, we carry out an implementation of the on-line games with texts. We introduce a game on coreference, *PlayCoref*, and games with words and white spaces in the sentence, *Shannon Game* and *Place the Space*, in great details. The game rules are designed to be language independent and the games are playable with both Czech and English texts by default. After a number of sessions played so far we revise our initial expectations and enthusiasm to design an attractive annotation game with a document.

1. Introduction

The Prague dependency treebanks[1] represent the annotation projects where both textual and spoken data have been annotated by experts. The annotation framework

[1] http://ufal.mff.cuni.cz/pdt.html

© 2011 PBML. All rights reserved.
Corresponding author: hladka@ufal.mff.cuni.cz
Cite as: Barbora Hladká, Jiří Mírovský, Jan Kohout. An attractive game with the document: (im)possible?. The Prague Bulletin of Mathematical Linguistics No. 96, 2011, pp. 5–26. doi: 10.2478/v10108-011-0008-z.

has a solid theoretical background, namely the Functional Generative Description (FGD, Sgall et al. (1986)), thus the annotation guidelines are coordinated with this theory. Consequently, the annotators are trained according to the guidelines.

The FGD conceives a language as a system of layers, so the Prague treebanking annotation schemes respect this system in such a way that the data is annotated on three layers going from the simplest morphological one through the syntactic-analytical one to the most complex tectogrammatical one. The higher the layer, the higher requirements on the annotator's qualification are expected. While the annotation on the morphological layer can be performed by secondary school students, the annotation on the tectogrammatical layer can be performed by linguists and carefully trained students of the philological studies mainly. The quality of the annotated data must be pursued while formulating the annotation strategy, i.e. criteria to ensure a high quality annotation must be elaborated and a proper number of annotators must be selected. Most of the annotation projects are scheduled at least for five years and the number of people involved in them varies. In average, up to ten member teams are established including annotators and technical staff.

Summing up the annotation experience, we conclude that the linguistic data annotation by experts is a *labour* and *time* and *money* consuming process.[2] No surprise that we care whether we can get the annotated data in a less expensive process. At the same time, we ask *Do we really need (more) annotated data?* Considering data-driven approaches to address natural language processing tasks, the positive answer is replied every time the correlation between the performance and the volume of data needed is evaluated.

In this paper, we focus on an alternative way of annotation to provide the data for NLP tasks that either have not been implemented yet or have been implemented with a performance lower than human performance. To be more specific, we are interested in ways of annotation gathered mostly under the terms 'crowdsourcing' and 'human computation'. One can encounter many other synonyms but we will use these two terms throughout the paper.

The paper is organized as follows. In Section 2, we briefly introduce the notion of crowdsourcing. As we are immersed in the textual data annotation, we turn our special attention to the crowdsourcing projects with texts. At this point, we are very close to the topic of the paper (Wang et al., 2010) discussing the phenomenon of crowdsourcing in NLP for the first time, at least to our knowledge. We will summarize it and add our points of view. We are delighted with the games with a purpose so much that we conceive them as a possible way of textual data annotation. We have proposed and implemented a number of games that are described in detail in Section 3. We conclude with Section 4.

[2] This conclusion is valid for the data in general.

2. Crowdsourcing/Human computation

The online encyclopedia Wikipedia[3] is an exemplary crowdsourcing/human computation system so we list its definition of crowdsourcing and human computation:

- **Crowdsourcing** is the act of outsourcing tasks, traditionally performed by an employee or contractor, to an undefined, large group of people or community (a "crowd"), through an open call.
- **Human-based computation** is a computer science technique in which a computational process performs its function by outsourcing certain steps to humans. This approach uses differences in abilities and alternative costs between humans and computer agents to achieve symbiotic human-computer interaction.

We interpret the distinction between these two terms as follows: the human computation (HC) is the qualification of crowdsourcing to computer-based issues. It is not our intention to discuss the definitions in details. Instead, we refer to a number of more profound resources, like (Crowdsourcing.org, 2011), (Doan et al., 2011), (Ipeirotis and Paritosh, 2011).

The human computation systems can be classified along many dimensions, see e.g. (Quinn and Bederson, 2009), (Yuen et al., 2009). Here, we highlight two of them:

1. The *nature of collaboration*. We are mainly interested in the classes of Games With A Purpose (GWAP), Highly Intelligence Tasks (HITs) hosted by Amazon's Mechanical Turk and Wisdom of the Crowds (WotC) systems. The nature of collaboration closely relates to the motivation to collaborate. The three mentioned classes exemplify motivation by fun, profit and enthusiasm to share knowledge, respectively.
2. The *input data type*. The users absorb the information provided by the input data through different activities like observing the picture, watching the video, listening to music, reading the web page, etc. Each of these activities takes some time the amount of which strongly depends on the input data type. For example, image content understanding takes much less time than understanding of paragraph content.

As long as we search for an alternative way of textual data annotation, we review HC systems that manipulate with the textual data, i.e. either individual words, sentences, paragraphs or even whole documents. We list GWAPs first, then HITs and finally WotC.

- **Jinx**, a two player game, (Seemakurty et al., 2010), shows the players a context, usually a sentence, with an underlined word. The players enter synonyms for the underlined word and attempt to match each other. The output synonym sets are tested against the WordNet (Miller, 1995) and the game data presents a valuable data for a task of word sense disambiguation.

[3] `http://www.wikipedia.org/`

- **Onto Games**, (Siorpaes and Simperl, 2010), create a semantic content. Articles from Wikipedia are presented during the sessions and players answer the pre-generated questions concerning the onthology concepts.
- **PackPlay**, (Green et al., 2010), is a game framework consisting of the *Entity Discovery* game and *Named That Entity* game. The players are asked to annotate named entities in the sentences.
- **Page Hunt**, a single-player game, (Ma et al., 2009), shows the player a random web page (its contents, not its web address) and the player is supposed to ask such a query that brings a given page in the top N results on a search engine. The queries from the winning trials can be used as terms in a task of query alternation.
- **Phrase Detectives**, a single-player game (Chamberlain et al., 2008), traces a relationship between words and phrases in a short text, namely the relationship of coreference. We present details of the game description in Section 3.1.
- **Verbosity**, a two-player game, (von Ahn et al., 2007), generates common sense facts so that one player gets a secret word and provides the hints in a form of sentence templates to the second player that guesses the secret word.
- **Amazon's Mechanical Turk** is an online job market hosting so-called highly intelligence tasks (HITs). Browsing the HITs with textual data, we meet mostly machine translation tasks, tasks like 'write a sentence with a given phrase' or 'write a summary of an article'. (Snow et al., 2008) investigated HITs on affect recognition, word similarity, recognizing textual entailment, event temporal ordering, and word sense disambiguation. They showed high agreement between Mechanical Turk non-expert annotations and existing gold standard labels provided by expert labelers. Similarly, a study by (Kittur et al., 2008) compares the rating of Wikipedia's articles assessed by both Mechanical Turkers and Wikipedia admins. The two experiments they conducted differ in a feature that enables verification how much the Mechanical Turkers are familiar with the content of what they are rating, i.e. how carefully they are reading the articles. They conclude that the Mechanical Turk is a promising platform for conducting various tasks, but special care must be taken in the design of the tasks to avoid unfair processing, especially if the tasks are subjective or qualitative.
- **Wikipedia** is a freely accessible online encyclopedia that everyone can change. It represents the only HC system that works with whole documents.
- **reCAPTCHA** is a system enabling to improve the quality of digitalized books (von Ahn et al., 2008). It iss designed as an upgrade of CAPTCHA system that recognizes whether a person (not computer) is responsing. The recognition runs like a test to rewrite a distorted string of characters exactly. reCAPTCHA submits two character strings, one of them digitally recognized correctly and the other one unrecognized. A user has to rewrite both strings correctly. The system of reCAPTCHA can be classified as WotC system with the attribute 'no other

choice' since the users simply have to rewrite strings to proceed their further web activities.

At least to our knowledge, there is no HC system guiding the user to **carefully** read a document and do some annotation. This fact and our sympathy to the Games with A Purpose methodology strongly motivate us to design and implement such a system.

3. Play the Language Games

We have implemented three games with textual data and published them at `http://www.lgame.cz` portal. The subsection (3.1) describes the PlayCoref game – a game with coreference. It is the only game out of the three that is meant to produce linguistically valuable data. The subsequent subsections (3.2) and (3.3) describe two remaining games – Shannon game and Place the Space, respectively. Their primary purpose is to attract people to this game portal.

We use *A Study in Scarlet* by Sir Arthur Connan Doyle to present the input data into the sessions of all three games. The choice has been made for practical reasons, namely the novel is publicly available and has been translated into many languages; moreover, a free English audio book exists. The book is not difficult to read and it is enjoyable. The English version comes from the Guttenberg project[4] and the Czech translation comes from the portal Literární doupě[5]. The raw data undertook some processing that we specify in the *Game data preparation* sections below.

3.1. PlayCoref

The PlayCoref is a single-player and two-player game with text. During a 5 minute session, the players read a short text and connect words that co-refer. Their task is to connect all co-referring words in as many sentences as possible.

Notion of coreference Let us present the terminology we use: a *referent* is an object referred to in the given text. A *referring expression* is a lexical representation of a referent. *Coreference* occurs when several referring expressions in the text refer to the same entity (e.g. person, thing, fact). A *coreferential pair* (link) is marked between subsequent pairs of the referring expressions. A sequence of coreferential pairs referring to the same entity forms a *coreference chain*.

In the passage from (Doyle, 1887, 2005), one can read the following coreference chain: *I, I, me, I, me man*; another coreference chain *someone, Stamford, who, dresser* can be seen there: *On the very day that **I** had come to this conclusion **I** was standing at the*

[4] `http://www.gutenberg.org/ebooks/244`

[5] `http://ld.johanesville.net/doyle-06-studie-v-sarlatove`

*Criterion Bar, when someone tapped **me** on the shoulder, and turning round. **I** recognized young Stamford who had been a dresser under **me** at Barts. The sight of a friendly face in the great wilderness of London is a pleasant thing indeed to a lonely **man**.*

Simplicity Our primary goal was to design the game as enjoyable as possible, and thus to attract the greatest possible number of the Internet users. In order to make the game attractive, we have simplified the understanding of coreference so that we do not burden the players with linguistic definitions. Instead, the players are encouraged to follow their language instinct in deciding what corefers in the text.

The coreferential links are undirected and we restrict the part-of-speech classes of coreferential pair members only to coreference-relevant classes, for details see below 3.1 and (Hladká et al., 2009b); words of coreference-irrelevant part of speech classes are locked. A simple algorithm for the detection of a few types of the closest multi-word expressions is applied. Thus, for example, *Sherlock Holmes* is presented to the players as a single annotation unit.

The game The game starts with several first sentences of the document displayed in the players' sentence window – see Figure 1. Unlocked words, i.e. potential members of coreferential pairs, are emphasized (here in black, e.g. *I*, *Sherlock Holmes*, *landlady*, *my*…), while the locked words (e.g. *good* or *usual*) are displayed in gray.

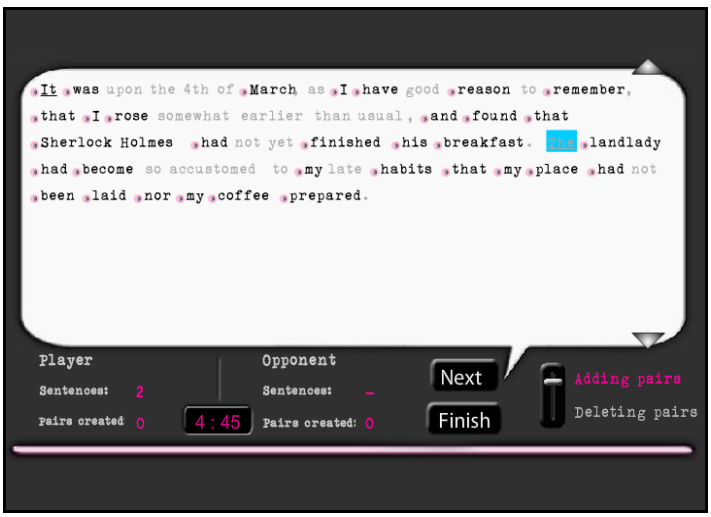


Figure 1. The PlayCoref game starts.

The players mark coreferential pairs as undirected links in the `Adding` mode simply by clicking on dots placed before the active words – see Figure 2 where the player has already created five links. Afterwards, he clicks the button `Next`, another sentence appears and the player adds more links. Links can be deleted in a similar way after switching to the `Deleting` mode.

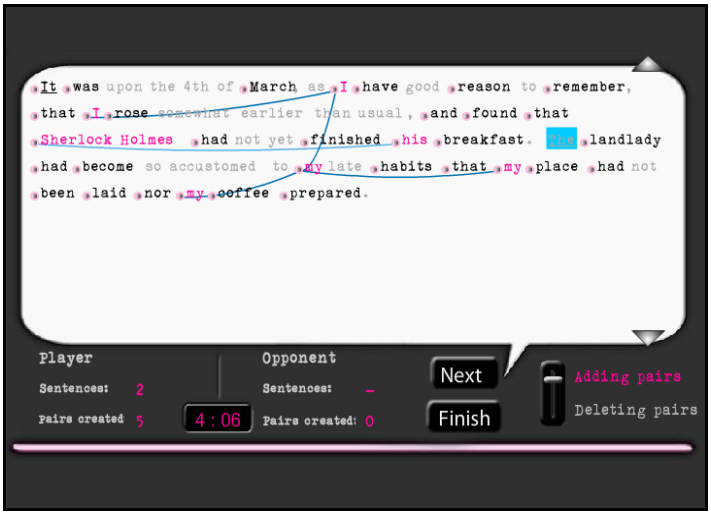


Figure 2. PlayCoref session: Adding links.

Whenever the player finishes pairing words in a visible part of the document (visible to him), he asks for the next sentence of the document by clicking the `Next` button. The sentence appears at the bottom of his sentence window, the first word of the added sentence is highlighted so that it can be recognized immediately. In this manner, the session goes on until the end of the session time (5 minutes) or until the player (both the players in the two-player version of the game) reaches the end of the document (no more sentences are offered and the button `Next` becomes inactive) and he decides to finish the session by clicking the `Finish` button.

During the session, the player has information about the remaining time, the number of his and the opponent's displayed sentences and the number of his and the opponent's created pairs. Revealing more information about the opponent's actions would affect the independency of the players' decisions. Especially, no running score is being presented during the game. Otherwise, the players might adjust their decisions according to the changes in the score, which is undesirable. See our elaboration on the

interactivity issues in (Hladká et al., 2009a). In the single-player version, naturally, no information about the opponent is available.

Figure 3 shows a possible situation of the game closely before its end. The player has already asked for several more sentences, so they do not fit into the window – the text can be scrolled up and down using the arrows on the right side or the mouse wheel. `Deleting` mode is active.

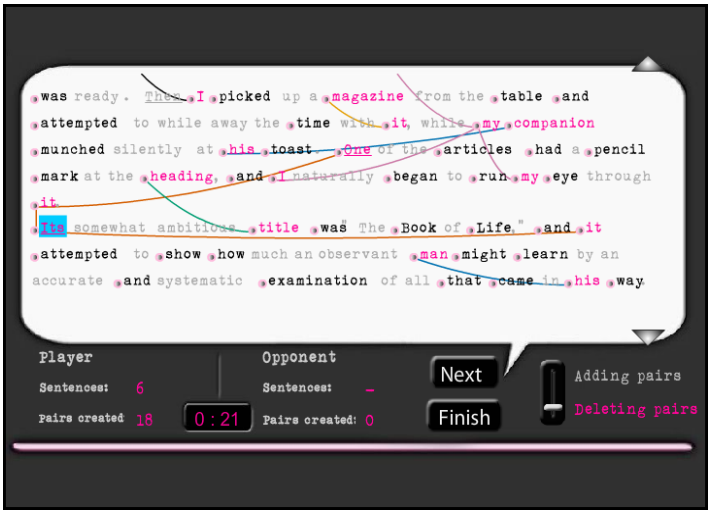


Figure 3. PlayCoref session closesly before its end. `Deleting` mode is active.

At the end of the session – see Figure 4, the result of the game is displayed. It contains information about the player: his final numbers of links, and, of course, his score (the scoring function is described below). In the two-player version, results for both the players are displayed.

Game data preparation In principle, any document can be used in the game, but the following processing steps are necessary.

Tagging The morphological tagging, usually preceded by tokenization, is required to recognize part-of-speech classes and sub-part-of-speech categories (if needed),

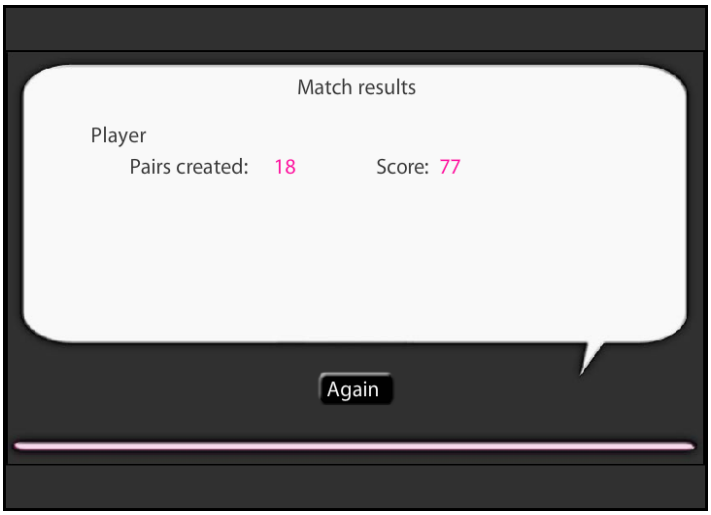


Figure 4. A result of the single-player version of PlayCoref.

in order to lock/unlock individual words for the game. For most languages, tagging is a well solved problem (e.g. for Czech the MORČE tagger[6], for English TnT tagger[7]).

Word locking Words of coreference-relevant POS classes are allowed to become parts of coreferential links marked between individual words or short named entities only. Coreferential pairs that link larger text parts (like several sentences) are disregarded since their marking would be too complex for the players.

We specify the coreference-irrelevant POS classes first. Then the particular words get locked and they are graphically distinguished, so that the players will not consider them at all during the sessions. For English, we work with the PennTreebank tagset (Marcus et al., 1993) and we lock words that are assigned with one of the following POS tags: DT (determiner), IN (preposition or subordinating conjunction), TO (*to*), RB (adverb), RP (particle), JJ (adjective). For Czech positional tag system (Zeman et al., 2005), Table 3.1 shows a list of locked sub-part-of-speech classes of pronouns. Some other POS classes get locked as well: A (adjective), C (numeral), D (adverb), I (interjection), R (preposition), T (particle), and Z (punctuation). So only nouns, selected pronouns, conjunctions and verbs are available for linking in the sessions with Czech texts.

[6] `http://ufal.mff.cuni.cz/morce`

[7] `http://www.coli.uni-saarland.de/~thorsten/tnt/`

Locked pronouns: subPOS and its description	
D	Demonstrative (”ten”, ”onen”, ..., lit. ”this”, ”that”, ”that”, ... ”over there”, ...)
E	Relative ”což” (corresponding to English which in subordinate clauses referring to a part of the preceding text)
L	Indefinite ”všechen”, ”sám” (lit. ”all”, ”alone”)
O	”svůj”, ”nesvůj”, ”tentam” alone (lit. ”own self”, ”not-in-mood”, ”gone”)
Q	Relative/interrogative ”co”, ”copak”, ”cožpak” (lit. ”what”, ”isn’t-it-true-that”)
W	Negative (”nic”, ”nikdo”, ”nijaký”, ”žádný”, ..., lit. ”nothing”, ”nobody”, ”not-worth-mentioning”, ”no”/”none”)
Y	Relative/interrogative ”co” as an enclitic (after a preposition) (”oč”, ”nač”, ”zač”, lit. ”about what”, ”on”/”onto” ”what”, ”after”/”for what”)
Z	Indefinite (”nějaký”, ”některý”, ”číkoli”, ”cosi”, ..., lit. ”some”, ”some”, ”anybody’s”, ”something”)

Table 1. List of pronoun sub-POS classes in the Czech positional tag system locked in PlayCoref.

Automatic and manual coreference annotation For calculating the players’ score (see below), some approximation of the correct solution is needed. If an automatic procedure for coreference resolution (ACR) is available for a language, it can be used. In our experience, however, all available ACR algorithms (both for English and Czech) perform very poorly[8] and cannot be used as a reasonable basis for the scoring function. Until another satisfactory way is found, we present to the game sessions data that is manually annotated, which is a sufficient solution for the initial experiments with the game.

A raw text format of Doyle’s novel was processed by a sequence of tools performing sentence segmentation, tokenization, morphological analysis, tagging, syntactical parsing and semantic parsing, using modules from the TectoMT system (Žabokrtský et al., 2008), and for Czech also the tool-chain from the CAC 2.0 CD-ROM (Hladká et al., 2008). Then two annotators trained for the coreference annotation [9] annotated

[8]For English, we tried Reconcile (Stoyanov et al., 2010), OpenNLP (`http://opennlp.sourceforge.net/models.html`), GuiTAR (`http://cswww.essex.ac.uk/Research/nle/GuiTAR/gtarNew.html`), and BART (Versley et al., 2008); some of the tools did not work at all, the others performed very poorly, especially on the text with dialogues. For Czech, there are almost no tools for ACR. The only one we know, (Novák, 2010) performs very poorly as well.

[9]Two students who participate in the project of coreference and bridging anaphora annotation in the Prague Dependency Treebank (Nedoluzhko et al., 2009)

coreferential links on the tectogrammatical layer. In English, this does not make much difference from the annotation on the surface layer, but in Czech, which is a pro-drop language, some post-processing had to be done. On the tectogrammatical layer in Czech, omitted pronouns are reconstructed and they naturally become parts of coreferential links/chains. As PlayCoref works on the surface layer, omitted pronouns have to be removed from the coreferential chains.

Figure 5 shows an example of a coreferential chain from which a reconstructed pronoun (omitted on the surface) needs to be removed during the transformation of the coreference annotation to the surface form of the text. It is an automatically parsed sentence (actually, two sentences incorrectly parsed as one): *„Tak je to správné.“ „Ano, je, ale přehánět se to nesmí.“*, in English literally: *“It is right so.” “Yes, [it] is, but it must not be exaggerated.“* The three pronouns *it* form a coreferential chain, however the middle one is omitted in the surface form of the Czech sentence. It has to be removed from the coreferential chain. Thus, a new coreferenctial link is created between the two remaining pronouns.

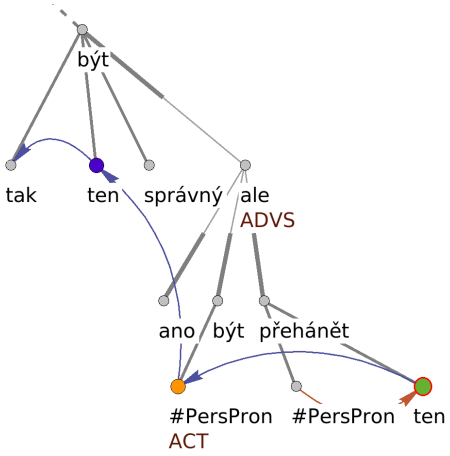


Figure 5. A reconstructed pronoun to be removed from a coreferential chain. In Czech, the two pronouns ”ten” are connected via the reconstructed pronoun (marked with #PersPron and ACT). During the transformation of the coreference annotation to the surface, a direct link between the two pronouns “ten“ is created.

Players' score We want to obtain a large volume of data and thus we must first attract the players and motivate them to play the game again and again. As a reward for their effort, we present scoring. We hope that the players' appetite to win, to confront with

their opponents and to place well in the long-term top scores tables correlate with our research aims and objectives.

Our goal is to ensure the highest quality of the annotation (see also (Hladká et al., 2009a)). The scoring function should reflect the data quality and thus motivate the players to produce correct data. The agreement with the manual expert annotation would be a perfect scoring function. However, the manual annotation is not available for all languages and above all, it is not our goal to annotate data already annotated.

An automatic coreference resolution procedure with a decent accuracy might serve as a first approximation for the scoring function (but as mentioned before, such procedures are not available). As the procedure makes errors, we need to add another component. We suppose that most of the players will mark the coreferential pairs reliably. Then an agreement between the players' pairs indicates correctness, even if the pair differs from the output of the automatic coreference resolution procedure. Therefore, the inter-player agreement becomes the second component of the scoring function. To motivate the players to ask for more parts of the text (and not only "tune" links in the initially displayed sentences), the third component of the scoring function awards the number of created coreferential links.

Scoring function After the game ends, coreference links are automatically checked for circles. If there are some, superfluous links are removed. Otherwise, the circles would harm the scoring function.

In the two-player version of the game, the players get scored (see also (Hladká et al., 2009b)) for their coreferential pairs according to the equation

$$\begin{aligned} score(Player_A) = \lambda_1 * F(Player_A, ACR \text{ or } Manual) \\ +\lambda_2 * F(Player_A, Player_B) \\ +\lambda_3 * min(12, sntncs)/12, \end{aligned}$$

where F stands for the $F - measure = \frac{2*Precision*Recall}{Precision+Recall}$. If the manual annotation is available, we check the player's annotation against it (i.e. we compute $F(Player_A, Manual)$). If a decent automatic coreference resolution method were available, we might check the player's solution against its output (i.e. we would compute $F(Player_A, ACR)$); $sntncs$ is the number of sentences used by the player in the game session. We include the ratio $min(12, sntncs)/12$ as a motivation parameter to inspire players to mark pairs in at least 12 sentences. We have selected the threshold of 12 sentences empirically, which is a reasonable number of sentences the players are able to read and process during the session time. Weights $0 \leq \lambda_1, \lambda_2, \lambda_3 \leq 1, \lambda_1, \lambda_2, \lambda_3 \in R$ (summing to 1) are set empirically.

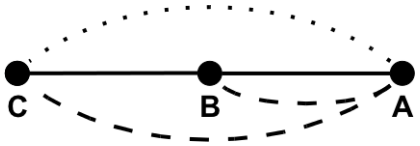


Figure 6. Player '1' pairs (A,C) – the dotted curve; player '2' pairs (A,B) and (B,C) – the solid lines; player '3' pairs (A,B) and (A,C) – the dashed curves. Although players '1' and '2' do not agree on the coreferential pairs at all, '1' and '3' agree only on (A,C) and '2' and '3' agree only on (A,B), for the purposes of the coreference chains reconstruction, the players' agreement is higher: players '1' and '2' agree on two members of the coreferential chain: A and C, players '1' and '3' agree on A and C as well, and players '2' and '3' achieved agreement even on all three members: A, B, and C.

In the single-player version of the game, the scoring function is similar – it only lacks the second member:

$$\begin{aligned} \mathrm{score}(\mathrm{Player_A}) = \lambda_1 * \mathrm{F}(\mathrm{Player_A}, \mathrm{ACR\ or\ Manual}) \\ +\lambda_2 * \mathrm{min}(12, \mathrm{sntncs})/12, \end{aligned}$$

During the calculation of the F-measure, the links of the "annotation" that we compare to are treated as a transitive relation. This solves the issues depicted and described in Figure 6. It does not matter whether the player connects the corefering words as a linear chain or as a star (all to one); also, omitting a word in the chain does not mean a complete disagreement.

Interactivity Issues The degree of the player-to-player interactivity contributes to the attractiveness of the game. From the player's point of view, the more interactivity, the better. For example, knowing both his and the opponent's running score would be very stimulating for the mutual competitiveness. From the linguistic point of view, once any kind of interaction is allowed, statistically pure independency between the players' decisions is lost. A reasonable trade-off between the interactivity and the independency must be achieved. Interactivity that would lead to cheating and decreasing the quality of the game data must be avoided.

Allowing the players to see their own running score would lead to cheating. The players might adjust their decisions according to the changes in the score. Another possible extension of interactivity that would lead to cheating is highlighting words that the opponent used in the coreferential pairs. The players might then wait for the opponent's choice and, again, adjust their decisions accordingly.

Such game data would be strongly biased. However, we still believe that a slight idea of what the opponent is doing can boost inter-coder agreement and yet avoid cheating. Revealing the information about the opponent's number of pairs and the

number of displayed sentences offers at least a little interactivity, yet it will not harm the quality of the data.

Comparison with Phrase Detectives At least to our knowledge, there are no other GWAPs dealing with the relationship among words in a text like PhraseDetectives and PlayCoref. Let us mention some important differences between these two games.

The main difference is in the basic principle of the games: PhraseDetectives game offers the player a whole paragraph and asks him one specific question at a time, e.g. "Are these two words coreferential?", or "Does this word co-refer with another word in the previous text? If so, with which one?". PlayCoref, on the other hand, presents the text to the player sentence by sentence and asks him to search for all coreferential relations in it. Table 2 offers a comparison of various features of the games.

PlayCoref	*PhraseDetectives*
detection of coreference chains	anaphora resolution
single or two-player game	single-player game
a document presented sentence by sentence	a paragraph presented at once
one text in several sessions	checking the pairs marked in the previous sessions
pairing not restricted to the position in the text	the closest antecedent
simple instructions	players training
scoring with respect to the automatic coreference resolution and to the opponent's pairs	scoring with respect to the players that played with the same document before
coreferential pairs correction	no corrections allowed

Table 2. PlayCoref vs. PhraseDetectives.

The very first sessions played We organized the very first PlayCoref competition as an associated event of the CLARA Course on Treebank Annotation.[10] The course participants were either computational linguistics graduates or research associates. In 10 days, 9 different players played 46 sessions that resulted in 945 coreferential pairs in 451 sentences.

We have measured the agreement between each player and the manual annotation and between the players. We use a very similar measure technique as in the scoring

[10] http://lgame.ms.mff.cuni.cz/lgame/sb/competition.php

	F_{chains} (%)
$Player_1$	75 | 57,9 | 69.5 | 72.1 | 75 | 62.6 | 56.3 | 56.1 | 32.1 | 38.6 | 55.8
$Player_2$	54.8 | 78.7 | 75 |81.6 | 79.9 | 72.7 | 55.3 | 68.6 | 68.5 | 56.2 | 58.3 | 46.5 | 75 | 69.1 | 68.2 | 72.5 | 71.9 | 57 | 64.6 | 65.7

Table 3. Most productive players and their performance

function. We calculate Recall, Precision and F-measure using their standard definitions, directly on the links and on the chains as well. Measuring the agreement between two players, only F-measure is interesting because it is symmetric. We propose two ways of calculation:

1. We assume individual links as annotation units for both players. We mark the agreement on links F_{links}.
2. We take links of one player and compare them with coreferential chains of the second player (or the manual annotation). I.e., if one player links nodes A—C, and the other player links nodes A—B and B—C, there is an agreement on the link A—C (see Figure 6). Using the first measure, it would be disagreement. This measure, marked F_{chains}, is the more important one. The same method is used in the scoring function in the game itself, as described above.

We observe first the game data for the players separately. We list statistics for two most productive players $Player_1$, $Player_2$ who played a great majority of the game sessions (11 and 20 out of 46) and we are also interested in whether their performance was improving with the increasing number of sessions. In Table 3, we list F_{chains} for successive sessions starting with the first session played. We can see that $Player_1$'s agreement was getting worse within his session series while $Player_2$'s agreement was more or less well balanced. In general, these numbers give a true picture how player concentration changes over playing time. Mainly, text comprehension in PlayCoref requires a relatively high concentration.

The average value of F_{chains} for all competitors is 60% and the minimal and maximal values are 13.5% and 81.6%, respectively. For illustration, we list the corresponding values of F_{links} – 55%, 10%, 81.6%.

We analyze the agreement between the manual annotation and the union and intersection of players' annotations (on the part of the data where the parallel annotations are available). The obvious expectation is that Precision of the intersection will be higher than Precision of the union, and on the other hand, Recall of the intersection will be lower than Recall of the union.

In total, 11 double player games were played, which resulted in 455 different coreferential pairs linked in 11 different documents: 123 pairs were linked by two players (i.e. they are elements of the intersection) and 332 links were marked by at least one

player (i.e. they are elements of the union). It is interesting to note that 252 pairs were linked in the manual annotation of 11 documents.

The average values of P^{union}_{chains} and R^{union}_{chains} are 65% and 71%, respectively, and the values of $P^{intersection}_{chains}$ and $R^{intersection}_{chains}$ are 88% and 43%, respectively. That confirms the theoretical expectation.

To set an upper bound of the players' annotation agreement, we measured the annotation agreement between the two annotators who manually marked the coreference chains during the preparation of the game data. On 110 sentences annotated in paralell we got $F_{links} = 94\%$ and $F_{chains} = 95\%$. The average agreeement of players who played 11 two-player games is $F_{links} = 57\%$ and $F_{chains} = 65\%$.

3.2. Shannon game

Shannon game is a game for one or two players with hidden words in the sentence. The players guess the words with the help of unhidden words in the sentence.

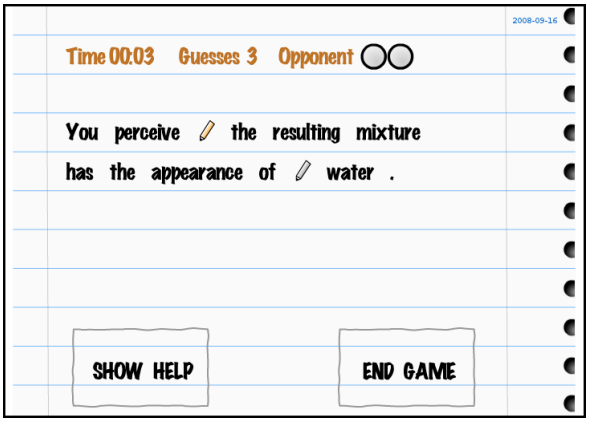


Figure 7. The Shannon game starts.

The game For each missing word, the player has three attempts (guesses). The player simply writes a word and pushes the `Enter` button. If the word is correct, it becomes green and the player moves to the next missing word. If it is not correct, the player looses one guess and can start writing another word as his next attempt. If he guesses incorrectly for three times, the correct word is displayed in red and the player moves to the next missing word. If all missing words have been (correctly or three times incorrectly) guessed, the game ends. The player can also end the game sooner by clicking on the button "End game".

At the beginning of the game, the player chooses one of three difficulty levels. The higher the difficulty, the higher the number of missing words in the sentence: either 2 or 3 or 4 hidden words.

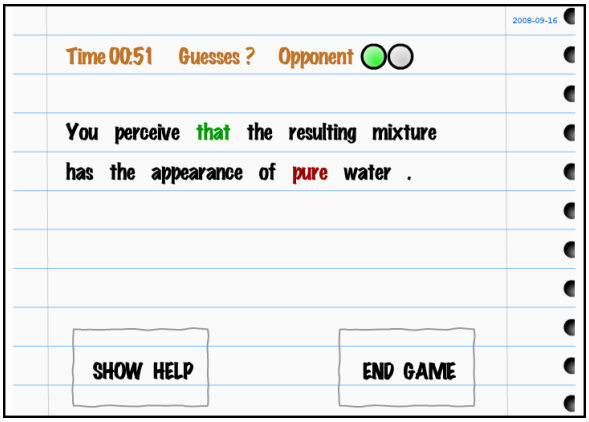


Figure 8. Shortly before the end of the Shannon game.

Game data preparation Any textual data can be used in the game. To pre-process the data, sentence segmentation and tokenization are needed. Only sentences of certain length and without the punctuation are selected. As proper names are almost impossible to guess without a broader context, they should not be used as missing words. Therefore, a procedure for proper names recognition is also needed.

Players' score We do not use any special formula to compute the players' score. Instead, a very straightforward point assignment is applied. For each word being guessed, the player gets points depending on the number of wrong guesses:

- 40 pts – if the 1st guess is correct
- 20 pts – if the 2nd guess is correct
- 10 pts – if the 3rd guess is correct
- -10 pts – if no guess is correct

For example, if two words are hidden, the total score of the player can range from -20 (no word guessed correctly) to 80 (both words guessed correctly at the first attempt) points.

Figure 9. The Shannon game – the players' score.

3.3. Place the Space

Place the Space is a single-player game of word segmentation. The player is presented with a sentence depicted without spaces between words. His task is to restore the spaces in a time-limit set up according to the length of the sentence.

The game To place the space, the player clicks on the character that should immediately follow the space. It can be later removed by clicking on the space.

Game data preparation Any textual data can be used in the game and only sentence segmentation is needed to process the data. To select sentences of a certain length (not too short and not too long), we also use tokenization and we select sentences according to the number of tokens. The number of characters would also be a sufficient measure.

Players' score The score ranges from 0 to 100 (both included) counted as the F-measure between the correct solution and the player's solution.

$$score(Player) = 100 * F(Player,\ Correct)$$

where F stands for the $F-measure = \frac{2*Precision*Recall}{Precision+Recall}$, and (in a standard way), Precision = (number of correctly guessed positions)/(number of guesses), and Recall = (number of correctly guessed positions)/(number of correct positions).

As spaces are naturally written in English and Czech texts, for these languages the game only serves to attract people to the game portal. However, it is a fast-paced and a

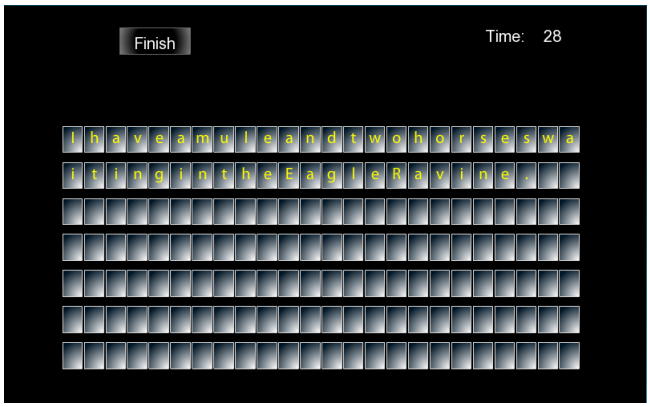


Figure 10. The Place the Space game starts.

very simple game that requires no training. For some languages like Thai or Chinese, where there are no spaces written between words, the game might produce some useful data. However, counting the score would require an automatic procedure for word segmentation; also a comparison to previous solutions by other players could be used.

4. Conclusion

We pay attention to crowdsourcing projects with the textual data, namely we concentrate on games with a purpose. These textual games present a minority of games simply because reading a text is not so enjoyable like for example observing pictures. Notwithstanding this fact, we have designed and implemented the PlayCoref game on marking coreference in the document. Even more, we have organized the very first PlayCoref competition where totally 46 sessions have been played. We are aware that such number of sessions is not large enough to make fundamental conclusions. On the other hand, the competition has encouraged our enthusiasm for PlayCoref game because the competition statistics make sense and the players enjoyed the game.

We implemented two more games, Shannon game and Place the Space. There is no specific natural language processing task to address through these games. We have designed them mainly to invite the Internet users to our language game portal.

To finish primary steps with the text games, a number of implementation actions will be done.

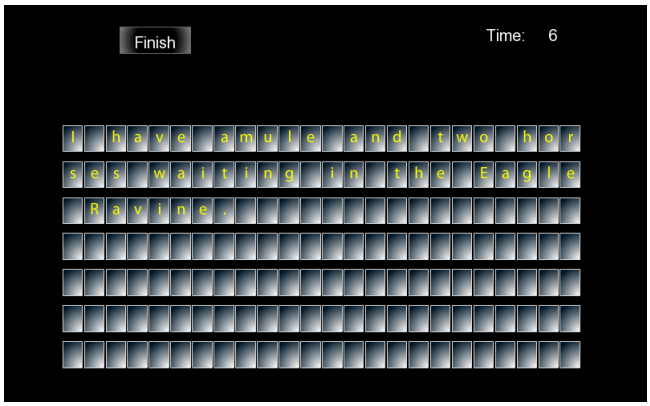


Figure 11. Place the Space: A player's solution with one error.

Figure 12. Place the Space: a player's score, along with the correct solution and an indication of the error.

Acknowledgements

We gratefully acknowledge the support of the Grant Agency of the Czech Republic (grants 405/09/0729 and P406/2010/0875) and the Czech Ministry of Education (grants MSM-0021620838 and LM2010013).

The authors would like to thank Eva Hajičová and Petr Sgall for their valuable comments that helped improve the paper. We really appreciate the patience of Martin Popel, the technical editor of PBML.

Bibliography

Chamberlain, Jon, Massimo Poesio, and Udo Kruschwitz. Phrase detectives: A web-based collaborative annotation game. In *Proceedings of I-SEMANTICS '08 Graz, Austria, September 3-5*, 2008.

Crowdsourcing.org, 2011. URL `http://crowdsouring.org`.

Doan, Anhai, Raghu Ramakrishnan, and Alon Y. Halevy. Crowdsourcing Systems on the World-Wide Web. *Communication of the ACM*, 54(4):86–96, 2011.

Doyle, Arthur Conan. *A Study in Scarlet*. 1887, 2005.

Green, Nathan, Paul Breimyer, Vinay Kumar, and Nagiza F Samatova. PackPlay: Mining semantic data in collaborative games. In *Proceedings of the Fourth Linguistic Annotation Workshop, Uppsala, Sweden*, pages 227–234. 2010.

Hladká, Barbora, Jiří Mírovský, and Pavel Schlesinger. Designing a Language Game for Collecting Coreference Annotation. In *Proceedings of the Third Linguistic Annotation Workshop (ACL-LAW III), Singapore*, pages 52–55, 2009a.

Hladká, Barbora, Jiří Mírovský, and Pavel Schlesinger. Play the Language: Play Coreference. In *Proceedings of ACL-IJCNLP 2009, Singapore*, pages 209–212, 2009b.

Hladká, Barbora Vidová, Jan Hajič, Jiří Hana, Jaroslava Hlaváčová, Jiří Mírovský, and Jan Raab. Czech academic corpus 2.0, 2008.

Ipeirotis, Panagiotis G. and Praveen K. Paritosh. Tutorial: Managing Crowdsourced Human Computation, 2011.

Kittur, Aniket, Ed H. Chi, and Bongwon Suh. Crowdsourcing User Studies with Mechanical Turk. In *Proceedings of the CHI 2008, Florence, Italy*, 2008.

Ma, Hao, Raman Chandrasekar, Chris Quirk, and Abhishek Gupta. Page Hunt: Improving Search Engines Using Human Computation Games. In *Proceedings of the SIGIR, Boston, MA, USA*, 2009.

Marcus, Mitchell P., Beatrice Santorini, and Mary Ann Marcinkiewicz. Building a Large Annotated Corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–330, 1993.

Miller, George A. Wordnet: A lexical database for english. *Communication of the ACM*, 38(11): 39–41, 1995.

Nedoluzhko, Anna, Jiří Mírovský, Radek Ocelák, and Jiří Pergler. Extended coreferential relations and bridging anaphora in the prague dependency treebank. In *Proceedings of the 7th Discourse Anaphora and Anaphor Resolution Colloquium (DAARC 2009), Goa, India*, pages 1–16. 2009.

Novák, Michal. Machine learning approach to anaphora resolution. Master's thesis, Charles University in Prague, Faculty of Mathematics and Physics, 2010.

Quinn, Alexander J. and Benjamin B. Bederson. A taxonomy of distributed human computation, 2009.

Seemakurty, Nitin, Jonathan Chu, Luis von Ahn, and Anthony Tomasic. Word Sense Disambiguation via Human Computation. In *Proceedings of the KDD-HCOMP, Washington, DC, USA*, 2010.

Sgall, Petr, Eva Hajičová, and Jarmila Panevová. *The Meaning of the Sentence in Its Semantic and Pragmatic Aspects*. D. Reidel Publishing Company, Dordrecht, 1986.

Siorpaes, Katharina and Elena Simperl. Incentives, Motivation, Participation, Games: Human Computation for Linked Data. In *CEUR Proceedings of the Workshop on Linked Data in the Future Internet at the Future Internet Assembly, Ghent, Belgium*, 2010.

Snow, Rion, Brendan O'Connor, Daniel Jurafsky, and Andrew Y. Ng. Cheap and Fast — But is it Good? Evaluating Non-Expert Annotations for Natural Language Tasks. In *Proceedings of EMNLP 2008, Honolulu*, page 254–263. 2008.

Stoyanov, V., C. Cardie, N. Gilbert, E. Riloff, D. Buttler, and D. Hysom. Coreference resolution with reconcile. In *Proceedings of ACL 2010, Short Paper*, 2010.

Versley, Yannick, Simone Ponzetto, Massimo Poesio, Vladimir Eidelman, Alan Jern, Jason Smith, Xiaofeng Yang, and Alessandro Moschitti. Bart: A modular toolkit for coreference resolution. In *Proceedings of LREC'08*, Marrakech, Morocco, May 2008. ELRA. ISBN 2-9517408-4-0. http://www.lrec-conf.org/proceedings/lrec2008/.

von Ahn, Luis, Shiry Ginosar, Mihir Kedia, and Manuel Blum. Verbosity: A Game for Collecting Common-Sense Knowledge. In *Proceedings of the SIGHI Conference on Human Factors in Computing Systems, ACM Press*, pages 75–78. 2007.

von Ahn, Luis, Benjamin Maurer, Colin McMillen, David Abraham, and Manuel Blum. reCAPTCHA: Human-based character recognition via web security measures. *Science*, 321 (5895):1465–1468, 2008.

Wang, Aobo, Cong Duy Vu Hoang, and Min-Yen Kan. Perspectives on Crowdsourcing Annotations for Natural Language Processing, 2010. URL `http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.172.559`.

Yuen, Man-Ching, Ling-Jyh Chen, and Irwing King. A survey of human computation systems. In *Proceedings of the Computational Science and Engineering, IEEE International Conference*, page 723–728. 2009. URL `http://doi.ieeecomputersociety.org/10.1109/CSE.2009.395`.

Žabokrtský, Zdeněk, Jan Ptáček, and Petr Pajas. TectoMT: Highly modular MT system with tectogrammatics used as transfer layer. In *ACL 2008 WMT*, pages 167–170, Columbus, OH, USA, 2008. ACL. ISBN 978-1-932432-09-1.

Zeman, Dan, Jiří Hana, Hana Hanová, Jan Hajič, Barbora Hladká, and Emil Jeřábek. A manual for morphological annotation, 2nd edition. Technical Report 27, ÚFAL MFF UK, Prague, Czech Republic, 2005.

Address for correspondence:
Barbora Hladká
`hladka@ufal.mff.cuni.cz`
UK MFF, ÚFAL
Malostranské nám. 25, 118 00 Prague 1, Czech Republic

PBML

The Prague Bulletin of Mathematical Linguistics
NUMBER 96 OCTOBER 2011 27–47

Specificity of the number of nouns in Czech and its annotation in Prague Dependency Treebank

Magda Ševčíková, Jarmila Panevová, Lenka Smejkalová

Charles University in Prague, Faculty of Mathematics and Physics, Institute of Formal and Applied Linguistics

Abstract

The paper focuses on the way how the grammatical category of number of nouns will be annotated in the forthcoming version of Prague Dependency Treebank (PDT 3.0), concentrating on the peculiarities beyond the regular opposition of singular and plural. A new semantic feature closely related to the category of number (so-called pair/group meaning) was introduced. Nouns such as *ruce* 'hands' or *klíče* 'keys' refer with their plural forms to a pair or to a typical group even more often than to a larger amount of single entities. Since pairs or groups can be referred to with most Czech concrete nouns, the pair/group meaning is considered as a grammaticalized meaning of nouns in Czech. In the present paper, manual annotation of the pair/group meaning is described, which was carried out on the data of Prague Dependency Treebank. A comparison with a sample annotation of data from Prague Dependency Treebank of Spoken Czech has demonstrated that the pair/group meaning is both more frequent and more easily distinguishable in the spoken than in the written data.

1. Introduction

In Czech, nouns typically have two sets of forms according to the grammatical category of number: singular forms and plural forms. Forms of the former set are used to denote a single entity (singularity meaning, sg), plural forms express, in general, more than one entity (plurality meaning, pl).

In addition to the existence of nouns accompanied in the lexicon with the feature "singulare tantum", which blocks the semantic opposition of sg vs. pl, and "plurale tantum", where the opposition of sg and pl is expressed by the same form, we introduce a new semantic feature closely related to the category of number, namely

© 2011 PBML. All rights reserved. Corresponding author: sevcikova@ufal.mff.cuni.cz
Cite as: Magda Ševčíková, Jarmila Panevová, Lenka Smejkalová. Specificity of the number of nouns in Czech and its annotation in Prague Dependency Treebank. The Prague Bulletin of Mathematical Linguistics No. 96, 2011, pp. 27–47. doi: 10.2478/v10108-011-0009-y.

the "pair/group meaning" (Section 2 of the paper). Nouns such as *ruce* 'arms', *boty* 'shoes' or *klíče* 'keys' refer with their plural forms rather to a pair or to a typical group even more often than to a larger amount of single entities; thus the plural *ruce* denotes a pair or several pairs of arms rather than several upper limbs, the form *boty* 'shoes' usually denotes a pair or several pairs of shoes, the form *klíče* 'keys' often means a bundle or more bundles of keys. Since pairs or groups can be referred to with most Czech concrete nouns and since this phenomenon is reflected in some peculiarities as to the compatibility of the particular nouns with numerals, the pair/group meaning is considered as a grammaticalized meaning of nouns in Czech.

In Section 3 of the paper, the manual annotation of the pair/group meaning is described, which was carried out on the data of Prague Dependency Treebank version 2.0 (PDT 2.0)[1] by two human annotators in parallel. The annotation was evaluated in several aspects (inter-annotator agreement, frequency of the pair/group meaning with particular nouns etc.).

Results of the annotation of the written data from PDT 2.0 are compared with a sample annotation of the data from Prague Dependency Treebank of Spoken Czech. The fact that the pair/group meaning is both more frequent and more easily distinguishable in the spoken than in the written data is briefly discussed in Section 4. The annotation of the pair/group meaning is to be included in the forthcoming version of Prague Dependency Treebank (PDT 3.0), which is designed as a both revised and extended version of PDT 2.0 (Sect. 5).

2. The pair/group meaning of Czech nouns

2.1. Nouns expressing pairs or groups

The starting point of the considerations on the pair/group meaning was an analysis of Czech nouns the plural of which usually refers to pairs or groups of entities, not to a plurality of single entities, though they are countable as single entities and also the regular opposition of sg and pl is applicable here (*jeden klíč* 'one key', *dva klíče* 'two keys' etc.). This phenomenon concerns especially nouns denoting body parts occurring in pairs or groups (*uši* 'eyes', *prsty* 'fingers', *vlasy* 'hair'), further clothes and accessories for these body parts (*náušnice* 'earrings', *rukavice* 'gloves'), family members such as *rodiče* 'parents', *sourozenci* 'siblings', and objects of everyday use and foods sold or used in typical amounts (*klíče* 'keys', *sirky* 'matches', *cigarety* 'cigarettes', *sušenky* 'biscuits').

Plural forms of other Czech concrete nouns may refer to pairs or groups of entities as well but, according to a detailed corpus analysis, they are mostly accompanied with a so-called set numeral in such contexts. Set numerals are considered to be a

[1] See (Hajič et al., 2006), http://ufal.mff.cuni.cz/pdt2.0

special sub-type of numerals in Czech (besides the cardinal, ordinal etc. numerals),[2] classification of numerals as set numerals is based on their formal shape, not on their meaning; set numerals are compatible with nouns in plural only.[3] The primary meaning of the set numerals is to express different sorts of the entities denoted by the noun (ex. (1)). However, the same set numerals, if combined with pluralia tantum nouns, express either the amount of single entities (i.e. the same meaning which is expressed by cardinal numerals with most nouns), or the number of sorts, cf. ex. (2). The set numerals in combination with the nouns which we are interested in in the present paper express the number of pairs or groups; this means that the set numerals are used here instead of cardinal numerals while the cardinals combined with these nouns express the number of single entities (cf. *dvoje boty* 'two pairs of shoes', *troje boty* 'three pairs of shoes' vs. *dvě boty* 'two shoes', *tři boty* 'three shoes').[4]

(1) *Máme **dvoje** sklenice – na bílé a červené víno.*[5]
'We have **two sets** of glasses – for the white and for the red wine.'

(2) *Na stole leží **troje** nůžky.*
'There are **three types//pieces** of scissors on the table.'

Due to the ambiguity of the set numerals as well as to the fact that pairs or groups are referred to mostly by nouns that are not accompanied with a set numeral, the pair/group meaning is not attributed to the numerals, it is proposed to be considered as a meaning of nouns. If a noun denoting a pair or group collocates with a numeral, the meaning of the noun is only reflected in the surface form of the numeral, i.e. the set numeral is used.

2.2. The pair/group meaning as a grammatical meaning

As we aimed at including the pair/group meaning into the theoretical description of the Czech language, namely into the framework of Functional Generative Descrip-

[2]Set numerals are available, for instance, in Russian, Serbian and Croatian as well; however, there are several differences in counting pairs and groups between these languages and Czech. In English and German, on the other hand, there are no numerals of this type, the number of pairs and groups is expressed by cardinal numerals in combination with the nouns such as *pair, bundle* and *Paar, Bündel*, respectively (cf. the English translations of the examples given in the text); see (Panevová and Ševčíková, 2011).

[3]Well established terms for formal and semantic aspects of numerals, suitable also for covering such irregularities, are missing in Czech linguistics.

[4]Within the tectogrammatical annotation of PDT 2.0 the numerals of both types *tři* and *troje* are represented by a single "deep" lexical item and the particular (cardinal, set etc.) meaning is represented as a separate semantic feature (grammateme numertype) according to the meaning of the counted noun and to the current context; see Fig. 2 in the paper, further details can be found in (Razímová and Žabokrtský, 2006).

[5]Examples (1) to (4), (16) and (17) were created as illustrative examples by the authors, all the remaining examples come from the PDT 2.0 data.

tion (FGD; (Sgall, 1967), (Sgall et al., 1986)) and possibly also into the annotation of PDT, the annotation scenario of which is based on FGD, the possibility to consider the pair/group meaning as a grammaticalized meaning of most Czech nouns was preferred to the possibility to treat it as a semantic feature of some of them (as a component of their lexical meaning). The latter possibility would have implied to split lexicon entries (at least) of the prototypical pair/group nouns into two entries, an entry with a common singular-plural opposition and an entry for cases in which the plural of the noun refers to pairs or groups (and behave, in fact, as pluralia tantum); the potential compatibility of the pair/group meaning with other nouns, though, would have remained unsolved. The broad coverage and the economy of the lexicon seem to be the main advantages that can be achieved when preferring the former solution in this case.

2.3. The pair/group meaning and the category of number

The pair/group meaning is closely connected with the grammatical number of nouns, though, we do not subsume it under this category; it is considered as a distinct one. The main reasons for this decision are, firstly, that the pair/group meaning is compatible both with singularity and plurality so that it cannot be considered as a third meaning of the category of number, and secondly, that the pair/group meaning is not subordinate to the meanings of the category of number so that it does not seem to be appropriate to consider it as a sub-value of singularity and plurality.

We thus worked with two oppositions in the theoretical description: the first opposition is the basic opposition of the category of number (i.e., sg vs. pl), the second one is constituted by the pair/group meaning (group) as opposed to the meaning of single entities (single). The combination sg-single (i.e. "one entity") is expressed by singular forms of nouns, the other three combinations (sg-group "one group", pl-single "more than one entity" and pl-group "more than one pair/group")[6] by plural forms; see the annotation choices in Sect. 3 and process of matching the annotation choices to the grammateme values in Sect. 5.

The ambiguous plural form is disambiguated either by the numeral, which, however, co-occurs rather rarely in the data, or on the basis of context or knowledge of the world (thus in ex. (3) the combination sg-group is preferred whereas the same noun form in ex. (4) is interpreted as pl-group). This fact can be hardly used for an automatic identification of the particular meanings, thus we decided for a manual annotation of the pair/group meaning.

(3) *Na rukou měl kožené **rukavice**.*
'He had leather **gloves** on his hands.'

[6]The combinations sg-group and pl-group are together referred to as the pair/group meaning in this paper.

(4) *V obchodě nabízejí nejrůznější **rukavice**.*
'In the shop different **gloves** are sold.'

3. Manual annotation of the pair/group meaning in the PDT 2.0 data

3.1. Selection of the data to be annotated

The aim of the annotation was to check whether the proposed pair/group meaning is distinctive enough in different contexts and how frequent it is in authentic language data. PDT 2.0 is a collection of Czech newspaper texts from 1990's, to which morphological tagging and annotation at two syntactic layers was added: at the so-called analytical layer (layer reflecting the surface syntactic structure) and at the tectogrammatical layer (layer of the linguistic meaning of the sentence); at both of them the sentence is represented as a dependency tree with labeled nodes and edges.

As the pair/group meaning is expressed by formally unmarked plural forms, all plural forms of nouns are candidates for the manual disambiguation of the meanings studied here; i.e. 60 thousand plural noun forms out of all 833 thousand tokens for which tectogrammatical annotation is available. Nevertheless, since a rather low frequency of the pair/group meaning was expected on the background of a pilot annotation experiment,[7] only plural forms of those nouns were manually annotated for which the pair/group meaning is considered as prototypical, in order to make the annotation as efficient as possible. In the (open) list of prototypical pair/group nouns to be annotated, nouns were involved which co-occur with a set numeral in the PDT 2.0 and in the SYN2005 data, the list was further enriched using grammar books and theoretical studies on number in Czech[8] as well as linguistic introspection. The resulting list consists of 141 Czech nouns:[9]

adidaska 'adidas shoe', *bačkora* 'slipper', *bačkorka/bačkůrka* 'slipper.*DIMIN*', *běžka* 'cross-country ski', *bok* 'hip', *bonbón* 'bonbon', *bota* 'shoe', *botaska* 'botas shoe', *botička* 'shoe.*DIMIN*', *botka* 'shoe.*DIMIN*', *brambor/brambora*

[7] In the pilot annotation 1,000 plural forms randomly selected from the SYN2005 corpus were involved, the pair/group meaning was preliminarily assigned with roughly 5 % of them. However, during the manual annotation of the PDT data, which is described in this paper, it turned out that the pair/group meaning is even much less frequent in the PDT data than in the pilot annotation. This fact might be connected with differences in the composition of the corpora (SYN2005 is a representative corpus of Czech, in PDT only newspapers are involved; see Sect. 4).

[8] Cf. esp. (Komárek et al., 1986), (Miko, 1962), and (Straková, 1960).

[9] The English translations capture the meaning of the listed Czech nouns; the formal characteristics of the English nouns thus do not correspond to those of the Czech ones in some cases (cf. the noun *těstovina*, which has both singular and plural forms in Czech, and its equivalent *pasta*). Some of the listed nouns are abbreviated from product names; well-known product names are included in the translation (cf. the noun *adidaska* 'adidas shoe'), if a specifically Czech product name is the source of the noun, its meaning is described without including the product name (*miňonka* 'chocolate biscuit'). Diminutives are formed with special suffixes in Czech, they are marked with ".*DIMIN*" in the translations. If there exist formally different variants with the same meaning, they are introduced using the slash (*brambor/brambora* 'potato').

'potato', *brusle* 'skate', *chlup* 'hair', *chodidlo* 'sole', *cigareta* 'cigarette', *čtyřče* 'quadruplet', *cvička* 'gym shoe', *datle* 'date', *dlaň* 'palm', *doklad* 'document', *dřeváček* 'clog.*DIMIN*', *dřevák* 'clog', *dvojče* 'twin', *fík* 'fig', *iniciála* 'initial', *kanada* 'working boot', *kapička* 'drop.*DIMIN*', *kapka* 'drop', *keks* 'cracker', *kel* 'tusk', *klíč* 'key', *klíček* 'key.*DIMIN*', *kolej* 'rail', *koleno* 'knee', *kolínko* 'knee.*DIMIN*', *končetina* 'limb', *kopačka* 'football boot', *kotník* 'ankle', *kozačka* 'boot', *křídlo* 'wing', *kroupa* 'barley', *kšanda* 'brace', *kulisa* 'scene', *kyčel* 'coxa', *lakýrka* 'patent shoe', *ledvina* 'kidney', *lék* 'medicine', *lentilka* 'chocolate candy', *lodička* 'pump', *loket* 'elbow', *lýtko* 'calf', *lyže* 'ski', *makaron* 'macaroni', *mandle* 'tonsil; almond', *mentolka* 'peppermint drop', *miňonka* 'chocolate biscuit', *mokasína* 'step-in shoe', *ňadro* 'breast', *náušnice* 'earring', *nehet* 'nail', *noha* 'foot, leg', *nozdra* 'nostril', *nožička* 'foot.*DIMIN*, leg.*DIMIN*', *nudle* 'noodle', *obočí* 'eyebrow', *očko* 'eye.*DIMIN*', *oko* 'eye', *oplatek/oplatka* 'wafer', *ořech* 'nut', *oříšek* 'nut.*DIMIN*', *osmerče* 'octuplet', *pantofle* 'slipper', *papuče* 'slipper', *parket/parketa* 'parquet', *paroh* 'horn', *partyzánka* 'cigarette', *pata* 'heel', *paterče* 'quintuplet', *piškot* 'sponge biscuit', *pistácie* 'pistachio', *plátěnka* 'canvas shoe', *plíce* 'lung', *podešev* 'sole', *podkolenka* 'knee sock', *ponožka* 'sock', *pouto* 'tie', *prarodič* 'grandparent', *prášek* 'pill', *prso* 'breast', *prst* 'finger', *punčocha* 'hose', *punčoška* 'hose.*DIMIN*', *rameno* 'shoulder', *řasa* 'eyelash', *ret* 'lip', *rodič* 'parent', *roh* 'horn', *rolnička* 'bell', *rozinka* 'raisin', *rtík* 'lip.*DIMIN*', *ručička* 'hand.*DIMIN*, arm.*DIMIN*', *ruka* 'hand, arm', *rukavice* 'glove', *sandál* 'sandal', *sardinka* 'sardine', *schod* 'stair', *schůdek* 'stair.*DIMIN*', *sedmerče* 'septuplet', *sirka* 'match', *sluchátko* 'earphone', *sourozenec* 'sibling', *sparta* 'cigarette', *stehno* 'thigh', *střevíc* 'shoe', *střevíček* 'shoe.*DIMIN*', *sušenka* 'biscuit', *šesterče* 'sextuplet', *škvarek/škvarka* 'crackling', *šle* 'brace', *špageta* 'spaghetti', *teniska* 'gym shoe', *těstovina* 'pasta', *trojče* 'triplet', *tyčinka* 'bar', *ubrousek* 'napkin', *ucho* 'ear', *vlas* 'hair', *vločka* 'flake', *vráska* 'wrinkle', *zápalka* 'match', *zápěstí* 'wrist', *závora* 'barrier', *závorka* 'bracket', *žiletka* 'blade', *zoubek* 'tooth.*DIMIN*', *zub* 'tooth'

Only 67 out of the listed nouns were found in the PDT 2.0 data; for these noun lemmas there are 618 instances of plural forms in the data. More than a half of the 618 selected plural forms belong to five noun lemmas only (*oko* 'eye' 89, *rodič* 'parent' 87, *ruka* 'hand, arm' 81, *doklad* 'document' 35, *bota* 'shoe' 30; see the "coverage" in Table 4), 40 out of the 67 nouns had less than five instances of plural forms in the data. The plural forms to be annotated were extracted from the data together with a short context (both preceding and following) and divided into 31 html files. The annotators worked thus with a simple, linear text with highlighted plural forms followed by a drop-down list with five annotation choices, from which one should be chosen (see Fig. 1):[10]

1 - plurality,
2 - one pair/group,
3 - several pairs/groups,
4 - one pair/group or several pairs/groups,
5 - cannot be resolved.

[10]The default string "- - -" labeled as the sixth choice in the Tables was used by the annotators to indicate a mistake (for instance, if a singular form was involved in the annotation because of a mistake in the morphological tagging).

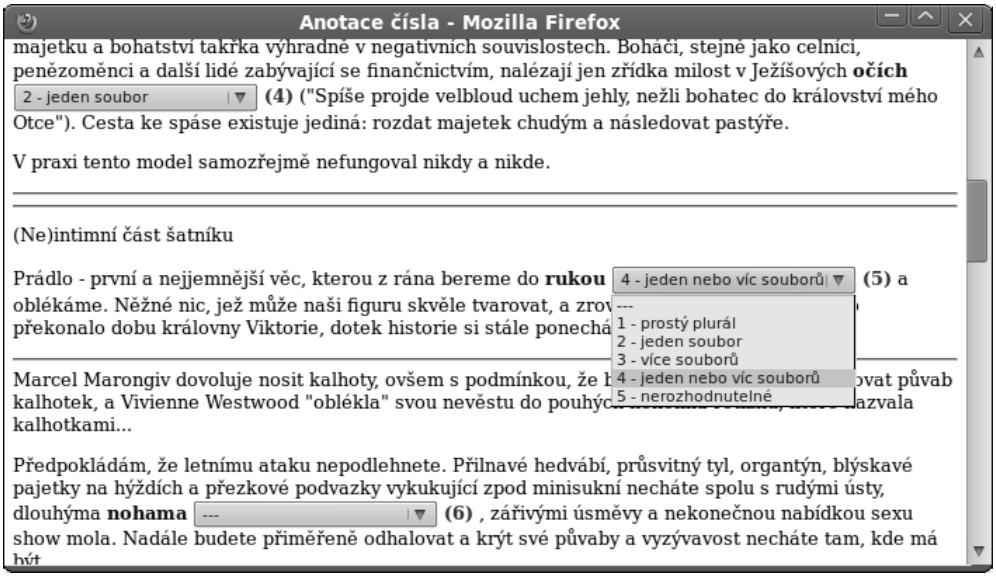


Figure 1. Screenshot of the html file to be annotated: linear text with highlighted instances followed by a drop-down list of annotation choices

3.2. Assigning the choices

All 31 files were annotated by two human annotators in parallel from October 2010 to January 2011, the annotation was preceded by a short training period. Both annotators are native Czech speakers; the language intuition of native speakers played a crucial role in the annotation process, several annotation "rules" formulated for problematic contexts are introduced in Section 3.3.

The first annotation choice, **1 - plurality**, was assigned to nouns denoting several single entities. The fact that single entities were referred to by the speaker was either obvious from the context (e.g. quantifier *několik* 'several' in ex. (5)) or could be inferred from the knowledge of the situation (cf. ex. (6)).

(5) *Středem pozornosti kamer je nakrátko ostříhaná, křehká Dolores, která v červené čapce a s několika* ***náušnicemi*** *na pódiu vypadá jako kolébající se pirátka.*
'The close-cropped, flimsy Dolores is the center of the attention of the cameras, who in a red cap and with several **earrings** looks like a waddling pirate on the stage.'

(6) *Sečíst pouhým okem stranickou příslušnost zvednutých* ***rukou*** *bylo ve dvousetčlenné Poslanecké sněmovně nemožné.*

'It was impossible to count up with the naked eye the party affiliation of the risen **hands** in the two-hundred member Chamber of Deputies.'

The second choice (**2 - one pair/group**) was the most frequently occurring choice in the annotation. It was assigned in basic contexts such as in ex. (7) (one human has one typical group of hairs on his head), but also in sentences like (8) in which the noun is used figuratively.

(7) *Bydlí v Přadlácké ulici a má silnější, 178 cm vysokou postavu, hnědé krátké **vlasy** ...*
'He lives in Přadlácká Street and is corpulent, 178 cm tall, has brown **hair** ...'

(8) *Lidé od zaniklých pojišťoven se vrátí pod **křídla** VZP.*
'People come back from the defunct insurance companies under the **wings** of the General Health Insurance Company.'

Unlike the previous choice, the annotators decided for the choice **3 - several pairs/groups** with less than 5 % of the annotated instances, mainly if the noun was accompanied by another noun in a close context which expresses the opposition of sg and pl regularly and was used in plural in the particular text. For instance, the noun *ruce* 'hands' in ex. (9) was assigned the choice 3 (which is in fact "plurality of pairs/groups") according to the plural form (plurality meaning) of the noun *hlavy* 'heads'.

(9) *Šikovné **ruce** a hlavy rovněž nejsou tak vzácné. Thajské dívky dnes vyrábějí elektroniku světové úrovně.*
'Even skillful **hands** and heads are not that rare. Today, Thai girls produce electronics of world-renowned quality.'

The choice **4 - one pair/group or several pairs/groups** has been proposed for cases in which the annotator preferred the pair/group meaning to the plurality meaning but was not certain which of the choices 2 or 3 is the right one. The annotators' uncertainty originated, on the one hand, from a lack of knowledge about the particular situation (ex. (10)) or, on the other, it was connected with the problem of expressing amounts in distributive contexts on the other. For instance, the plural *očích* 'eyes' in ex. (11) can be interpreted both as several pairs because eyes of several people are denoted, and as one pair/group since each of the people should have glasses on his pair of eyes (cf. the noun *na očích* could be substituted by singular as well as plural form *na nose/nosech* 'on their nose/noses' in the particular Czech sentence; for the distributivity issue see Sect. 3.3). The choice one pair/group or several pairs/groups was the second most frequent choice in the annotation, nearly 25 % of the instances were assigned this value.

The annotation choice **5 - cannot be resolved** was used if there were neither linguistic features (context) nor extra-linguistic evidence that make the decision between the plurality meaning and the pair/group meaning possible (ex. (12)).

(10) *Pro něho připravila firma Lotto speciální **kopačky**.*
'The Lotto company developed special **football boots** for him.'

(11) *... aby lidé při sváření měli na **očích** ochranné brýle.*
'... so that people have protective glasses on their **eyes** during the welding.'

(12) *... ... je to také odpověď na vzdělávací požadavky **rodičů**, žáků, ale i měnícího se trhu práce.*
'... it is an aswer to educational requirements of the **parents**, pupils, but of the changing job market as well.'

3.3. Annotation of questionable cases: figurative usage, collectives, distributivity

Already during the short pre-annotation training, we came across many figurative contexts as well as phrasemes, titles etc. in which none of the proposed choices was intuitively preferred by the annotators. In order to achieve good annotation results even in these cases, we agreed on a rather general principle: the nouns should be interpreted in a possibly simple way. Thus for instance, the noun in ex. (8) mentioned above was treated as if we deal with a literal, non-figurative context (one pair/group), the nouns in the phrasems in ex. (13) and (14) were assigned the same choice. A suggestion to exclude phraseological and figurative contexts from the annotation does not seem to be feasible in practice since in many cases the boundary between the literal and another kind of usage cannot be reasonably delimited.

(13) *rozhovor z **očí** do **očí***
lit.: a talk from **eyes** to **eyes**
'a face-to-face talk'

(14) *hnutí Na vlastních **nohou***
lit.: movement On own **feet**
'movement On one's own two feet'

Another type of contexts discussed before the annotation started were sentences in which the noun to be annotated relates to a noun with a collective meaning, cf. the noun *oči* 'eyes' relates to the collective noun *posádka* 'crew', *uši* 'ears' to the noun *publikum* 'audience' etc. in ex. (15). According to the "rule" accepted by the annotators, such contexts were treated as if the whole group of persons referred to by the collective noun had just one pair of eyes, ears etc., thus the nouns marked in bold in ex. (15) were each assigned the second annotation choice (one pair/group). This rule might seem to be in conflict with an intuitive interpretation, since one can easily imagine the (exact) number of persons referred to by the noun *posádka* 'crew' in the particular context; however, taking into account that the nouns *publikum* 'audience' or *vláda* 'government' could be understood either as several individuals or as a body (this reading comes close to the ex. (8)) in the same context, the above rule proved to be useful for the annotators to keep consistency.

(15) *před **očima** posádky, **uším** publika, v **rukou** vlády, do **rukou** státu*
'in front of the **eyes** of the crew, to the **ears** of the audience, in the **hands** of the government, into the **hands** of the state'

Distributivity, which is an issue extensively studied by formal semanticists and linguists, is addressed here just as affecting the annotation of the pair/group meaning (cf. (Dotlačil, 2010) who deals, among other languages, also with Czech). There is often a relation between a noun which was involved in the annotation and another noun of the sentence which refers to an amount of entities. If the entities denoted by the annotated noun relate to each of the entities referred to by the other noun, it is a case of distributivity; cf. ex. (11) where the noun *očích* 'eyes' denotes the distributed entities and the noun *lidé* 'people' the targets of the distribution.[11]

As for the nouns with a regular opposition of singular and plural in Czech, the distributed entities are expressed either by singular or plural when distributing one entity to each of the targets (ex. (16), (17); at the tectogrammatical layer the nouns were assigned sg or pl according to their form, without taking into account their interchangeability).[12] Nouns we deal with are used always in plural if denoting distributed entities – when selecting one of the proposed choices, the question arose whether the plural should be interpreted as denoting one pair/group or several pairs/groups; the substitution test obviously does not help in such cases. The annotators decided with regard to the close context (ex. (18) and (19)); the noun *oči* 'eyes' in ex. (18) was assigned the choice several pairs/groups due to the plural form of the noun *nosy* 'noses', in ex. (19) the nouns *vrásky* 'wrinkles' and *oči* 'eyes' were both assigned one pair/group in accordance with the singular of the nouns *hlas* 'voice', *úsměv* 'smile' and *mluva* 'speech' in the particular sentence. In case there was no formal "clue" in the context, the choice one pair/group or several pairs/groups was assigned (ex. (20)). However, examples as (21), in which the choice was used twice (the noun *křídla* 'wings' was assigned the choice due to the lack of knowledge how many pairs of wings are concerned whereas the noun *nohy* 'feet' got this assignment due to the distributivity), has led us to the decision to distinguish the distributivity as a separate choice for the next annotation phase (see Sect. 4.2).

(16) *Studenti kroutili **hlavou/hlavami**.*[13]
'Students shook their **head/heads**.'

(17) *Studenti měli na **hlavě/hlavách čapku/čapky**.*
'Students had **hat/hats** on their **head/heads**.'

[11] The distributivity is, of course, not limited to the nouns expressing pairs/groups.

[12] Categories of nouns in distributive contexts can vary from language to language; cf. (Lashevskaja, 1999) for Russian and (Corbett, 2000) for other languages.

[13] In the examples (16) to (21) the nouns denoting the target of the distribution are underlined, the nouns with the pair/group meaning are marked in bold (as in the whole article), they express the distributed entities.

Choices by Annotator 1	Choices by Annotator 2						Total
	1	2	3	4	5	6	
1 - plurality	**115**	6	18	7	8	0	154
2 - one pair/group	5	**180**	1	16	5	0	207
3 - several pairs/groups	1	1	**22**	7	9	0	40
4 - one pair/group or several pairs/groups	2	27	14	**112**	15	0	170
5 - cannot be resolved	1	3	3	3	**35**	0	45
6 - - - -	0	1	0	1	0	**0**	2
Total	124	218	58	146	72	0	618

Table 1. Inter-annotator agreement in the annotation carried out on the PDT 2.0 data. The number of instances assigned each annotation choice by the first annotator are given in rows, the total number for each choice (the last column) is divided according to the choices by the second annotator, which are displayed in columns following the same principle (cf., 154 instances in total were assigned the choice 1 by the first annotator, in 115 out of them the second annotator assigned the same choice, in 6 of them the second annotator assigned the choice 2 etc.). Numbers of instances assigned the same choice by both annotators are marked in bold on the diagonal.

(18) *Rozkvetlým městem chodí kýchající lidé s červenými nosy, oteklýma **očima**, plnýma slz, a z kapes jim vypadávají papírové kapesníky.*
'Through the flowering town, sneezing people with red noses are walking, with swollen **eyes**, full of tears, and paper tissues are falling out of their pockets.'

(19) *Všude na světě to pánové dělají hlouběji posazeným hlasem, neprůstřelným úsměvem, pomalejší mluvou s dobře oddělovanými slovy, milým vějířkem **vrásek** kol **očí**.*
'All around the world men do it with help of a deeper-set voice, a bullet-proof smile, a slower speech with well-separated words, a nice fan of **wrinkles** around the **eyes**.'

(20) *Divil jsem se, že mu to Američané povolili, když všechny zprávy procházely jejich **rukama**.*
'I was surprised that Americans allowed it to him, when all messages went through their **hands**.'

(21) *... aby v dunění **křídel** dobyla vítězství Těch, kdo alespoň pomocí **nohou** uprchli tíze neplodného těla.*
'... so that in the rumble of **wings** she gains the victory of Those who, at least with help of the **feet**, escape the heaviness of the sterile body.'

3.4. Agreement analysis

The annotators agreed on 464 (75.1 %) out of 618 instances annotated, with a Kappa score of 0.67.[14] Another 64 instances were assigned either the choice 2 or 3 by the first annotator and the choice 4 by the second annotator, or vice versa. Thus, if having a less granular scale of annotation choices, an even higher agreement score might be expected. An overview of choices assigned by each of the annotators and the number of instances both annotators (dis)agreed on is given in Table 1.

After the parallel annotation had been finished, instances of disagreement were decided by a third annotator and the instances on which annotators agreed were revised in order to check the correctness and consistency of the annotation; the revised annotation is referred to as final annotation in the sequel.

With 69 of the 154 differently annotated instances, the choice of the first annotator was preferred, the choice of the second annotator was acknowledged to be the right one with 61 of the instances, the remaining 24 instances of disagreement were assigned a choice different from that of the first as well as the second annotator. Concerning the 464 instances which annotators agreed on, only three of them were changed by the third annotator during the revision.

In the final annotation, the annotation choice 2 was the most frequent one, see Table 2. As the choices 2, 3 and 4 are, in fact, particular meanings of the pair/group meaning, we can state that 414 plural forms were assigned the pair/group meaning in the presented annotation, i.e. in 67.0 % of the annotated instances (cf. the sum of choices 2, 3 and 4 in Table 2).

Further, we were interested in how frequent the pair/group meaning was with single noun lemmas which were involved in the annotation. For nouns *dvojče* 'twin', *pouto* 'tie', *ledvina* 'kidney', *vlas* 'hair', *kopačka* 'football boot', *ucho* 'ear', *lyže* 'ski', and *schod* 'stair', even all their instances were assigned one of the choices 2, 3 or 4. In Table 3 the percentage of the instances assigned the pair/group meaning among all instances of nouns with five or more plural occurrences in the PDT 2.0 data is specified. Table 4 gives a detailed overview of all annotation choices for each noun with five or more plural instances in the PDT 2.0 data, the numbers correspond to the final annotation; the inter-annotator agreement for each of these nouns is shown as well.

4. Pair/group meaning in the written vs. spoken data

4.1. Low frequency in the PDT 2.0 data

It is apparent from the analysis of the manual annotation that the pair/group meaning has a very low frequency in the PDT 2.0 data. We faced thus the question

[14] The Cohen's Kappa measure is used, which takes into account the effect of agreement by chance (Cohen, 1960).

Annotation choice	# of instances assigned	Percentage
1 - plurality	133	21.5 %
2 - one pair/group	230	37.2 %
3 - several pairs/groups	30	4.9 %
4 - one pair/group or several pairs/groups	154	24.9 %
5 - cannot be resolved	70	11.3 %
6 - - - -	1	0.2 %
Total	618	100.0 %

Table 2. Annotation choices in the final annotation of the PDT 2.0 data

Noun lemma	# of plural forms	# of pl. forms with the pair/group meaning	Percentage	Noun lemma	# of plural forms	# of pl. forms with the pair/group meaning	Percentage
dvojče 'twin'	5	5	100.0 %	*noha* 'foot, leg'	20	17	85.0 %
pouto 'tie'	5	5	100.0 %	*kulisa* 'scene'	6	5	83.3 %
ledvina 'kidney'	7	7	100.0 %	*koleno* 'knee'	5	4	80.0 %
vlas 'hair'	11	11	100.0 %	*bota* 'shoe'	30	24	80.0 %
kopačka 'football shoe'	5	5	100.0 %	*klíč* 'key'	8	5	62.5 %
ucho 'ear'	9	9	100.0 %	*zub* 'tooth'	14	8	57.1 %
lyže 'ski'	13	13	100.0 %	*rodič* 'parent'	87	37	42.5 %
schod 'stair'	6	6	100.0 %	*křídlo* 'wing'	17	5	29.4 %
ruka 'hand, arm'	81	77	95.1 %	*doklad* 'document'	35	8	22.9 %
prst 'finger/toe'	10	9	90.0 %	*cigareta* 'cigarette'	17	3	17.6 %
oko 'eye'	89	80	89.9 %	*lék* 'medicine'	16	2	12.5 %
rameno 'shoulder'	9	8	88.9 %	*brambor* 'potato'	9	1	11.1 %
rukavice 'glove'	8	7	87.5 %	*těstovina* 'pasta'	7	0	0.0 %
kolej 'rail'	16	14	87.5 %	**Total**	**618**	**414**	**67.0 %**

Table 3. Noun lemmas with five or more plural instances in the PDT 2.0 data are arranged according to the percentage of instances assigned the pair/group meaning (i.e. the sum of the instances assigned the choices 2, 3 or 4) among all plural instances of these nouns in the final annotation.

Noun lemma	# of plural forms	Coverage	# of instances of agreement	Instances of agreement (%)	1 - plurality	2 - one pair/group	3 - several pairs/groups	4 - one pair/group or several pairs/groups	5 - cannot be resolved	6 - - - -
oko 'eye'	89	14.4 %	67	75.3 %	7	44	1	35	1	1
rodič 'parent'	87	28.5 %	56	64.4 %	1	28	0	9	49	0
ruka 'hand, arm'	81	41.6 %	67	82.7 %	4	35	2	40	0	0
doklad 'document'	35	47.2 %	27	77.1 %	26	5	0	3	1	0
bota 'shoe'	30	52.1 %	18	60.0 %	2	7	8	9	4	0
noha 'foot, leg'	20	55.3 %	19	95.0 %	3	12	1	4	0	0
cigareta 'cigarette'	17	58.1 %	15	88.2 %	14	1	2	0	0	0
křídlo 'wing'	17	60.8 %	14	82.4 %	12	3	0	2	0	0
kolej 'rail'	16	63.4 %	8	50.0 %	2	11	0	3	0	0
lék 'medicine'	16	66.0 %	8	50.0 %	8	0	2	0	6	0
zub 'tooth'	14	68.3 %	8	57.1 %	2	5	0	3	4	0
lyže 'ski'	13	70.4 %	10	76.9 %	0	2	0	11	0	0
vlas 'hair'	11	72.2 %	9	81.8 %	0	9	0	2	0	0
prst 'finger/toe'	10	73.8 %	7	70.0 %	1	8	0	1	0	0
brambor 'potato'	9	75.2 %	8	88.9 %	8	1	0	0	0	0
rameno 'shoulder'	9	76.7 %	9	100.0 %	1	6	0	2	0	0
ucho 'ear'	9	78.2 %	7	77.8 %	0	5	0	4	0	0
klíč 'key'	8	79.4 %	5	62.5 %	2	4	0	1	1	0
rukavice 'glove'	8	80.7 %	6	75.0 %	1	2	4	1	0	0
ledvina 'kidney'	7	81.9 %	6	85.7 %	0	4	0	3	0	0
těstovina 'pasta'	7	83.0 %	6	85.7 %	7	0	0	0	0	0
kulisa 'scene'	6	84.0 %	3	50.0 %	0	5	0	0	1	0
schod 'stair'	6	85.0 %	4	66.7 %	0	5	1	0	0	0
dvojče 'twin'	5	85.8 %	5	100.0 %	0	5	0	0	0	0
koleno 'knee'	5	86.6 %	4	80.0 %	1	2	0	2	0	0
kopačka 'football boot'	5	87.4 %	4	80.0 %	0	1	0	4	0	0
pouto 'tie'	5	88.2 %	5	100.0 %	0	1	0	4	0	0
Total	618	100.0 %	464	75.1 %	133	230	30	154	70	1

Table 4. Noun lemmas with five or more plural instances in the PDT 2.0 data, arranged according to their frequency. In the coverage column, the percentage of the instances of the first to the last of the listed nouns with respect to the total number of instances to be annotated is expressed. The inter-annotator agreement is specified for each noun both in number of instances and in percentage. In the right part of the Table, the number of the instances assigned the choices 1 to 5 (and 6) in the final annotation is shown for each noun.

	Choices by Annotator 2							
Choices by Annotator 1	1	2	3	4	5	6	7	Total
1 - plurality	**82**	8	1	2	1	3	0	97
2 - one pair/group	11	**350**	0	4	0	24	0	389
3 - several pairs/groups	0	0	**10**	0	2	7	0	19
4 - one pair/group or several pairs/groups	0	4	1	**4**	0	1	0	10
5 - cannot be resolved	0	0	0	0	**0**	1	0	1
6 - distributivity	1	10	1	0	4	**43**	0	59
7 - - -	3	0	0	0	0	0	**0**	3
Total	97	372	13	10	7	79	0	578

Table 5. Inter-annotator agreement in the annotation carried out on the spoken data of PDTSC. Numbers of instances assigned the same choice by both annotators are marked in bold on the diagonal.

of whether or not this meaning should be included in the forthcoming version of the treebank (PDT 3.0).

The 414 instances assigned the pair/group meaning (i.e. the choices 2, 3 or 4) during the annotation correspond to only 0.69 % of all 60,017 plural forms of nouns at the tectogrammatical layer. However, if we compared the number of instances of the pair/group meaning to the frequency of other attributes annotated at the tectogrammatical layer, namely to the frequency of functor values (i.e. dependency relations, semantic roles), 14 functors (out of 67) do not reach this number; e.g. the functor HER for modifications with the meaning of heritage or TFRHW for modifications with the temporal meaning "from when". There are also several grammatemes whose values are less frequent than the pair/group meaning (for instance, only 375 tectogrammatical nodes are assigned the value imp in the grammateme of verbal mood corresponding to the imperative mood of verbs).

4.2. A higher frequency in the spoken data

Taking into account the fact that, as already mentioned, in PDT 2.0 only written newspaper texts are involved, we were wondering whether the frequency of the pair/group meaning would be different (higher) in spoken data or in written data from other genres.

After the manual annotation of the PDT 2.0 data had been finished, a manual annotation was carried out on the data from the Prague Dependency Treebank of Spoken Czech (PDTSC), which is currently built at the Institute of Formal and Applied Linguistics at Charles University in Prague (Hajič et al., 2008).[15] The instances to be

[15]PDTSC is the Czech part of the Prague Dependency Treebank of Spoken Language (PDTSL, http://ufal.mff.cuni.cz/pdtsl).

Annotation choice	# of instances assigned	Percentage
1 - plurality	106	18.3 %
2 - one pair/group	380	65.7 %
3 - several pairs/groups	12	2.1 %
4 - one pair/group or several pairs/groups	7	1.2 %
5 - cannot be resolved	5	0.9 %
6 - distributivity	67	11.6 %
7 - - -	1	0.2 %
Total	578	100.0 %

Table 6. Annotation choices in the final annotation of the data from PDTSC

annotated were selected from the tectogrammatically annotated data of PDTSC (316 thousand tokens for which tectogrammatical annotation was available at that moment) using the same procedure as described in Sect. 3.1. The annotation was carried out by the same two annotators. The list of annotation choices was enriched with the choice distributivity for nouns in distributive contexts (see Sect. 3.3), so that the choice one pair/group or several pairs/groups was used only in clear contexts exemplified by the ex. (10).

The annotators agreed on 489 out of 578 annotated plural nouns, i.e. on 84.6 % of the instances, Kappa score 0.71. The choices assigned by the annotators are compared in Table 5, number of instances assigned the particular choices in the final annotation are listed in Table 6.

For the spoken data, a significantly higher inter-annotator agreement was achieved than for the written data. The percentage of the instances assigned the pair/group meaning among all annotated instances was also higher in the data from PDTSC than from PDT 2.0; see the sum of choices 2, 3 and 4 in Table 2 (67.0 %) vs. choices 2, 3, 4 and 6 in Table 6 (80.6 %; the new choice distributivity is another particular value of the pair/group meaning). This difference is related, among other facts, for instance to a higher frequency of every-day contexts and a lower frequency of figuratively used nouns and phrasems in the spoken than in the written data.

The hypothesis that the relatively low frequency of the pair/group meaning of the PDT 2.0 data has a relation to the type of texts involved in the treebank is further supported by a comparison with three large corpora of written Czech texts, namely with balanced copora which were built in the Czech National Corpus;[16] see Table 7. In the table, several corpora are compared only as to the number of plural forms of nouns from the working list of pair/group nouns; the manual annotation of the pair/group meaning was not provided for all the corpora. Nevertheless, we conclude on this background that the involvement of the annotation of the pair/group meaning in the PDT 3.0 is worth the effort.

[16] http://ucnk.ff.cuni.cz

Corpus	# of plural forms of nouns from the working list	Size of the corpus (# of tokens)	# of noun forms	# of plural forms of nouns
PDT 2.0	618	833,195 0.07 %	256,271 0.24 %	60,017 1.03 %
PDTSC	578	316,086 0.18 %	48,976 1.18 %	12,104 4.78 %
SYN2000	161,004	120,908,724 0.13 %	32,479,355 0.50 %	7,712,904 2.09 %
SYN2005	271,949	122,419,382 0.22 %	31,315,440 0.87 %	7,440,382 3.66 %
SYN2010	273,680	121,667,413 0.22 %	29,808,857 0.92 %	7,225,687 3.79 %

Table 7. Number of plural instances of the nouns for which the pair/group meaning is supposed to be prototypical (see the working list) in the PDT 2.0 and PDTSC data compared with three sub-corpora of the Czech National Corpus. The percentage stated bellow the numbers of instances in each row is the percentage of the plural forms of the nouns from the working list among all tokens, all noun forms and all plural forms of nouns, respectively, in each particular corpus.

5. Matching the annotation on the data

5.1. Inserting the manual annotation into the data

As explained in Section 2, the pair/group meaning is treated as a grammaticalized meaning constituting a new grammatical category of Czech nouns, which is closely related to the category of noun number. In FGD as well as in the annotation scenarios of PDT 2.0 and PDT 3.0, which are based on this theoretical framework, grammatical meanings are captured within the so-called grammatemes, which are attributes of nodes of the tectogrammatical tree. Grammatemes correspond to morphological categories, such as number with nouns, degree of comparison with adjectives or tense with verbs.[17]

For the purpose of including the pair/group meaning into the tectogrammatical annotation of PDT 3.0, a new grammateme typgroup was added to the existing set of 15 grammatemes used in PDT 2.0 (Mikulová et al., 2006). For the typgroup grammateme, two values were defined: group for the pair/group meaning and single for the meaning of single entities. To be able to represent all the semantic nuances distinguished in the manual annotation (choices 1 to 5 in Sect. 3) in the treebank data, the values of the grammateme typgroup must be combined with the values of the grammateme number. For each of the both grammatemes, a third value (nr, "not recognized") is used.

The annotation choices 1 to 5 were matched to the values of the grammatemes number and typgroup as follows. The annotation choice is given first, the arrow is followed by the values of the number and typgroup grammatemes, respectively:

- 1 - plurality → number=pl, typgroup=single
- 2 - one pair/group → number=**sg**,[18] typgroup=group
- 3 - several pairs/groups → number=pl, typgroup=group
- 4 - one pair/group or several pairs/groups → number=**nr**, typgroup=group
- 5 - cannot be resolved → number=**nr**, typgroup=nr

5.2. Automatic annotation of the pair/group meaning with remaining nouns

Since, according to our proposal, the pair/group meaning concerns potentially all Czech nouns, nouns which were involved neither in the list and, thus, nor in the manual annotation, were assigned a value of the typgroup grammateme fully automatically. A simple, two-step "algorithm" was provided for the automatic annotation: in the first step, nouns accompanied with a set numeral *jedny* 'one pair/group' (pluralia tantum excepted) were assigned the value group of the typgroup grammateme and the

[17] Unlike the mentioned categories, there are no grammatemes, e.g., for number of adjectives or case of nouns since these categories are imposed by agreement or government, respectively.

[18] Those number values are marked in bold which were changed from the pl value (as available in the PDT 2.0 annotation) to the marked value, influenced by the annotation of the pair/group meaning.

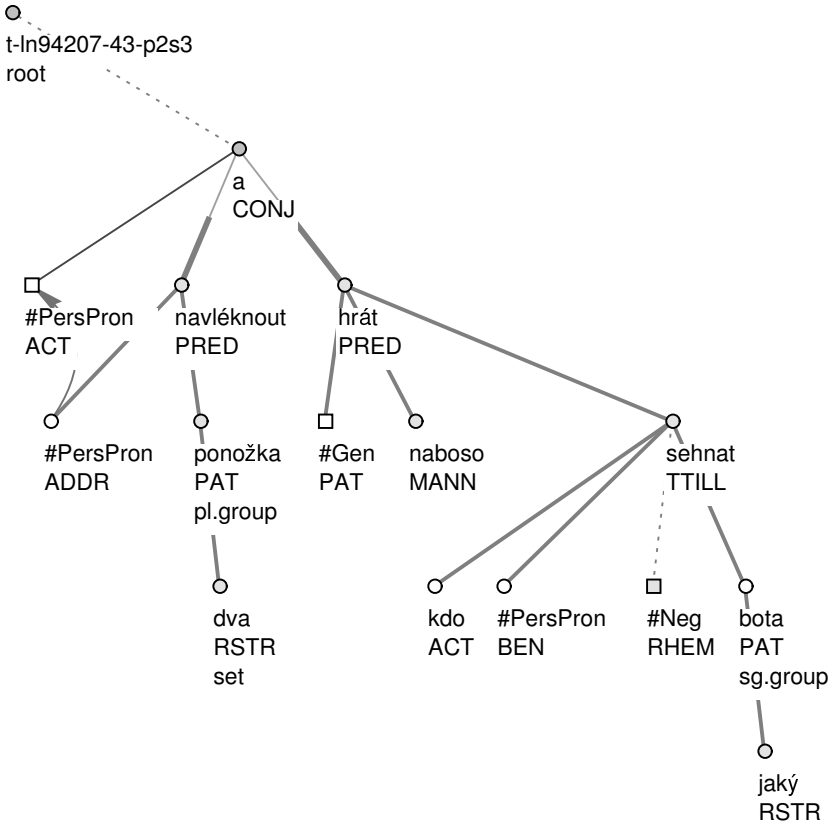


Figure 2. Tectogrammatical tree of the sentence "Navlékla bych si dvoje ponožky a hrála bych naboso, dokud by mi někdo nesehnal nějaké boty." 'I would put on two pairs of socks and would play barefooted until somebody would get some shoes for me.' For each node the tectogrammatical lemma, the functor and the values of the grammatemes number, typgroup and numertype are given; the numertype (value set) is assigned only to the node with the lemma "dva" 'two', which represents the set numeral "dvoje" 'two pairs/sets'.

value of the number grammateme was changed to sg in this connection; if the noun collocated with a set numeral of a higher numeric value (*dvoje* 'two pairs/groups', *troje* 'three pairs/groups' etc.), the value group was filled in the grammateme typgroup whereas the number grammateme remained unchanged (i.e. pl). Secondly, all the other nouns were assigned the value single in the typgroup grammateme, the value of the number grammateme was not changed in these cases, compared to the PDT 2.0 data. A sample tectogrammatical tree with nodes assigned the number and typgroup values is displayed in Fig. 2.

6. Conclusions

The main focus of the present paper has been laid on the manual assignment of the pair/group meaning with selected Czech nouns. With regard to the fact that the pair/group meaning is a very semantic issue, which is complicated with a strong ambiguity and has been studied only recently for Czech, the achieved inter-annotator agreement is rather satisfactory. The manual annotation was completed with the automatic assignment of typgroup values to the nouns which were not involved in the manual part. In PDT 3.0, thus, all nouns will be assigned the pair/group meaning.

We have in mind that there are several other issues in the domain studied here that are open for further investigation, for instance, (a) systematic study of the numerals from the point of view of their form and function with regard to their compatibility with the different types of nouns, (b) consequences of (a) for the deep-lexical representation of the different types of numerals in lexicon, outcoming from the preliminary solution given in (Razímová and Žabokrtský, 2006), (c) consideration about possibilities and limits of the semi-automatic annotation of quantified noun phrases as to the grammatemes number and typgroup and its implementation.

Acknowledgments

The work reported on in the present paper has been supported by the grant projects GA ČR P406/2010/0875 and MŠMT ČR LC536. We would like to thank to Eva Fernandez de Jesus and Hana Hanová who carried out the annotation described in the paper.

Bibliography

Cohen, Jacob. A coefficient of agreement for nominal scales. *Educational and Psychological Measurement*, 20:37–46, 1960.

Corbett, Greville G. *Number*. Cambridge University Press, Cambridge, 2000.

Dotlačil, Jakub. *Anaphora and Distributivity*. LOT, Utrecht, 2010.

Hajič, Jan, Silvie Cinková, Marie Mikulová, Petr Pajas, Jan Ptáček, Josef Toman, and Zdeňka Urešová. PDTSL: An annotated resource for speech reconstruction. In *Proceedings of the 2008 IEEE Workshop on Spoken Language Technology*, pages 93–96, Goa, India, 2008. IEEE.

Hajič, Jan, Jarmila Panevová, Eva Hajičová, Petr Sgall, Petr Pajas, Jan Štěpánek, Jiří Havelka, Marie Mikulová, Zdeněk Žabokrtský, and Magda Ševčíková Razímová. *Prague Dependency Treebank 2.0*. Linguistic Data Consortium, Philadelphia, 2006.

Komárek, Miroslav, Jan Kořenský, Jan Petr, and Jarmila Veselková et al. *Mluvnice češtiny 2*. Academia, Praha, 1986.

Lashevskaja, Olga N. Chislovoe oformlenie predmetnyh suschestvitel'nyh v distributivnom kontekste. *Nauchnaja i tekhnicheskaja informacija. Ser. 2. Informacionnyje processy i sistemy*, 11: 30–36, 1999.

Miko, František. *Rod, číslo a pád podstatných mien*. Vydavatel'stvo Slovenskej akadémie vied, Bratislava, 1962.

Mikulová, Marie, Alevtina Bémová, Jan Hajič, Eva Hajičová, Jiří Havelka, Veronika Kolářová, Lucie Kučová, Markéta Lopatková, Petr Pajas, Jarmila Panevová, Magda Razímová, Petr Sgall, Jan Štěpánek, Zdeňka Urešová, Kateřina Veselá, and Zdeněk Žabokrtský. Annotation on the tectogrammatical level in the Prague Dependency Treebank. Annotation manual. Technical Report 30, ÚFAL MFF UK, Prague, Czech Rep., 2006.

Panevová, Jarmila and Magda Ševčíková. Jak se počítají substantiva v češtině: poznámky ke kategorii čísla. *Slovo a slovesnost*, 72:163–176, 2011.

Razímová, Magda and Zdeněk Žabokrtský. Annotation of Grammatemes in the Prague Dependency Treebank 2.0. In *Proceedings of the LREC 2006 Workshop on Annotation Science*, pages 12–19, 2006.

Sgall, Petr. *Generativní popis jazyka a česká deklinace*. Academia, Praha, 1967.

Sgall, Petr, Eva Hajičová, and Jarmila Panevová. *The Meaning of the Sentence in Its Semantic and Pragmatic Aspects*. D. Reidel Publishing Company, Dordrecht, 1986.

Straková, Vlasta. Ke kategorii čísla. In *Rusko-české studie*, pages 75–85, 1960.

Address for correspondence:
Magda Ševčíková
sevcikova@ufal.mff.cuni.cz
Charles University in Prague
Faculty of Mathematics and Physics
Institute of Formal and Applied Linguistics
Malostranské náměstí 25
118 00 Praha 1, Czech Republic

PBML

The Prague Bulletin of Mathematical Linguistics
NUMBER 96 OCTOBER 2011 49–58

Ncode: an Open Source Bilingual N-gram SMT Toolkit

Josep M. Crego[a], François Yvon[ab], José B. Mariño[c]

[a] LIMSI-CNRS, BP 133, 91430 Orsay Cedex, France
[b] Université Paris-Sud, 91430 Orsay, France
[c] Universitat Politècnica de Catalunya, Jordi Girona S/N, Barcelona 08034, Spain

Abstract

This paper describes Ncode, an open source statistical machine translation (SMT) toolkit for translation models estimated as n-gram language models of bilingual units (*tuples*). This toolkit includes tools for extracting tuples, estimating models and performing translation. It can be easily coupled to several other open source toolkits to yield a complete SMT pipeline. In this article, we review the main features of the toolkit and explain how to build a translation engine with Ncode. We also report a short comparison with the widely known Moses system. Results show that Ncode outperforms Moses in terms of memory requirements and translation speed. Ncode also achieves slightly higher accuracy results.

1. Introduction

This paper describes Ncode, an open source statistical machine translation decoder and its companion tools. Ncode implements the bilingual n-gram approach to SMT as described in (Mariño et al., 2006; Crego and Mariño, 2007), which can be seen as an alternative to the standard phrase-based approach (Zens et al., 2002). Ncode main features include the use of multiple n-gram language models estimated over bilingual units, source words and/or target words or any factor decomposition, lexicalized reordering, several tuple (unigram) models, *etc.*. As for nearly all current statistical approaches to machine translation, these models are embedded in a linear model combination. Ncode splits the reordering and decoding problems of SMT in two separate modules, aiming at better tackling each of the problems. However, hard reordering decisions are avoided by means of using permutation lattices.

© 2011 PBML. All rights reserved.
Corresponding author: jmcrego@limsi.fr
Cite as: Josep M. Crego, François Yvon, José B. Mariño. Ncode: an Open Source Bilingual N-gram SMT Toolkit. The Prague Bulletin of Mathematical Linguistics No. 96, 2011, pp. 49–58. doi: 10.2478/v10108-011-0010-5.

The toolkit implements algorithms for tuple extraction, modeling estimation and translation. Several algorithms for optimization and evaluation are borrowed from distinct open source projects, embedded in our toolkit to accurately work with our translation engine. The decoder takes advantage of multi-threaded architectures, which are quickly becoming the norm. As far as memory is concerned, major improvements have been obtained by replacing the SriLM[1] interface (used in previous versions) with the new and much leaner kenlm[2] libraries. The toolkit is mainly written in *C++* and *Perl*, with special attention to clean code, extensibility and efficiency, and is available under an open-source license. It is mainly developed to run on Linux systems. Prerequisites to compile the decoder are kenlm and OpenFst[3] libraries. Prerequisites to run the entire system are SriLM and the minimum error rate training (Och and Ney, 2002) implementation available in the Moses[4] SMT toolkit. Note that the toolkit is able to build translation systems starting from a parallel set of word-aligned sentences and typically employs part-of-speeches to learn rewrite rules. Therefore, although they are not directly required by our SMT system, a word alignment and symmetrization algorithms as well as POS taggers for the source and target languages are needed to perform the entire SMT pipeline.

The toolkit was originally developed at UPC, further extended at LIMSI to its current state. It has been successfully used in a number of machine translation evaluations. A detailed description of the system with examples and full documentation is available in the LIMSI's web site[5]. The rest of this paper is organized as follows. In Section 2, we briefly introduce the bilingual n-gram approach to statistical machine translation. In Sections 3, 4 and 5, we detail the main components of the toolkit: training, decoding and tuning. After a short comparison with Moses in Section 6, we finally draw conclusions in Section 7.

2. Bilingual N-gram Approach to Statistical MT

The bilingual n-gram approach to SMT has been derived from the finite-state perspective (Casacuberta and Vidal., 2004). However, while translation models are implemented as weighted finite-state transducers in the finite-state perspective, our approach implements translation models as simple n-gram language models. The elementary translation units are *tuples*, that is pairs of variable-length sequences of source and target words. Hence, the translation model defines probability over sequences of tuples. Training such a translation model requires that (i) the source and

[1] http://www.speech.sri.com/projects/srilm/

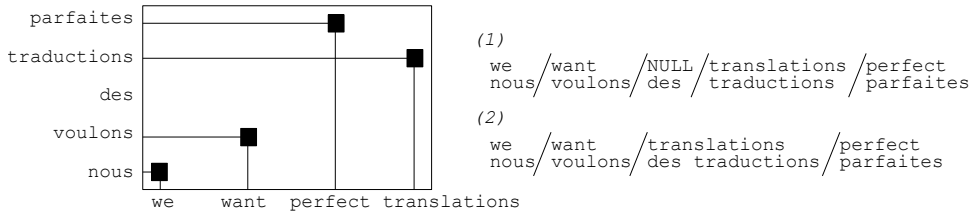
[2] http://kheafield.com/code/kenlm/

[3] http://www.openfst.org

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

[5] http://ncode.limsi.fr

Figure 1. Tuple extraction from a word-aligned sentence pair (English-French).



target side tuples are synchronized, *i.e.* that they occur in the same order in their respective languages and (ii) a joint segmentation of the source and target sentences in tuples is available, which uncovers the tuple boundaries. Given a parallel training corpus, two pre-processing steps are thus necessary to meet these requirements. First, word alignments are derived for each sentence pair; based on this information, a joint segmentation of the source and target sentences in tuples is then produced. The segmentation in tuples is made (almost) deterministic by enforcing the following constraints: (i) no word inside a tuple can be aligned to a word outside the tuple; (ii) segmentation must respect the order of words as they occur on the target side. Reordering is permitted in the source side so as to synchronize the source and target sides of a sentence; and (iii) no smaller tuples can be found without violating the previous constraints. Figure 1 presents a simple example illustrating the unique tuple segmentation for a given word-aligned pair of sentences. Note that the English source words *perfect* and *translations* have been reordered in the final tuple segmentation, while the French target words are kept in their original order. The resulting sequence of tuples *(1)* is further refined *(2)* to avoid *NULL* words in the source side of tuples. Refer to (Crego and Mariño, 2007) for further details on the tuple extraction process.

The bilingual n-gram language model expects synchronized tuple streams in training; likewise, it produces synchronized streams in inference. This means that the input source stream has to be reordered *prior to translation*, so as to reproduce the word order changes introduced during the training process. In our system, several possible reorderings of the source are considered in parallel. To this end, the sentence to be translated is first turned into a word lattice containing a set of promising reordering hypotheses. It is then pretty straightforward to search this lattice in a monotonous fashion for the best translation hypothesis. See (Crego and Mariño, 2007) for further details on word reordering.

3. Training a NCODE SMT system

In this section, we outline the training process of a NCODE system. Training basically involves the estimation of the set of models used by the decoder to perform machine translation. The script `training.perl` implements the training process, assuming source and target files, as well as the corresponding word alignment. It performs the following steps:

Tuple extraction From a word-aligned training bitext, tuples are extracted following the algorithm sketched in Section 2.

Tuple refinement Tuples with a *NULL* source side are then discarded by merging the unaligned target word with either the previous or the next unit (see Figure 1).

Tuple pruning and uncontextualized scores As word alignments are often noisy, we then filter tuples using a set of simple constraints. A tuple is discarded if it exceeds a maximum number of source (or target) words (`--max-tuple-length`); or a maximum ratio of source/target length (`--max-tuple-fert`). Additionally, only the n best translation choices for each tuple source side are considered (`--tuple-nbest`). Four scores are then associated with each tuple, corresponding to conditional probabilities computed as relative frequencies:

$$P_{rf}(e,f) = \frac{count(f,e)}{\sum_{f'} count(f',e)} \quad ; \quad P_{rf}(f,e) = \frac{count(f,e)}{\sum_{e'} count(f,e')} \tag{1}$$

where f and e respectively denote the tuple source and target side.
Word-based lexicon weights are also computed for each translation unit:

$$P_{lw}(e,f) = \frac{1}{(J+1)^I} \prod_{i=1}^{I} \sum_{j=0}^{J} P_{lex}(e,f) \quad ; \quad P_{lw}(f,e) = \frac{1}{(I+1)^J} \prod_{j=1}^{J} \sum_{i=0}^{I} P_{lex}(f,e) \tag{2}$$

where the probability distribution P_{lex} is computed based on counts using the word alignments.

Bilingual n-gram LM The training bitext expressed in the form of tuples is used to estimate a standard n-gram language model. It is estimated using SRILM, any of its modeling available options can be used (use `--options-bm`). When several bitexts are available, several LMs can be learned and used in parallel. In this case, `--name-src-bm` and `--name-trg-bm` are used to identify the potentially multiple language model files.
Tuples are typically built from words in the source and target sides, however, different factors may be used. *i.e.* tuples may be built from source POS tags and target lemmas (use `--train-src-bm train.f.pos --train-trg-bm train.e.lem`). Source and target factored training files must match with the alignment file, and are bound to contain the same number of sentences and words per sentence. Notice also that our current implementation considers **one single** factored tuple associated with each original word-based tuple (enabling a straightforward

implementation of factored language models), even though several different instances may exist. In such case, the most frequent is only considered.

Reordering rules Rewrite rules are automatically learned from the bitext word alignments. Following on the example of Figure 1, the rule *perfect translations* $\rightsquigarrow$ *translations perfect* produces the swap of the English words that is observed for the French and English pair. Note that such simple reordering patterns can be modeled using finite-state transducers. Typically, POS tags are used to increase the generalization power of such rules (use `--train-src-rules train.f.pos`), however, any other factor form can be used. Again, the source file used to extract reordering rules must be parallel to the original source training file.
In order to discard rules derived from noisy alignments, rules are pruned according to a length (`--max-rule-length`) and minimum probability threshold (`--max-rule-cost f`). The probability of each rule is estimated as:

$$P(f \rightsquigarrow f^r) = \frac{count(f, f^r)}{\sum_{f' \in perm(f)} count(f, f')} \tag{3}$$

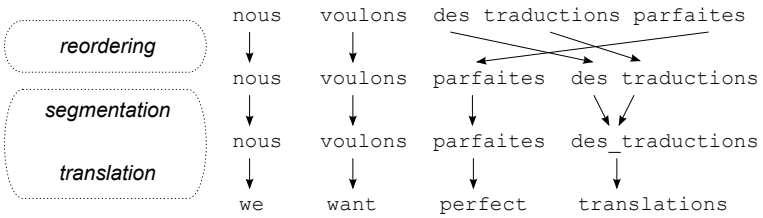
where f and f^r are the original (or left-hand side) and reordered (right-hand side) sequence of source words (or factors) of the considered rewrite rule, and $perm(f)$ is the set of permutations of f.

Lexicalized reordering NCODE implements the standard lexicalized reordering (Tillman, 2004) with four basic reordering types: **(m)**onotone order; **(s)**wap with previous tuple; **(f)**orward jump; **(b)**ackward jump. In addition, we also consider two aggregated types: **(d)**iscontinuous, corresponding to (b) and (f) and finally **(c)**ontinuous, corresponding to (m) and (s). In order to estimate these models, we count how often each tuple is found with each of the four orientation types ($orientation \in \{m, s, f, b\}$), and used the following smoothed maximum likelihood estimators ($\sigma = 1/\sum_o count(o, f, e)$):

$$P(orientation|f, e) = \frac{(\sigma/4) + count(orientation, f, e)}{\sigma + \sum_o count(o, f, e)} \tag{4}$$

Source-reordered n-gram LM N-gram language models estimated over the source words (or factors) of the training corpus are also estimated. Note that the training source words are first reordered following the tuple extraction process. The model scores a given source-side reordering hypothesis according to the reorderings performed in the training sentences following the word alignments. Again, the model is estimated as any standard n-gram language model, use `--options-sm` and `--name-src-unf` respectively to set the language model options and to identify the potentially multiple language model files. See (Crego and Yvon, 2009) for details.

Figure 2. Decoding with NCODE.



4. Decoding

Figure 2 details the various processing steps taken in our system: source words are first shuffled in various ways (*reordering*) so as to reproduce the target word order; source sentences are then segmented (*segmentation*); translations of each segment (*translation*) are produced in the last step. The final translation hypothesis is obtained by concatenation of such partial hypotheses. Note that multiple choices are considered at each step, defining the *decoding search space*.

As introduced in Section 1, our toolkit splits the reordering and decoding problems into separate modules (detailed below), aiming at better tackling each of the problems. The first module computes *reordering* hypotheses producing a word permutation lattice. The word lattice is then traversed in the second step, where *segmentation* and *translation* take place. An intermediate step is introduced after the reordering module, which only keeps those units than are useful to translate the input sentence.

Permutation lattice and test filtering Sentences to be translated are encoded as word lattices (use `binrules`) containing the most promising reordering hypotheses, so as to reproduce the word order modifications introduced during the tuple extraction process. Hence, reordering rules are applied on top of the input sentences to be translated. More formally, given an input sentence, f, in the form of a linear word automaton, and N optional reordering rules to be applied on f, each of which is represented by a finite-state transducer τ_i, the resulting reordering lattice f^* is obtained by the sequential composition of FSTs, as:

$$f^* = \tau_N \circ \tau_{N-1} \cdots \circ \cdots \tau_1 \circ f$$

where $\circ$ denotes the composition operation. Note that the sequence of FSTs is sorted according to the length of the left-hand side (LHS) of the rule. More specific rules, having a larger LHS, are applied (composed) first, in order to ensure the recursive application of the rules. Hence, some paths are obtained by applying reordering on top of already reordered paths. The applied rules can be limited to a maximum size of words (use `-maxr`) and to a maximum cost, or negative log of the probability estimated according to Equation 3 (use `-maxc`).

Once the word lattice is computed, the tuple vocabulary is then filtered for each input sentence (use `binfiltr`), removing all tuples but those units whose source-side matches any of the n-grams appearing in the lattice. The resulting file contains all the modeling information needed by the decoder to translate sentences, with the exception of n-gram language models scores. Tuples (source and target) size can be limited to a given number of words (use `-maxs`).

Search As for any statistical MT system, translation are built by searching a vast (theoretically infinite) set of translation hypotheses. Search stops when the most likely hypothesis covering (translating) the entire input sentence is attained (use `bincoder`). The algorithm is slightly modified to output n-best (use `-nbest`) hypotheses or the complete search graph (use `-ograph`).

As in most cases an exhaustive search is unfeasible pruning techniques are required, aiming at discarding partial hypotheses based on (more or less fair) hypotheses comparisons strategies. For NCODE, partial hypotheses are spread over multiple stacks[6] according to the source words they translate (use `-s 2J`). Each stack contains hypotheses that translate strictly the *same source words*.

A well-known heuristic technique then consists in discarding the worst ranked hypotheses of each stack. This idea can be implemented in several ways, the most common being known as *beam pruning* (use `-b i`), which expands the subset of the i-*best stack hypotheses*. Another common practice consists in considering only the i-*best translation choices* for each source segment (use `-t i`), what provides additional computational savings, but typically yields crudest heuristics, as the pruning is only based on non-contextualized translation scores. Hypotheses *recombination* is also implemented by NCODE, another risk-free way to discard hypotheses. However, the decoder is very often interested in the translation search graph, for which hypotheses recombinations need to be carefully recorded rather than discarded.

This organization of the search space ensures that, within a stack, hypotheses can be compared on a fair basis; this is at the cost, however, of inefficiencies when the size of the input lattice increases. As each node in this lattice corresponds to one stack of the decoder, we would need up to 2^J stacks to process a lattice encoding all the permutations of a sentence of length J. An alternative is organize stacks based simply on the *number of target words* (use `-s J`). This solution is more efficient, yet, it may bias the search towards translating first the easiest parts of the source sentence. This bias can be reduced using estimations of future costs, a workaround that has not yet been implemented, due to the complexity of accurately computing these estimations in our architecture.

Finally note that a simple way to increase efficiency when translating multiple input sentences over multi-threaded architectures consists of running on several

[6] More precisely: priority lists. We prefered to stick to the usual MT terminology here.

threads (use `-threads i`). Up to `i` sentences, depending on the server architecture and availability, will be translated *'in parallel'*.
Ncode's default policy to handle out-of-vocabulary words (source words never seen in the training text) is to output the source (unknown) word in the translation stream. An alternative option is to drop the unknown word (use `-dropunk`). The set of translation units used in the o*n*e-best translation option can be produced (use `-units`). A verbose mode (use `-verbose`) is also available which produces an *extremely detailed* output of the search process.

5. Tuning the Set of Models

As explained above, Ncode scores hypotheses based on a linear combination of several model scores. In order to obtain accurate translation results, the contribution of each model in the translation process needs to be tuned. Such optimization is carried out in our toolkit by the script `mert-run.perl`, which merely serves as wrapper for the MERT toolkit[7] implemented in the Moses toolkit. Once the optimal set of model weights is found, the script `mert-tst.perl` performs the translation of test sentences, using the models and configuration used in the optimization process, together with the optimized set of model weights.

6. A Comparison with Moses

In this section, we compare the performance of Ncode with that of Moses (Koehn et al., 2007), a state-of-the-art SMT system for phrase-based translation models.

The systems are compared on two different French-to-German translation tasks. The first (**news**) is a *small* size data task considering the *News Commentary* bitext made available in the *Sixth Workshop on Statistical Machine Translation*[8]. A second task (**full**) includes additional training data, ending up with a bitext of near four million sentences. Development work (tuning) is carried out for both systems using the *newstest2010* test set. Performance is evaluated over the *newstest2009* and *newstest2011* test sets available for the same translation task.

Thus, both approaches are compared using the same training corpora, target language model and word alignments, obtained performing Giza++[9]. The TreeTagger[10] toolkit was used to obtain the POS tags needed by Ncode. Afterwards, a default configuration of both toolkits is also used to perform training, tuning and decoding. Moses performs translation using the default 14 scores, while Ncode employs 2 bilingual *n*-gram language models. The first is estimated over tuples built from surface

[7] http://www.statmt.org/moses/?n=FactoredTraining.Tuning

[8] http://www.statmt.org/wmt11/translation-task.html

[9] http://www.fjoch.com/GIZA++.html

[10] http://www.ims.uni-stuttgart.de/projekte/corplex/TreeTagger/

Table 1. Performance statistics measured for Ncode *and* Moses *SMT systems.*

System	*Task*	*BLEU*		*#units*	*Speed*	*Memory*
		newstest2009	*newstest2011*			
Ncode	*news*	13.89	13.83	0.5	54.4	7.7
	full	15.09	15.26	7.5	33.9	9
Moses	*news*	13.70	13.51	7.5	23.1	7.9
	full	14.66	14.51	141	14.7	16

word forms, the second with tuples built from POS tags in both sides. Table 1 details performance measurements obtained for both systems. Translation accuracy is measured by the *BLEU* score (Papineni et al., 2002). The total number of units (in millions) obtained after training (tuples and phrases are limited to a maximum of 6 words) is displayed in the fifth column. Translation speed is reported in terms of words per second in the sixth column. Finally, the last column contains an approximation of the memory (in Mb) used by each decoder.

As can be seen, translation accuracy results are slightly higher for Ncode in both translation tasks and test sets, although all differences fall within the statistical confidence margin (add ± 1.50 BLEU for a 95% confidence level). In terms of data efficiency, Ncode clearly outperforms Moses: a unique segmentation is used to collect tuples, yielding a much smaller set of tuples than of phrases. In the case of the **full** task, the vocabulary of phrases is 20 times larger than the corresponding set of tuples. Ncode also outperforms Moses when considering the amount of memory needed by the decoder. Ncode needs about half of the memory needed by Moses (**full** data task). Notice that in the case of the **small** data task, the difference in the amount of memory needed is very small. This can be explained by the fact that both vocabularies of translation units are very small compared to the target language model, which account for most of the memory needs of both decoders. According to translation speed, measures were taken performing single-threaded translations by both decoders. Results show that Ncode is nearly twice faster than Moses.

7. Conclusions

We have described Ncode, an open source statistical machine translation toolkit for translation models estimated as n-gram language models. It can be downloaded from http://ncode.limsi.fr. We reviewed the main features that are currently implemented. Additionally, we carried out a short comparison with the widely known Moses SMT system. Ncode showed slightly higher French-to-German translation accuracy results than Moses. Our decoder also outperformed Moses in terms of memory requirements and translation speed.

Acknowledgements

This work was partially funded by OSEO under the Quaero program. The authors also want to thank those researchers of the UPC (Adrià de Gispert, Marta Ruiz, Patrik Lambert) and LIMSI (Alexandre Allauzen, Aurélien Max, Thomas Lavergne, Artem Sokolov) who contributed to create and extend the toolkit.

Bibliography

Casacuberta, F. and E. Vidal. Machine translation with inferred stochastic finite-state transducers. Computational Linguistics 30(4):205–225, 2004.

Crego, Josep M. and José B. Mariño. Improving SMT by coupling reordering and decoding. Machine Translation, 20(3):199–215, 2007.

Crego, Josep M. and Franc cois Yvon. Improving reordering with linguistically informed bilingual n-grams, August 2009.

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. Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL), demonstration session, Prague, Czech Republic, June, 2007.

Mariño, José B., Rafael E. Banchs, Josep M. Crego, Adrià de Gispert, Patrick Lambert, José A.R. Fonollosa, and Marta R. Costa-Jussà. N-gram-based machine translation. Computational Linguistics 32(4):7–549, 2006.

Och, Franz Josef and Hermann Ney. Discriminative training and maximum entropy models for statistical machine translation. Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL), 295–302, Philadelphia, PA, July, 2002.

Papineni, Kishore, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation, 2002.

Tillman, Christoph. A unigram orientation model for statistical machine translation. Proceedings of the HLT-NAACL'04, 101–104, Boston, MA, USA, May, 2004.

Zens, Richard, Franz Joseph Och, and Herman Ney. Phrase-based statistical machine translation. Proceedings of the 25th Annual German Conference on AI: Advances in Artificial Intelligence, 18–32, 2002.

Address for correspondence:
Josep M. Crego
jmcrego@limsi.fr
LIMSI-CNRS, BP 133, 91430 Orsay Cedex (France)

PBML

The Prague Bulletin of Mathematical Linguistics
NUMBER 96 OCTOBER 2011 59–67

Hjerson: An Open Source Tool for Automatic Error Classification of Machine Translation Output

Maja Popović

DFKI, Language Technology Group

Abstract

We describe Hjerson, a tool for automatic classification of errors in machine translation output. The tool features the detection of five word level error classes: morphological errors, reodering errors, missing words, extra words and lexical errors. As input, the tool requires original full form reference translation(s) and hypothesis along with their corresponding base forms. It is also possible to use additional information on the word level (e.g. POS tags) in order to obtain more details. The tool provides the raw count and the normalised score (error rate) for each error class at the document level and at the sentence level, as well as original reference and hypothesis words labelled with the corresponding error class in text and HTML formats.

1. Motivation

Human error classification and analysis of machine translation output presented in (Vilar et al., 2006) have become widely used in recent years in order to get detailed answers about strengths and weaknesses of a translation system. Another types of human error analysis have also been carried out, e.g. (Farrús et al., 2009) suitable for the Spanish and Catalan languages. However, human error classification is a difficult and time consuming task, and automatic methods are needed.

Hjerson is a tool for automatic error classification which systematically covers the main word level error categories defined in (Vilar et al., 2006): morphological (inflectional) errors, reordering errors, missing words, extra words and lexical errors. It implements the method based on the standard word error rate (WER) combined with the precision and recall based error rates (Popović and Ney, 2007) and it has been

© 2011 PBML. All rights reserved.
Corresponding author: maja.popovic@dfki.de
Cite as: Maja Popović. Hjerson: An Open Source Tool for Automatic Error Classification of Machine Translation Output. The Prague Bulletin of Mathematical Linguistics No. 96, 2011, pp. 59–67.
doi: 10.2478/v10108-011-0011-4.

tested on various language pairs and tasks. It is shown that the obtained results have high correlation (between 0.6 and 1.0) with the results obtained by human evaluators (Popović and Burchardt, 2011; Popović and Ney, 2011).

The tool is written in Python, and is available under an open-source licence. We hope that the release of the toolkit will facilitate the error analysis and classification for the researchers, and also stimulate further development of the proposed method.

2. Hjerson Toolkit

2.1. Algorithm

Hjerson implements the edit distance algorithm (Levenshtein, 1966) and identifies actual words contributing to the the standard Word Error Rate (WER) as well as to the recall/precision based Position-independent Error Rates called Reference PER (RPER) and Hypothesis PER (HPER) (Popović and Ney, 2007).

The dynamic programming algorithm for WER enables a simple and straightforward identification of each erroneous word which actually contributes to WER – the WER errors are marked as substitutions, deletions or insertions. The RPER errors are defined as the words in the reference which do not appear in the hypothesis, and analogously, the HPER errors are the words in the hypothesis which do not appear in the reference. Once the WER, RPER and HPER errors have been identified, the base forms for each word are added in order to perform error classification in the following way:

- inflectional error — a word which full form is marked as RPER/HPER error but the base forms are the same.
- reordering error — a word which occurs both in the reference and in the hypothesis thus not contributing to RPER or HPER, but is marked as a WER error.
- missing word — a word which occurs as deletion in WER errors and at the same time occurs as RPER error without sharing the base form with any hypothesis error.
- extra word — a word which occurs as insertion in WER errors and at the same time occurs as HPER error without sharing the base form with any reference error.
- incorrect lexical choice — a word which belongs neither to inflectional errors nor to missing or extra words is considered as lexical error.

Although the method is generally language-independent, availability of base forms for the particular target language is a requisite. If the error classification would be carried out without base forms, the morphological errors could not be detected and the rest of the results would be noisy, which would especially be problematic for morphologically rich(er) languages.

Figure 1 shows the workflow of the procedure. The details about the input and output options are described in following sections.

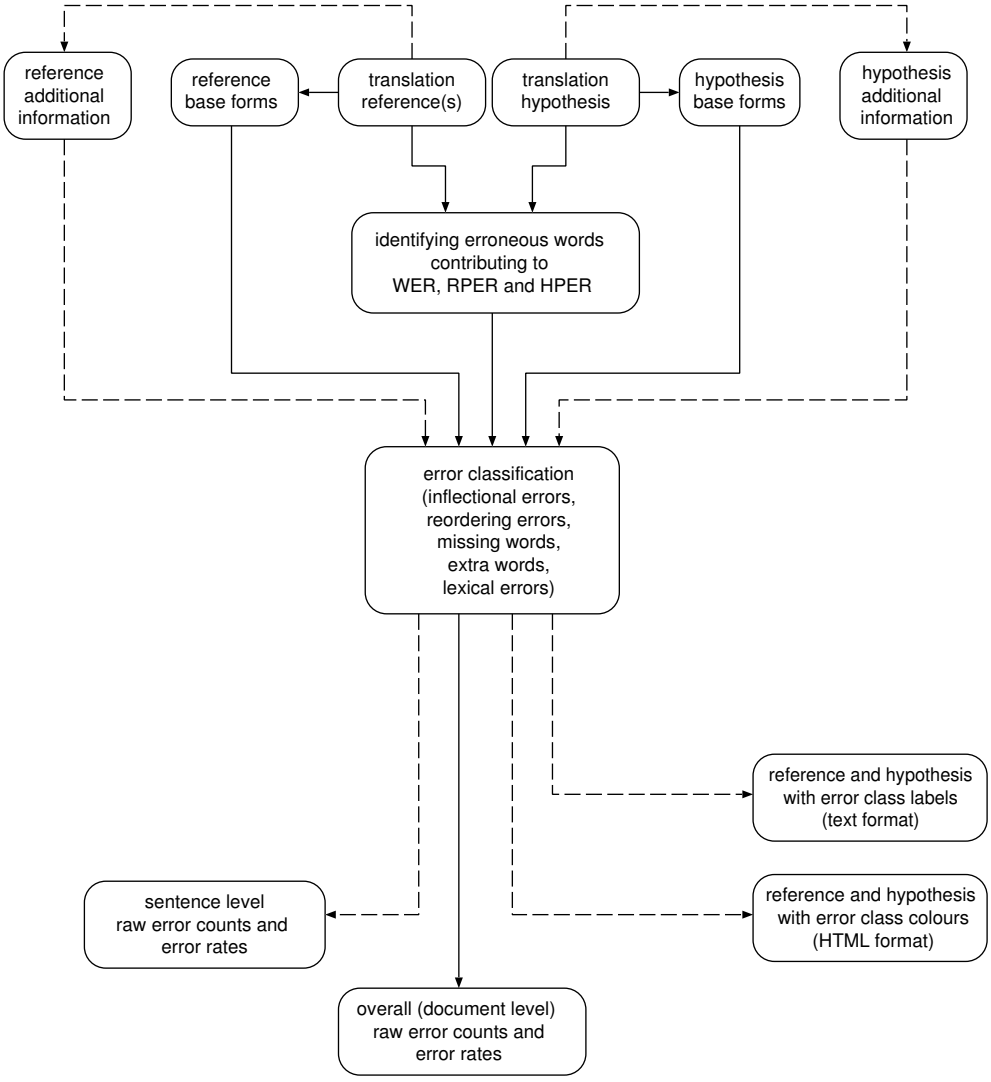


Figure 1. **Workflow of the automatic error classification by Hjerson:** *Continuous lines represent required inputs and default outputs, dashed lines represent optional inputs and outputs.*

2.2. Usage

Hjerson supports the option `-h/--help` which outputs a description of the available command line options.

The input options are:

- `-R, --ref` translation reference
- `-H, --hyp` translation hypothesis
- `-B, --baseref` reference base forms
- `-b, --basehyp` hypothesis base forms
- `-A, --addref` additional reference information
- `-a, --addhyp` additional hypothesis information

Inputs `-R`, `-H`, `-B` and `-b` are required. If any additional information at the word level is available (for example POS tags), it is possible to incorporate it by using options `-A` and/or `-a` in order to obtain more details. The additional information can be provided only for the reference, only for the hypothesis, for both, or not at all.

The required format for all input files is row text containing one sentence per line. In the case of multiple references, all available reference sentences must be separated by the symbol #. For the error classification, the reference sentence with the lowest WER score will be used.

The output options are:

- standard output
  The default output of the tool are the overall (document level) raw error counts and error rates (counts normalised over the reference or hypothesis length) for each of the five error classes:
  - reference and hypothesis inflectional errors (INFER);
  - reference and hypothesis reordering errors (RER);
  - missing words (MISER);
  - extra words (EXTER);
  - reference and hypothesis lexical errors (LEXER).

  For each class, the raw block error counts and block error rates are calculated as well, where block refers to a group of successive words belonging to the same error class. In addition, the values of the initial error rates, i.e. WER, RPER and HPER, are also provided together with their raw error counts.

- `-s, --sent sentence_errors.txt`
  The sentence level raw counts and error rates are written in the given text file `sentence_errors.txt`.

example.hyp	example.ref
This time , the reason for the collapse on Wall Street . The proper functioning of the market and a price .	This time the fall in stocks on Wall Street is responsible for the drop . The proper functioning of the market environment and the decrease in prices .

example.hyp.base	example.ref.base
This time , the reason for the collapse on Wall Street. The proper functioning of the market and a price .	This time the fall in stock on Wall Street be responsible for the drop . The proper functioning of the market environment and the decrease in price .

example.hyp.pos	example.ref.pos
DT NN , DT NN IN DT NN IN NP NP SENT DT JJ NN IN DT NN CC DT NN SENT	DT NN DT NN IN NNS IN NP NP VBZ JJ IN DT NN SENT DT JJ NN IN DT NN NN CC DT NN IN NNS SENT

Table 1. Example of translation hypothesis and its corresponding reference translation.

- `-c, --cats categories.txt`
  This option enables writing original reference and hypothesis words labelled with a corresponding error class in the given text file `categories.txt`. If additional information has been used, it is also contained in this file, which is suitable for potential further processing.

- `-m, --html categories.html`
  The results are writen in the given HTML file `categories.html` where the error classes are visualised by using colours.

An example of input and output files is shown in the next section.

2.3. Example

Table 1 presents an example of translation hypothesis consisting of two sentences and its corresponding reference translation together with their base forms as well as POS tags as additional information.

A program call without additional information:

```
hjerson.py --ref example.ref --hyp example.hyp --baseref example.ref.base
--basehyp example.hyp.base --html example.html --cats example.cats --sent
example.senterrorrates > example.totalerrorrates
```

will produce the following outputs:

- `example.totalerrorrates` – a file containing overall raw counts and error rates:

Wer:	15	53.57			
Rper:	11	39.29			
Hper:	5	22.73			
rINFer:	1	3.57	brINFer:	1	3.57
hINFer:	1	4.55	bhINFer:	1	4.55
rRer:	2	7.14	brRer:	1	3.57
hRer:	2	9.09	bhRer:	1	4.55
MISer:	6	21.43	bMISer:	4	14.29
EXTer:	2	9.09	bEXTer:	2	9.09
rLEXer:	4	14.29	brLEXer:	2	7.14
hLEXer:	2	9.09	bhLEXer:	2	9.09

  where prefixes ”r” and ”h” denote reference and hypothesis, and prefix ”b” denotes blocks.

- `example.senterrorrates` – a file containing raw counts and error rates for each sentence (sentence number is indicated for each error class, for example “1::rRer”).

- `example.html` – a HTML file containing original sentences with visualised error categories: pink (italic) inflectional errors, green (underlined) reordering errors, blue (bold) missing and extra words and red (bold+italic) lexical errors:

  REF: This time the ***fall in stocks*** on Wall Street **is responsible** for the **drop** .
  HYP: This time **,** the **reason** for the ***collapse*** on Wall Street .

  REF: The proper functioning of the market **environment** and **the decrease** ***in*** *prices* .
  HYP: The proper functioning of the market and ***a*** *price* .

- `example.cats` – a text file containing original words labelled with corresponding error category; the label “x” denotes absence of errors, i.e. correct word.

1::ref-err-cats: This~~x time~~x the~~x fall~~lex in~~lex stocks~~lex on~~x Wall~~x Street~~x is~~miss responsible~~miss for~~reord the~~reord drop~~miss .~~x
1::hyp-err-cats: This~~x time~~x ,~~ext the~~x reason~~ext for~~reord the~~reord collapse~~lex on~~x Wall~~x Street~~x .~~x

2::ref-err-cats: The~~x proper~~x functioning~~x of~~x the~~x market~~x environment~~miss and~~x the~~miss decrease~~miss in~~lex prices~~infl .~~x
2::hyp-err-cats: The~~x proper~~x functioning~~x of~~x the~~x market~~x and~~x a~~lex price~~infl .~~x

If POS tags are used as additional information:

```
hjerson.py --ref example.ref --hyp example.hyp --baseref example.ref.base
--basehyp example.hyp.base --addref example.ref.pos --addhyp example.hyp.pos
--html example.html --cats example.cats --sent example.senterrorrates >
example.totalerrorrates
```

the file `example.cats` will contain additional information together with error class labels:

1::ref-err-cats: This#DT~~x time#NN~~x the#DT~~x fall#NN~~lex in#IN~~lex stocks#NNS~~lex on#IN~~x Wall#NP~~x Street#NP~~x is#VBZ~~miss responsible#JJ~~miss for#IN~~reord the#DT~~reord drop#NN~~miss .#SENT~~x
1::hyp-err-cats: This#DT~~x time#NN~~x ,#,~~ext the#DT~~x reason#NN~~ext for#IN~~reord the#DT~~reord collapse#NN~~lex on#IN~~x Wall#NP~~x Street#NP~~x .#SENT~~x

2::ref-err-cats: The#DT~~x proper#JJ~~x functioning#NN~~x of#IN~~x the#DT~~x market#NN~~x environment#NN~~miss and#CC~~x the#DT~~miss decrease#NN~~miss in#IN~~lex prices#NNS~~infl .#SENT~~x
2::hyp-err-cats: The#DT~~x proper#JJ~~x functioning#NN~~x of#IN~~x the#DT~~x market#NN~~x and#CC~~x a#DT~~lex price#NN~~infl .#SENT~~x

The POS tags will also be visible in the HTML file:

REF: This#DT time#NN the#DT ***fall#NN in#IN stocks#NNS*** on#IN Wall#NP Street#NP **is#VBZ responsible#JJ** for#IN the#DT **drop#NN** .#SENT
HYP: This#DT time#NN **,#,** the#DT **reason#NN** for#IN the#DT ***collapse#NN*** on#IN Wall#NP Street#NP .#SENT

REF: The#DT proper#JJ functioning#NN of#IN the#DT market#NN **environment#NN** and#CC **the#DT decrease#NN *in#IN*** *prices#NNS* .#SENT
HYP: The#DT proper#JJ functioning#NN of#IN the#DT market#NN and#CC ***a#DT*** *price#NN* .#SENT

3. Conclusions

We presented Hjerson, a toolkit for automatic error classification which we believe will be of value to the machine translation community. It can be downloaded from `http://www.dfki.de/~mapo02/hjerson/`. And for those wondering: Hjerson is a detective solving mysteries (hidden error classes) – he is a recursively fictional character[1] in several books of Agatha Christie.

Acknowledgments

This work has partly been developed within the TARAXÜ project financed by TSB Technologiestiftung Berlin – Zukunftsfonds Berlin, co-financed by the European Union – European fund for regional development.

Bibliography

Farrús, Mireia, Marta R. Costa-jussà, Marc Poh, Adolfo Hernández, and José B. Mariño. Improving a Catalan-Spanish statistical translation system using morphosyntactic knowledge. In *Proceedings of the 13th Annual Conference of the European Association for Machine Translation (EAMT 09)*, pages 52–57, Barcelona, Spain, May 2009.

Levenshtein, Vladimir Iosifovich. Binary Codes Capable of Correcting Deletions, Insertions and Reversals. *Soviet Physics Doklady*, 10(8):707–710, February 1966.

Popović, Maja and Aljoscha Burchardt. From Human to Automatic Error Classification for Machine Translation Output . In *Proceedings of the 15th Annual Conference of the European Association for Machine Translation (EAMT 11)*, pages 265–272, Leuven, Belgium, May 2011.

[1] A fictional character in books written by a fictional character.

Popović, Maja and Hermann Ney. Word Error Rates: Decomposition over POS classes and Applications for Error Analysis. In *Proceedings of the 2nd ACL 07 Workshop on Statistical Machine Translation (WMT 07)*, pages 48–55, Prague, Czech Republic, June 2007.

Popović, Maja and Hermann Ney. Towards Automatic Error Analysis of Machine Translation Output. *Computational Linguistics*, 37(4):xx–xx, December (to appear) 2011.

Vilar, David, Jia Xu, Luis Fernando D'Haro, and Hermann Ney. Error Analysis of Statistical Machine Translation Output. In *Proceedings of the 5th International Conference on Language Resources and Evaluation (LREC 06)*, pages 697–702, Genoa, Italy, May 2006.

Address for correspondence:
Maja Popović
maja.popovic@dfki.de
German Research Center for Artificial Intelligence (DFKI)
Language Technology Group (LT)
Alt-Moabit 91c
10559 Berlin, Germany

PBML

The Prague Bulletin of Mathematical Linguistics
NUMBER 96 OCTOBER 2011 69–78

Margin Infused Relaxed Algorithm for Moses

Eva Hasler, Barry Haddow, Philipp Koehn

Institute for Language, Cognition and Computation, University of Edinburgh

Abstract

We describe an open-source implementation of the Margin Infused Relaxed Algorithm (MIRA) for statistical machine translation (SMT). The implementation is part of the Moses toolkit and can be used as an alternative to standard minimum error rate training (MERT). A description of the implementation and its usage on core feature sets as well as large, sparse feature sets is given and we report experimental results comparing the performance of MIRA with MERT in terms of translation quality and stability.

1. Introduction

1.1. Background

Statistical Machine Translation (SMT) systems usually consist of a number of models, each dealing with a particular aspect of the translation task. The core features of models like those described in (Koehn, 2010), i.e. phrase table, language model and reordering model, are likelihood-based features that are estimated in a generative fashion. These features are combined in a log-linear model as shown in equation (1), which produces a weighted score of all feature functions h_k given a source sentence $\mathbf{f}$, a target sentence $\mathbf{e}$ and a derivation $\mathbf{d}$.

$$P(\mathbf{e},\mathbf{d}|\mathbf{f}) = \frac{\exp \sum_{k=1}^{K} \lambda_k h_k(\mathbf{e},\mathbf{d},\mathbf{f})}{\sum_{\mathbf{e}',\mathbf{d}'} \exp \sum_{k=1}^{K} \lambda_k h_k(\mathbf{e}',\mathbf{d}',\mathbf{f})} \quad (1)$$

Feature functions can consist of the generative features mentioned above but can also be arbitrary features whose values are not to be interpreted as probabilities, e.g. a word or phrase penalty. It is quite straightforward to improve the discriminative

© 2011 PBML. All rights reserved.
Corresponding author: e.hasler@ed.ac.uk
Cite as: Eva Hasler, Barry Haddow, Philipp Koehn. Margin Infused Relaxed Algorithm for Moses. The Prague Bulletin of Mathematical Linguistics No. 96, 2011, pp. 69–78. doi: 10.2478/v10108-011-0012-3.

power of the model by adding feature functions h_k. For example we might want to use binary phrase features as in equation (2) to measure how much a particular phrase pair helps to discriminate between good and bad translations.

$$h_k(f_i, e_i) = \begin{cases} 1, & \text{if } f_i\text{="dieses Haus" and } e_i\text{="this house"} \\ 0, & \text{otherwise} \end{cases} \tag{2}$$

If we assign a weight λ_i to each of them, we can let the parameter tuning algorithm decide which features are useful for translation and which should be dropped. However, since the number of fine-grained features like these can easily grow in the thousands or millions, they pose a challenge for parameter tuning algorithms.

The Margin Infused Relaxed Algorithm (MIRA) (Crammer and Singer, 2003) is an online large margin algorithm that enforces a margin between different translations of the same sentence. This margin can be tied to a loss function which makes it straightforward to integrate BLEU (Papineni et al., 2002) or another quality measure. Given that we can provide the learning algorithm with good oracle translations, the model is tuned to score hypothesis translations with higher BLEU scores better than translations with lower BLEU scores. Picking oracle translations that represent good, reachable translations to update towards is an important part of the algorithm.

MIRA learns a weight $\mathbf{w}$ vector by additively updating the current decoder weights. After each new input sentence $f_i \in \{f_1, .., f_n\}$ was translated by the decoder, MIRA seeks the smallest update to the current weights subject to the following constraint. The difference in model scores, $\Delta\mathbf{h}_j \cdot \mathbf{w} = (\mathbf{h}(e_i^*) - \mathbf{h}(e_{ij})) \cdot \mathbf{w}$, between an oracle translation e_i^* and a hypothesis translation $e_{ij} \in \{e_{i1}, .., e_{im}\}$ must be at least as large as the loss $L(e_i^*, e_{ij}) = l_j$ between them. $\mathbf{h}(e_i)$ is a feature vector representation of translation e_i and the loss is defined as the difference in BLEU scores here but could be measured by other metrics as well.

The constrained optimization problem is illustrated in equation (3), where ξ is a non-negative slack variable[1], C is a positive *aggressiveness parameter* that controls the relative size of the update, t ranges over rounds of the algorithm and j ranges over a subset of hypothesis translations.

$$\begin{aligned} &\mathbf{w}_{t+1} = \text{argmin}_{\mathbf{w}} \frac{1}{2}\|\mathbf{w} - \mathbf{w}_t\|^2 + C \sum_j \xi_j \\ &\text{subject to} \\ &l_j - \Delta\mathbf{h}_j \cdot \mathbf{w} \leq \xi_j, \quad \forall j \in J \subseteq \{1, .., m\} \end{aligned} \tag{3}$$

[1]Slack variables are introduced when the data are not linearly separable, see (Cortes and Vapnik, 1995).

Constructing the Lagrangian of equation (3) and taking partial derivatives yields the update rule in equation (4) which is defined in terms of the dual variables α_j. The vector $\boldsymbol{\alpha}$ of dual variables constitutes the step size for MIRA. Equation (5) shows how to solve for α if there is only one constraint in the optimization problem. The parameter C functions as an upper bound for α in the dual formulation[2]. For large optimization problems, $\boldsymbol{\alpha}$ can be found using iterative algorithms such as the Hildreth algorithm (Censor and Zenios, 1997) or SMO (Platt, 1998).

$$\mathbf{w}_{t+1} = \mathbf{w}_t + \sum_j \alpha_j \Delta\mathbf{h}_j \tag{4}$$

$$\alpha = \min\left\{C, \frac{l - \Delta\mathbf{h} \cdot \mathbf{w}}{\|\Delta\mathbf{h}\|^2}\right\} \tag{5}$$

1.2. Motivation

Minimum Error Rate Training (MERT) is to date the most frequently used parameter tuning algorithm for SMT systems. It iteratively optimizes the parameters of an SMT model separately along each feature dimension and because of this, the number of parameters it can tune reliably ranges between only 15 to 30. The final weights depend on the given start weights which means that some prior knowledge about how a "good" set of feature weights looks like is advantageous. Another issue with MERT is that it uses random restarts and so the results can vary considerably between different runs.

MIRA has been suggested for tuning machine translation systems with larger features sets. Watanabe et al. (2007) and Chiang et al. (2009) added thousands of features to their baseline systems and showed improved translation quality after tuning the enhanced models with MIRA. Arun and Koehn (2007) explored training a phrase-based SMT system in a discriminative fashion with MIRA. McDonald et al. (2005) were the first to apply MIRA to train a dependency parser.

In order to promote further research for SMT systems in terms of feature engineering, it is important to have a method for tuning feature-rich models efficiently. The requirement for larger tuning sets when training sparse models suggests that online methods like MIRA will prove especially useful when scaling up discriminative training.

2. MIRA implementation for Moses

MIRA computes its weight updates by selecting hypothesis and oracle translations and solving an optimization problem with the constraints posed by these translations. Our implementation of MIRA solves a similar optimization problem to those of

[2] For details and derivations see (Crammer et al., 2006).

(Watanabe et al., 2007) and (Chiang et al., 2008) and was shown above in equation (3). The update for the optimization problem solved in every iteration of MIRA is computed with the Hildreth algorithm. The specific setup of the optimization problem can be controlled with parameters, for example, the number and type of hypothesis translations used for discriminative training can be varied.

Oracle translations are selected according to a modified decoder objective function: $\hat{e} = \arg\max_e$ (model score(e) + approx. BLEU score(e)). It is possible to use 3 different lists of hypothesis translations as suggested by Chiang et al., who used an n-best list according to the model score, a list of "good" translations (*hope*) according to the oracle selection objective and a list of "bad" translations (*fear*) according to $\hat{e} = \arg\max_e$ (model score(e) - approx. BLEU score(e)). Another option is to use only the *hope* and *fear* lists and leave out the n-best model translations.

The algorithm works by iterating over the training set sentence by sentence (or batch by batch), running the decoder on the current example to produce hypothesis translations. Given the resulting batch of translations, the feature vector representations are turned into constraints from which an update is computed and the algorithm moves on to the next example.

2.1. Main parameters

The following list shows the most important parameters for MIRA training. They were adapted from the literature about MIRA for SMT (mostly (Chiang et al., 2009), (Chiang et al., 2008), (Watanabe et al., 2007), (Arun and Koehn, 2007)). An epoch denotes a complete pass through the tuning data.

--hope-fear (def: true), **--model-hope-fear** (def: false), 2 n-best lists: constraints are formed between all pairs of *hope* and *fear* translations, or 3 n-best lists: each translation forms a constraint with the 1-best hope translation (oracle)

--learner Perceptron update or MIRA update (def: "mira")

--shuffle the development set may be shuffled to avoid sequence bias (def: false)

--average-weights the final weights can be computed over all seen weight vectors (def: false) or only those of the current epoch

--batch-size number of input sentences processed as a batch (def: 1)

--slack MIRA updates can be regularized (def: 0.01); smaller values mean more regularization, 0 means no regularization (parameter C in objective)

--sentence-bleu (def: true), **--history-of-oracles**, **--history-of-1best** (def: false) sentence-level BLEU with smoothed precision counts for ngrams with n > 1 or approximate document-level BLEU using a history as suggested by Chiang et al.

--mixing-frequency see §2.3 for description (def: 5)

--perceptron-learning-rate (default: 0.01), **--mira-learning-rate** (def: 1) learning rates for weight vector updates

--scale-update scale MIRA update by oracle BLEU score (better oracles yield larger updates, def: false)

2.2. Stopping criterion and final weight selection

MIRA stops when no update has been performed during a full epoch and by default, it also stops when during three consecutive epochs the sum of all updates in each dimension has not changed by more than a predefined value (default: 0.01). This is supposed to capture the point when no more important updates to the weight vector are made and hence convergence was reached.

However, the selection of the final weights depends on MIRA's performance on a development test set that is translated using the current decoder weights during training (the number of times is configurable). According to our experiments, selecting the best weights found during 5-10 training epochs yields good results and the algorithm does not seem to improve further with more training epochs.

It is also possible to set a decreasing learning rate that reduces the size of the updates as the training progresses (parameter `--decr-learning-rate`).

2.3. Parallelization

MIRA can be run on multiple processors to speed up training times. In general, parallelization for online learning methods is not straightforward, because the updates build on top of each other sequentially. (McDonald et al., 2010) proposed a variation of a parameter mixing strategy that has proven useful for log-linear models, called *iterative parameter mixing*. It splits the training data into *shards* and each of n processors working in parallel updates its weight vector only according to the examples in its shard. After each training epoch, the resulting n weight vectors are mixed together using mixture coefficients. McDonald et al. showed for a named-entity recognition and a dependency parsing task that iterative parameter mixing yields performance as good as or better than training serially on all data.

Our implementation makes use of the Message Passing Interface (MPI). It follows the description of iterative parameter mixing and parameterizes the number of times per epoch the weight vectors are mixed across processors (no mixture coefficients are used). When the mixing frequency is set to 0, no mixing of the current weight vectors is performed but the accumulated weights from all past updates are still averaged across processors before dumping the average weights to disk.

2.4. Adding new feature functions

How to add new feature functions to Moses is described on the Moses website `http://www.statmt.org/moses/?n=Moses.FeatureFunctions`. When using score producers with an unlimited number of score components (features), it makes sense to keep their weights separate from the moses.ini file because in that case, we do not have a predefined set of feature IDs. Implementing those kinds of features is described in paragraph Moses.SparseFeatureFunctions.

2.5. Usage

The MIRA implementation is currently located at *mosesdecoder/branches/mira-mtm5/*. In order to use multiple processors for MIRA, it should be run from a training script[3] that starts the requested number of mira processes with mpirun. The script also picks up weight files dumped by MIRA to automatically translate a given development test set with the respective weights and compute its BLEU score. When using sparse features on top of the core features, the training script will separate them from the dumped weight files and put them into a separate weight file that can be passed to the Moses decoder with the parameter `-weight-file`.

Caching of translation options in Moses should be switched off in the moses.ini file used by MIRA (`[use-persistent-cache] 0`) in order to keep translations options always up to date with respect to the current decoder weights.

Available parameters can be printed with the `--help` flag. To start MIRA, run: `mira-mtm5/mira/mira -f moses.ini -i source-file -r reference-file` or `mira-mtm5/mira/training-expt.perl -config expt.cfg -exec` and pass parameters and the path to MIRA in the config file.

3. Experiments

All experiments where carried out on the news commentary corpus, with development set nc-dev2007, dev. test set nc-test2007 (for selecting final weights) and test sets nc-devtest2007 and news-test2008, see (Callison-Burch et al., 2008). Experiments were configured to use one oracle and one hypothesis translation (1 hope/1 fear) and sentence-level BLEU. Results are reported for language pairs en-de, en-fr and de-en. We show how the performance of MIRA compares to the performance of MERT on a core feature set and then show some results of models with additional large set of sparse features. We also report observations on parallelization and start weights.

3.1. MERT and MIRA results for models with core features

Tables 1 and 2 compare the performance of MERT and MIRA on a model with 14 core features. The results for MIRA were obtained by running the algorithm for 10 epochs on the original tuning set as well as well as two shuffled versions of it (3 runs) in order to test how much the final results depend on the order of the tuning set. The MERT results were obtained by averaging the results of 3 runs to account for the randomness within MERT. The tables show averaged values over the different runs as well as the standard deviation of BLEU and length ratio for the development test set. We can see that for 7 out of 9 compared BLEU scores, MIRA yields equivalent or better results, while the standard deviation shows that the order of the tuning sentences does not have a big impact on the final results for MIRA.

[3] *mira-mtm5/mira/training-expt.perl*, sample config file: *expt.cfg*

Lang. pair	Avg BLEU(dev test)	σ	Avg BLEU(test1)	Avg BLEU(test2)
en-de	17.6 (0.988)	0.083	**15.1** (0.966)	11.0 (1.046)
en-fr	28.2 (1.000)	**0.045**	**15.2** (1.125)	17.7 (1.016)
de-en	26.5 (1.012)	0.082	22.9 (1.048)	**15.5** (0.950)

Table 1. Average results of 3 MERT runs, news commentary data, length ratio in brackets

Lang. pair	Avg BLEU(dev test)	σ	Avg BLEU(test1)	Avg BLEU(test2)
en-de	**17.7** (0.981)	**0.013**	14.9 (0.957)	**11.1** (1.041)
en-fr	**28.3** (0.994)	0.077	**15.2** (1.119)	**17.8** (1.011)
de-en	**26.6** (1.000)	**0.041**	**23.2** (1.034)	15.4 (0.939)

Table 2. Average results of 3 MIRA runs (10 epochs), news commentary data, length ratio in brackets

Table 3 shows results on the two test sets when choosing the best weights from the first 5 epochs. The results are only slightly different (some even better) than the results for 10 epochs of training, which suggests that training for 5 epochs might already be sufficient. MERT using 8 threads on a machine with 8 CPUs took 10-21 hours for training (for 7-14 iterations), MIRA using 8 parallel processors took 4 hours for 5 iterations and 8 hours for 10 iterations. If a development test set is used to determine the final weight vector, some extra resources are needed for decoding that set.

3.2. MIRA results for models with large feature sets

Table 4 compares results on the development test set for a model with core features and two models with sparse target bigram (TB) features, one using word bigrams with 33,300 active features and the other using POS bigrams with 1,400 active features. Even though these features do not seem to improve translation performance very much[4], the results show that MIRA can deal with a large number of features and manages to train the core weights properly at the same time.

[4] Also the results on the test sets varied only slightly (less than $\pm$ 0.1 BLEU).

Lang. pair	Avg BLEU(dev test)	σ	Avg BLEU(test1)	Avg BLEU(test2)
en-de	**17.6** (0.970)	**0.024**	14.8 (0.945)	**11.2** (1.031)
en-fr	28.0 (0.987)	0.059	**15.3** (1.112)	**17.8** (1.005)
de-en	**26.5** (0.997)	**0.039**	**23.3** (1.030)	15.3 (0.936)

Table 3. Average results of 3 MIRA runs (5 epochs), news commentary data, length ratio in brackets

Lang. pair	en-de
core features	17.7 (0.981)
core + word TB features	17.8 (0.984)
core + POS TB features	17.7 (0.986)

Table 4. Average BLEU scores on dev. test set (avg. of 3 MIRA runs) over 10 epochs, news commentary data, length ratio in brackets.

Feature name	Feature weight
Distortion	0.207147
WordPenalty	-1.34204
LM	0.645341
dlmb_<s>:ART	0.247516
dlmb_<s>:NN	-0.10823
dlmb_ADJ:NN	0.137049
dlmb_NN:ADJ	-0.164686

Table 5. Example feature weights of model with core + TB features

Table 5 shows some of the core feature weights and the weights of some frequently occurring sparse features. We can see that the bigram feature dlmb_<s>:ART got a positive weight while the feature dlmb_<s>:NN got a negative weights, which means that the model prefers German sentences starting with articles to those starting with nouns. The model also learned that an adjective is likely to preceed a noun in German while it is not likely to follow a noun.

3.3. Parallelization

We ran experiments with different numbers of processors to investigate the variance between different parallel setups. Table 6 shows results for the same MIRA setup with 1, 2, 4, and 8 processors, all with a mixing frequency of 5. Since parallelization works by dividing the development set into shards that are sent to different processors, doubling the number of processors reduces the training time by half. Even though there is some variation in the results, we cannot observe a general tendency across language pairs that increasing the number of processors would change the results in a systematic way. The difference in BLEU scores within the same language pair was at most 0.2 which can be considered a negligible loss.

3.4. Start weights

MERT is usually initialized with feature weights that yield better performance than uniform weights according to past experience. The reported results for the MIRA experiments were achieved with uniform start weights (all weights 0.1), while the MERT experiments were initialized with the following weights: language model=0.5, distortion/reordering=0.3, translation features=0.2, word penalty=-1.

In the experiment reported in table 7, the development of the word penalty weight for uniform and preset start weights is shown. The reported values were measured at the end of each epoch, for 10 epochs. Even though the uniform weight started at

Lang. pair	# processors	Best BLEU(dev. test set)
en-de	1	17.7 (0.985)
	2	17.7 (0.977)
	4	17.7 (0.975)
	8	17.7 (0.973)
en-fr	1	28.3 (1.002)
	2	28.4 (0.997)
	4	28.2 (1.002)
	8	28.3 (0.999)
de-en	1	26.6 (1.007)
	2	26.6 (1.007)
	4	26.6 (1.005)
	8	26.5 (1.003)

Table 6. Varying the number of processors with a mixing frequency of 5, best results on dev. test set during 10 epochs, length ratio in brackets

WP start	1	2	3	4	5	6	7	8	9	10
0.1	-0.26	-0.60	-0.85	-1.00	-1.14	-1.25	-1.33	-1.41	-1.50	-1.54
-1	-1.09	-1.22	-1.32	-1.41	-1.46	-1.53	-1.58	-1.61	-1.64	-1.67

Table 7. Word penalty weight after each of 10 epochs, for uniform and preset start weights.

0.1 and the preset weight at -1, they became quite similar after a few epochs. The best result on the development test set was BLEU=17.68 with uniform start weights and BLEU=17.66 with preset start weights which shows that uniform initialization does not harm the performance of MIRA. However, after the first epoch, the development test performance was BLEU=17.14 for uniform start weights and BLEU=17.65 for preset start weights, indicating that good start weights result in shorter training times.

4. Conclusions

We presented an open-source implementation of the Margin Infused Relaxed Algorithm for the Moses toolkit. We reported results on core feature sets as well as large, sparse feature sets, showing that MIRA yields comparable performance to MERT on the core features and is able to handle much larger feature sets. We have also given evidence that MIRA can be run on parallel processors with negligible or no loss and that it works well with uniform start weights. In the future we want to scale up to larger training sets and explore new ways of integrating sparse features.

Bibliography

Arun, A. and P. Koehn. Online Learning Methods For Discriminative Training of Phrase Based Statistical Machine Translation. In *MT Summit XI, 2007*, Copenhagen, September 2007.

Callison-Burch, C., C. Fordyce, P. Koehn, C. Monz, and J. Schroeder. Further meta-evaluation of machine translation. In *Proceedings of the Third Workshop on Statistical Machine Translation*, pages 70–106, Columbus, Ohio, June 2008. ACL.

Censor, Y. and S. A. Zenios. *Parallel Optimization - Theory, Algorithms, and Applications*. Oxford University Press, New York/Oxford, 1997.

Chiang, D., Y. Marton, and P. Resnik. Online large-margin training of syntactic and structural translation features. In *Proceedings of EMNLP 08*, Morristown, NJ, USA, 2008. ACL.

Chiang, D., K. Knight, and W. Wang. 11,001 new features for statistical machine translation. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the NACL*, Stroudsburg, PA, USA, June 2009. ACL.

Cortes, C. and V. Vapnik. Support-vector networks. In *Machine Learning*, pages 273–297, 1995.

Crammer, K. and Y. Singer. Ultraconservative online algorithms for multiclass problems. *Journal of Machine Learning Research*, 3(4-5):951–991, January 2003.

Crammer, K., O. Dekel, J. Keshet, S. Shalev-Shwartz, and Y. Singer. Online passive-aggressive algorithms. *JOURNAL OF MACHINE LEARNING RESEARCH*, 7:551–585, 2006.

Koehn, P. *Statistical Machine Translation*. Cambridge University Press, 2010.

McDonald, R., K. Crammer, and F. Pereira. Online Large-Margin training of dependency parsers. In *Proceedings of ACL*, 2005.

McDonald, R., K. Hall, and G. Mann. Distributed Training Strategies for the Structured Perceptron. In *Human Language Technologies: The 2010 Annual Conference of the NACL*, pages 456–464, Los Angeles, California, 2010. ACL.

Papineni, Kishore, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *ACL 02: Proceedings of the 40th Annual Meeting of the ACL*, pages 311–318, Morristown, NJ, USA, July 2002. ACL.

Platt, J. C. Sequential Minimal Optimization: A Fast Algorithm for Training Support Vector Machines, 1998.

Watanabe, T., J. Suzuki, H. Tsukada, and H. Isozaki. Online large-margin training for statistical machine translation. In *Proceedings of EMNLP-CoNLL*, pages 764–773, Prague, June 2007. ACL.

Address for correspondence:
Eva Hasler
e.hasler@ed.ac.uk
Informatics Forum, 10 Crichton Street
Edinburgh EH8 9AB, UK

PBML

The Prague Bulletin of Mathematical Linguistics
NUMBER 96 OCTOBER 2011 79–88

Addicter: What Is Wrong with My Translations?

Daniel Zeman[a], Mark Fishel[b], Jan Berka[a], Ondřej Bojar[a]

[a] Charles University in Prague, Faculty of Mathematics and Physics, Institute of Formal and Applied Linguistics
[b] University of Zurich, Institute of Computer Linguistics

Abstract

We introduce Addicter, a tool for Automatic Detection and DIsplay of Common Translation ERrors. The tool allows to automatically identify and label translation errors and browse the test and training corpus and word alignments; usage of additional linguistic tools is also supported. The error classification is inspired by that of Vilar et al. (2006), although some of their higher-level categories are beyond the reach of the current version of our system. In addition to the tool itself we present a comparison of the proposed method to manually classified translation errors and a thorough evaluation of the generated alignments.

1. Introduction

Most efforts on translation evaluation to date concentrate on producing a single score – both in manual evaluation (HTER, fluency/adequacy, ranking) and automatic metrics (WER, BLEU, METEOR, TER, etc.). Such evaluation techniques are convenient for comparing two versions of a system or of competing systems but they do not provide enough detail to steer further development of the system.

If the score is unsatisfactory, it is necessary to know *what* exactly went wrong in order to improve it. Some metrics provide some further details (e.g. unigrams matched by BLEU) but we may be more interested in the frequency of errors of a particular type – e.g. erroneous inflection of an otherwise correct lemma. To achieve that, we need to closely inspect the system output *and input* (including the training corpus).

Addicter (standing for Automatic Detection and DIsplay of Common Translation ERrors) is a set of open-source tools that automate these analysis tasks partially or fully. The main tools include automatic translation error analysis, a training and testing corpus browser and word (or phrase) alignment info summarization.

© 2011 PBML. All rights reserved.
Corresponding author: zeman@ufal.mff.cuni.cz
Cite as: Daniel Zeman, Mark Fishel, Jan Berka, Ondřej Bojar. Addicter: What Is Wrong with My Translations?. The Prague Bulletin of Mathematical Linguistics No. 96, 2011, pp. 79–88. doi: 10.2478/v10108-011-0013-2.

Addicter is powered by Perl scripts that generate HTML reports; the viewer proper is any web browser providing a cheap and portable text-oriented GUI. In addition to static HTML reports, there is a possibility of dynamic web pages to enable the processing of large corpora without generating millions of files, most of which nobody will look at. The dynamic approach enables easy access to all occurrences of any word in the corpus. Dynamic content viewing requires a locally installed web server[1].

For most part, Addicter relies on the parallel corpora being word-aligned. A lightweight narrow-scope monolingual word aligner (that will be described later on) is included in the tool set, but it is just as possible to use an external word aligning tool, such as GIZA++ (Och and Ney, 2003) or Berkeley aligner (Liang et al., 2006).

Section 2 describes the components of Addicter. In Section 3, we present an initial evaluation, based on a corpus of automatic English-Czech translations with manually labelled translation errors. We conclude by describing related work in Section 4.

2. Addicter Components

Addicter consists of a monolingual aligner, error detector and labeler, corpus browser and alignment summarizer.

Detailed instructions on downloading, installing and using Addicter can be found at `https://wiki.ufal.ms.mff.cuni.cz/user:zeman:addicter`.

2.1. Monolingual Aligner

The monolingual alignment component finds the word-to-word correspondence between the hypothesis and reference translations. In this lightweight approach we produce only *injective* alignments, i.e. all words are aligned at most once.

The aligner accepts factored input to support the usage of linguistic analysis tools: each token consists of a number of factors, separated by a vertical bar, for example `joined|join|verb-3prs-past`. Thus, in addition to surface forms, it is possible to align translations based on lemmas (for detecting wrong word forms of correct lemmas), synsets (for detecting synonymous translations) or any other categories.

The main difficulty in finding a word alignment between the hypothesis and reference is ambiguity, caused by frequently present repeated tokens (punctuation, particles), words sharing the same lemma, etc.

Here we approach the problem of resolving ambiguity by introducing a first-order Markov dependency for the alignments, stimulating adjacent words to be aligned similarly, which results in a preference towards aligning longer phrases. The approach is very similar to bilingual HMM-based word alignment (Vogel et al., 1996), except here the probability distributions of the model are hand-crafted to only allow aligning tokens with the same factors; considering the injectivity requirement, repeating words

[1] Such as the freely available multi-platform Apache (`http://httpd.apache.org/`)

Source	In the first round, half of the amount is planned to be spent.
Reference	V prvním kole bude použita polovina částky.
Reference gloss	*In the first round will be used half of amount.*
Google output	V prvním `punct`::kole, `extra`::což `extra`::je `ops`::polovina této částky má být `form`::utracen.

Figure 1. Example of manually flagged translation errors. The flags in the last line describe the differences between the reference and hypothesis – e.g. `extra` marks superfluous hypothesis words, and `ops` marks the beginning of a misplaced phrase.

are allowed to remain unaligned to make way for other, potentially better alignments of the same hypothesis word. The model has the advantages of HMM-based word alignment, while the lack of a learning phase enables the application of the model to sets of varying sizes starting with single sentences.

As a result of aligning only tokens with equal factors, this method produces high-precision alignments, with possible low coverage. That also means that wrong lexical choices cannot be detected with this alignment method alone.

2.2. Error Detector and Labeler

Based on the reference-hypothesis alignment, this component automatically finds and identifies translation errors in the hypothesis. Similarly to state-of-the-art approaches to automatic translation evaluation our method compares the hypothesis to a reference translation. To alleviate the problems that come with matching a translation to a single reference, the method supports taking into account multiple references. In the current version analysis is done on the word-by-word basis, using injective alignments, such as the output of our lightweight aligner.

The translation error taxonomy is taken from the work of Bojar (2011), which in turn is based on the taxonomy, proposed by Vilar et al. (2006). An example of a manually annotated translation is given in Figure 1. The error flags and the methods of finding and labelling them are presented in the following.

Lexical Errors

- unaligned words in the reference are marked as missing words; these are further classified into content (`missC`) and auxiliary (`missA`) words using POS tags;
- unaligned words in the hypothesis are marked as untranslated if present in the source sentence (`unk`) and superfluous (`extra`) otherwise;
- aligned words with different surface forms but same lemmas are marked (`form`);

- aligned words with different lemmas can either be synonyms or wrong lexical choices (`disam` and `lex`, respectively); telling them apart is left for future work and the errors are tagged with a joint flag;
- tokens differing in punctuation symbols only are flagged as (`punct`);
- errors of a higher level (such as idiomatic or style errors) are currently not covered.

Detecting Order Errors

The common approach to evaluating order similarity is to calculate order similarity metrics, e.g. Birch et al. (2010). Here however, we aim at detecting misplaced words explicitly to provide a great deal more detail than general similarity.

We approach this task by doing a breadth-first search for fixing the order in the aligned hypothesis words. The weighted directed graph for the search is such that

- there is one node per every permutation of the hypothesis,
- there is an arc between two nodes only if the target node permutation differs from the source permutation by two adjacent symbols,
- the arc weight is 1; in order to enable block shifts, the weight is 0 if the current arc continues shifting a token in the same direction.

As a result, switched word pairs can be marked as short-range order errors (`ows`); a word misplaced by several positions is marked as a long-range order error (`owl`). The phrase reordering errors (`ops` and `opl` in the taxonomy of Bojar (2011)) are left uncovered because of the word-based nature of the approach.

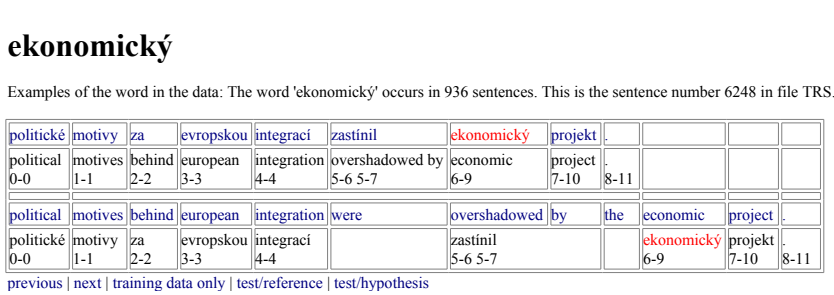
Multiple Reference Handling

A single source sentence can have several correct translations, and a translator should be allowed to produce any one of them. Therefore our evaluation method includes support for multiple reference translations. Alignments between the hypothesis and every reference translation are found. Based on that, errors are determined with respect to every reference translation. Finally, only the reference with the fewest errors in total is used and the respective errors are reported.

2.3. Test and Training Data Browser

The browser components enable the user to comfortably traverse the test or training corpora. Word alignment is used as well; unlike in the translation error component, many-to-one alignments are supported.

Alignments of the training corpus can be obtained with any bilingual word aligners. The test corpus can either be aligned with Addicter's own monolingual aligner or with bilingual aligners. Since the size of a typical test corpus can be insufficient for bilingual aligners to be trained directly on it, a feasible alternative is to independently



ekonomický

Examples of the word in the data: The word 'ekonomický' occurs in 936 sentences. This is the sentence number 6248 in file TRS.

politické	motivy	za	evropskou	integrací	zastínil	ekonomický	projekt	.			
political 0-0	motives 1-1	behind 2-2	european 3-3	integration 4-4	overshadowed by 5-6 5-7	economic 6-9	project 7-10	8-11			
political	motives	behind	european	integration	were	overshadowed	by	the	economic	project	.
politické 0-0	motivy 1-1	za 2-2	evropskou 3-3	integrací 4-4		zastínil 5-6 5-7			ekonomický 6-9	projekt 7-10	. 8-11

previous | next | training data only | test/reference | test/hypothesis

Figure 2. Training data viewer with a sentence pair for the Czech word ekonomický.

align both the reference and the hypothesis to the source text (by additionally using a large bilingual corpus) and to extract monolingual alignment from there.

The test data browser can both generate static reports and work together with a web server and generate dynamic content. Every separate page presents the source sentence, the hypothesis and reference translations. Alignments between the three sentences and automatically identified translation errors are listed as well.

The training data browser is constricted to dynamic content generation; it enables browsing both through training datasets and phrase tables. A sample screenshot is in Figure 2: both source and the translation are equipped with the corresponding words from the other language and the alignment links that lead to them. Every displayed word links to a separate page, listing all examples of its occurrence.

Currently the browsers are purely surface form-based. In order to make Addicter more suitable for highly inflecting languages (especially Slavic, Uralic, Turkic languages, etc.) it is necessary to enable browsing different forms of the same lemma; currently lemmatization is one of the main plans for future work.

2.4. Alignment Summarizer

The alignment summarization component displays the frequency-ordered list of all words or phrases, that were aligned in the training corpus, together with their counterparts, see Figure 3. Similarly to the training corpus browser, by clicking on aligned counterparts one can navigate through the translation space.

3. Experimental Evaluation

In this section we evaluate Addicter's monolingual alignment and translation error detection and labelling components by comparing them to their respective references, done manually. Evaluating the other components requires feedback from many users and is beyond the scope of this paper.

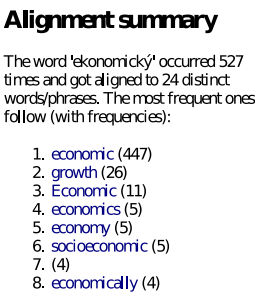


Figure 3. Most frequent English counterparts of the Czech word ekonomický*. Line 7 indicates that in 4 cases the word was unaligned.*

3.1. Target Corpus

To the best of our knowledge the only (publicly available) corpus of hypothesis translations marked with translation errors is the one of Bojar (2011). It contains four English-to-Czech news text translations from the WMT'09 shared task and consists of 200 sentences from each translation, tagged with translation errors. The translation error taxonomy used in this dataset and in Addicter is adapted from Vilar et al. (2006).

Hypothesis translation words are manually annotated with flags such as, for example, `lex`, or `form` indicating errors from the Vilar taxonomy. Most sentences have alternate markups by different annotators; the inter-annotator agreement is rather low (43.6% overall), probably due to different intended correct translations.

Since each word of a hypothesis can have several flags (e.g. `form` and `ows`, indicating a wrong surface form of a correct lemma that is also locally misplaced) we simplify the annotation by grouping the flags into four independent categories: wrong hypothesis words (lexical errors), missing reference words, misplaced words (order errors) and punctuation errors; at most one error flag from each category is allowed.

Half of the hypothesis and source translations were aligned manually by fixing Addicter's alignments; the annotators restricted themselves to one-to-one alignments whenever possible.

3.2. Alignments

In addition to the built-in lightweight alignment component other alignment methods are tested; all of these were applied to lemmatized texts:

- **Alignment from METEOR** (Banerjee and Lavie, 2005), adapted to Czech;
- **Bilingual aligners**, trained on and applied directly to the four hypothesis and reference translations: the Berkeley aligner (Liang et al., 2006) and GIZA++

Alignment Method	Alignment			Translation Errors		
	Prec.	Rec.	AER	Prec.	Rec.	F-score
addicter&via_source	86.39	85.89	**13.86**	15.27	54.06	**23.82**
addicter	98.89	72.18	**16.55**	10.36	43.76	16.75
addicter&meteor	97.90	71.54	**17.33**	10.38	43.78	16.78
addicter&giza++$_{\text{intersect}}$	85.99	77.78	18.32	13.47	49.61	21.18
addicter&berkeley&via_source	73.67	83.50	21.72	16.91	54.39	**25.80**
addicter&berkeley	71.23	78.31	25.40	15.38	52.02	**23.74**
addicter&giza++$_{\text{grow-diag}}$	65.93	74.58	30.01	14.71	48.56	22.58
via_source	85.00	74.60	20.54	13.80	54.90	22.06
giza++$_{\text{intersect}}$	81.65	64.09	28.19	11.82	48.11	18.97
berkeley*	68.12	74.38	28.89	15.16	51.56	23.43
meteor	90.37	55.04	31.59	6.08	28.68	10.04
giza++$_{\text{grow-diag}}$*	61.54	69.95	34.52	14.50	47.99	22.27

Table 1. Different alignments by their error rate (AER) and their effect on translation error detection scores; asterisk (*) marks alignments with enforced injectivity. Manual alignments are based on the output of Addicter so not comparable to others.

(Och and Ney, 2003) (intersection or diagonal-growing heuristic for symmetrical alignments);

- **Alignment via the source text**, as described in the test corpus browser section: the reference and hypothesis are independently aligned to the source by combining them with a large bilingual corpus (we used CzEng (Bojar and Žabokrtský, 2009)) and GIZA++, and the reference-hypothesis monolingual alignment is then obtained by intersecting the two bilingual alignments.

Some of these aligners produce alignments with many-to-one correspondences. Injectivity was enforced upon them using Addicter's aligner by substituting the alignment in question for the aligner's search space, originally consisting only of tokens with the same lemmas. The result is the optimal injective subset of the alignment.

In a similar way alignments were combined: the search space of Addicter's aligner was replaced with all alignment pairs, suggested by any aligners. To reward alignment points that are suggested by more than one alignment method, their emission probability is set to be proportional to the number of alignments that had them.

3.3. Results

Table 1 presents the alignment error rates of different alignment methods and their effect on the quality of detecting translation errors. Addicter's alignment method has an advantage in the evaluation of AER so we list it separately.

The most important observation is that alignment quality does not correlate with translation error detection quality: the best alignment, which is a combination of three

Wrong hypothesis word				Misplaced word			
Flag	Prec.	Rec.	F-score	Flag	Prec.	Rec.	F-score
extra	19.24	64.68	29.65	ows	14.42	48.88	22.27
unk	13.39	12.98	13.18	owl	2.47	47.69	4.70
form	38.16	40.62	39.36	ops	0.00	0.00	0.00
lex/disam	18.48	75.91	29.72	opl	0.00	0.00	0.00
Missing reference word				**Punctuation error**			
miss_c	2.17	15.28	3.80	punct	29.75	81.65	43.61
miss_a	4.78	27.23	8.14				

Table 2. Evaluation results, based on the combination of Addicter's aligner, Berkeley aligner and alignment via source with GIZA++: precision, recall and F-score of every error flag inside its corresponding group.

separate methods, has by far the highest error detection F-score (25.80), but a rather high AER (21.72). Together with the fact that the best alignment quality is mostly shown by Addicter and its combinations with other methods, this rather indicates that injective alignments do not suit the error detection task too well; it is thus essential to test translation error detection on the level of phrases or syntactic structures.

Unfortunately even the best scores of translation error detection are rather low; detailed scores of the best alignment method are given per error code in Table 2. Addicter clearly tends to overkill with almost all error types, leading to relatively high recalls and (sometimes very) low precisions. Precisions of missing and extra words are especially low; obviously these are most commonly assigned superfluously.

4. Related Work

Part of the Failfinder project[2] implemented visualization of mismatches of up to two systems compared to the reference translation. Apart from that, probably the only implemented and published toolkit with the same goal is Meteor-xRay[3] (Denkowski and Lavie, 2010). Neither of these approaches tries to classify errors as we do.

A number of software packages addresses translation evaluation in one way or another. Two recent examples include iBLEU[4], which allows the developer to inspect the test corpus and the BLEU scores of individual sentences, and Blast (Stymne, 2011), a framework for manual labelling of translation errors.

Concerning translation error analysis, Popović and Burchardt (2011) describe a language-independent method, tested on Arabic-English and German-English trans-

[2]Done at the MT Marathon 2010, Dublin; http://code.google.com/p/failfinder/.

[3]http://www.cs.cmu.edu/~alavie/METEOR/

[4]http://code.google.com/p/ibleu/

lations. Although their method shows high correlation with human judgements, their taxonomy and analysis are much less fine-grained than ours. Some recently suggested metrics include explicit modeling of specific error types or groups in them, like LRscore (Birch and Osborne, 2011) and the ATEC metric (Wong and Kit, 2010). Other attempts of decomposing metrics to get some insight into the translation system performance have been made as well (Popović and Ney, 2007). Popović et al. (2006) made a direct attempt at automatically analyzing translation errors using morpho-syntactic information, but their work only focused on specific verb-related errors.

Giménez and Màrquez (2008) report an interesting idea where a large pool of the single-outcome metrics can be used to obtain a refined picture of error types the evaluated systems make.

5. Conclusions

The open source toolkit Addicter was introduced; it implements monolingual alignment, automatic translation error analysis, browsing test and training corpora and viewing alignments. The tools are mostly independent of the translation method and language pair; no in-depth analysis of the SMT system itself is offered, but it assists the developer in searching for the reasons behind the translation errors.

Addicter includes a component for automatic detection and labelling of translation errors. Our experiments show that despite reasonable quality of the used alignments the translation error precisions are rather low, unlike relatively high recalls.

Future work on Addicter includes fully supporting lemmatization to increase applicability to highly inflectional languages, improving the translation error analysis performance and further testing the toolkit. New datasets with tagged translation errors for other language pairs and user studies are a necessity for further development. An important development direction is phrase- or structure-based error analysis.

We believe some kind of automated error analysis will soon become an inherent step in MT system development and that future development will increase the match with human annotation.

Acknowledgements

This work was supported by the Czech Science Foundation grants P406/11/1499 and P406/10/P259 and the Estonian Center of Excellence in Computer Science.

Bibliography

Banerjee, Satanjeev and Alon Lavie. METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In *Proc. of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, pages 65–72, Ann Arbor, Michigan, 2005.

Birch, Alexandra and Miles Osborne. Reordering metrics for MT. In *Proc. of ACL'11*, pages 1027–1035, Portland, Oregon, USA, 2011.

Birch, Alexandra, Miles Osborne, and Phil Blunsom. Metrics for MT evaluation: Evaluating reordering. *Machine Translation*, 24(1):15–26, 2010.

Bojar, Ondřej. Analyzing Error Types in English-Czech Machine Translation. *Prague Bulletin of Mathematical Linguistics*, 95, 2011.

Bojar, Ondřej and Zdeněk Žabokrtský. CzEng 0.9: Large parallel treebank with rich annotation. *Prague Bulletin of Mathematical Linguistics*, 92:63–83, 2009.

Denkowski, Michael and Alon Lavie. Extending the METEOR Machine Translation Evaluation Metric to the Phrase Level. In *Proc. of HLT-NAACL'10*, pages 250–253, 2010.

Giménez, Jesús and Lluis Màrquez. Towards heterogeneous automatic MT error analysis. In *Proc. of LREC'08*, Marrakech, Morocco, 2008.

Liang, Percy, Ben Taskar, and Dan Klein. Alignment by agreement. In *Proc. of HLT-NAACL'06*, pages 104–111, New York City, USA, 2006.

Och, Franz Josef and Hermann Ney. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51, 2003.

Popović, Maja and Hermann Ney. Word error rates: decomposition over POS classes and applications for error analysis. In *Proc. of WMT'07*, pages 48–55, Stroudsburg, PA, USA, 2007.

Popović, Maja, Adrià de Gispert, Deepa Gupta, Patrik Lambert, Hermann Ney, José B. Mariño, Marcello Federico, and Rafael Banchs. Morpho-syntactic information for automatic error analysis of statistical machine translation output. In *Proc. of WMT'06*, pages 1–6, New York, USA, 2006.

Popović, Maja and Aljoscha Burchardt. From human to automatic error classification for machine translation output. In *Proc. of EAMT'11*, pages 265–272, Leuven, Belgium, 2011.

Stymne, Sara. Blast: A tool for error analysis of machine translation output. In *Proc. of ACL-HLT'11*, pages 56–61, Portland, Oregon, 2011.

Vilar, David, Jia Xu, Luis Fernando D'Haro, and Hermann Ney. Error analysis of machine translation output. In *Proc. of LREC'06*, pages 697–702, Genoa, Italy, 2006.

Vogel, Stephan, Hermann Ney, and Christoph Tillmann. HMM-based word alignment in statistical translation. In *Proc. of COLING'96*, pages 836–841, Copenhagen, Denmark, 1996.

Wong, Billy and Chunyu Kit. The parameter-optimized ATEC metric for MT evaluation. In *Proc. of the Joint WMT'10 and MetricsMATR*, pages 360–364, Uppsala, Sweden, 2010.

Address for correspondence:
Daniel Zeman
zeman@ufal.mff.cuni.cz
Charles University in Prague, Faculty of Mathematics and Physics
Institute of Formal and Applied Linguistics
Malostranské náměstí 25
CZ-11800 Praha 1, Czech Republic

PBML

The Prague Bulletin of Mathematical Linguistics
NUMBER 96 OCTOBER 2011 89–98

eppex: Epochal Phrase Table Extraction for Statistical Machine Translation

Česlav Przywara, Ondřej Bojar

Charles University in Prague
Faculty of Mathematics and Physics
Institute of Formal and Applied Linguistics

Abstract

We present a tool that extracts phrase pairs from a word-aligned parallel corpus and filters them on the fly based on a user-defined frequency threshold. The bulk of phrase pairs to be scored is much reduced, making the whole phrase table construction process faster with no significant harm to the ultimate phrase table quality as measured by BLEU. Technically, our tool is an alternative to the *extract* component of the *phrase-extract* toolkit bundled with Moses SMT software and covers some of the functionality of *sigfilter*.

1. Motivation

Phrase tables in Statistical Machine Translation (SMT) systems generally take the form of a list of pairs of phrases s and t, s being the phrase from the source language and t being the phrase from the target language, along with scores that should reflect the goodness of translating s as t. The standard approach to obtain such scores is to use *maximum likelihood probability* of the phrase t given the phrase s and vice versa. The probabilities $p(s|t)$ and $p(t|s)$ are often referred to as *forward* and *reverse translation probabilities*.

To estimate $p(s|t)$ and $p(t|s)$, frequency counts $C(t,s)$, $C(s)$ and $C(t)$ are usually collected from the entire training corpus. For substantial coverage of source and target languages, such corpora are often very big so all phrase pairs and their counts cannot fit in the physical memory of the computer. To overcome this limitation, phrase table construction methods often simply dump observed phrases to local disk and sort and

© 2011 PBML. All rights reserved.
Corresponding author: bojar@ufal.mff.cuni.cz
Cite as: Česlav Przywara, Ondřej Bojar. eppex: Epochal Phrase Table Extraction for Statistical Machine Translation. The Prague Bulletin of Mathematical Linguistics No. 96, 2011, pp. 89–98.
doi: 10.2478/v10108-011-0014-1.

count them on disk. This approach allows to construct phrase tables of size limited only by the capacity of the disk. The obvious drawback of this solution is that much more time is needed to build the table.

Moses (Koehn et al., 2007), an open-source SMT toolkit with a full set of tools required for SMT system training, adheres to this concept. Both of the two main components used to construct phrase tables (*extract* to observe phrases and *scorer* to score them) treat their input as an unbounded stream of data, keeping only limited span of this stream in physical memory and using local disk for temporary storage.

It is known that phrase table quality is not strictly determined by its size. Johnson et al. (2007) presented a method for the reduction of phrase table size causing no harm to translation quality as measured by BLEU (Papineni et al., 2002). Their method employs significance testing of phrase pair co-occurrence in the parallel corpus to distinguish between valuable phrase pairs and random noise. Because significance testing of phrase pair co-occurrences is based on their frequencies, the method was designed as a post-processing filter applied to a finished phrase table. The phrase table extraction process and its runtime requirements are unaffected by this method.

In this paper, we present *eppex*, a tool designed as a drop-in alternative for *extract* component in Moses training. Like *extract*, our tool extracts phrase pairs from a word-aligned parallel corpus. Unlike *extract*, *eppex* filters out phrase pairs with frequency below a user-defined threshold. As a result, the subsequent sorting and scoring have to process a reduced set of phrase pairs, so the whole phrase table extraction pipeline requires less time to finish.

In the rest of the paper we present the implementation details and results of the experiments aiming at comparison of the standard approach, the standard approach with additional significance filtering and the epochal extraction with respect to translation quality and runtime performance.

2. Implementation

Our tool, just like the *extract* component, processes the input parallel corpus in a single pass. Our implementation reuses the code from *extract* that implements the extraction of individual phrase pairs from word-aligned parallel corpora as proposed by Och and Ney (2003). In contrast to *extract*, extracted phrase pairs are not immediately printed to a temporary storage on the disk, but instead they are fed into an algorithm that on the fly filters out low frequency items.

To carry out the filtration within manageable memory demands, we employ an algorithm for approximate frequency counting proposed by Manku and Motwani (2002). Their *Lossy Counting* algorithm expects two user-defined thresholds: *support* $s \in (0,1)$ and *error* $\epsilon \in (0,1)$, such that $\epsilon \ll s$. At any point of time (after being fed with N items) the algorithm can output the list of items with their approximate frequencies and guarantee the following:

- All items whose true frequency exceeds sN are output (*no false negatives*).

- No item whose true frequency is less than $(s - \epsilon)N$ is output (*few false positives*).
- Estimated frequencies are less than the true frequencies by at most ϵN.
- The space used by the algorithm is $O(\frac{1}{\epsilon} \log \epsilon N)$.

2.1. Lossy Counting Algorithm

The Lossy Counting algorithm conceptually divides the incoming stream of items into epochs of fixed size $w = \lceil \frac{1}{\epsilon} \rceil$ (thus the name *epochal extraction*). In order to deliver the frequency estimates, the algorithm maintains a data structure D consisting of triples (e, f, Δ), where e is an element from the stream, f is its estimated frequency and Δ is the maximum possible error in f. When a new item e arrives, a lookup for e in D is performed. If e is already present, its frequency f is incremented by one. Otherwise a new triple $(e, 1, T - 1)$ is added to D, where T denotes the ID of current epoch (with IDs starting from 1).

At the end of each epoch (determined by $N \equiv 0 \bmod w$), the algorithm prunes off all items whose maximum true frequency is small. Formally, at the end of the epoch T, all triples satisfying the condition $f + \Delta \leqslant T$ are removed from D. When all elements in the stream have been processed, the algorithm returns all triples (e, f, Δ) where $f \geq (s - \epsilon)N$.

The idea behind the algorithm is that frequent elements show up more than once within each epoch so their frequencies are increased enough to survive the filtering.

2.2. Memory Management

To optimize the usage and access speed of the memory, we implement a couple of tricks. First, we explicitly store the vocabulary of the phrases read so far. The phrases are then represented as vectors of word indices instead of full strings. (The vocabulary is not subject to pruning at epoch boundaries for efficiency reasons.)

Second, we take advantage of the fact the vocabulary size of MT corpora usually is on the order of millions. We therefore represent words as 4-byte integers allowing to store up to 4 billions of word types. Using directly the pointer to the string representation of the word would be more expensive in a 64-bit environment.

Third, we optimize the process of memory allocation for newly created word types by using memory pools as implemented in the *Boost* library.

2.3. Usage

Eppex is implemented as C++ program and does not require any special libraries to compile and run, except for moderately recent version of *gcc*. The authors did not attempt to compile *eppex* on Windows, but the code is free of system-dependent hacks, so porting to Windows should be fairly straightforward.

Eppex input and output format is fully compatible with that of *extract*. We also keep the command line syntax very similar. The main change is that instead of the *max-*

phrase-length parameter, one has to specify the ϵ and s values for the Lossy Counting algorithm. A different pair of thresholds can be given for each phrase pair length[1] allowing for a more fine-grained pruning.

The command line syntax is:

```
eppex tgt src align extract lossy-counter [lossy-counter-2 [...]] \
  [orientation [ --model [wbe|phrase|hier]-[msd|mslr|mono] ]]
```

Every `lossy-counter` specifies the *error* and *support* for the Lossy Counting algorithm for phrases of a length within an interval. The parameter takes the form *length:error:support*, where *length* is either a number or an interval specification and *error* and *support* are two floats. For example, to keep in all phrase pairs with length 1 and prune all phrase pairs of length from 2 to 4 with $\epsilon = 2 \times 10^{-7}$ and $s = 8 \times 10^{-7}$, two lossy counters must be declared as: `1:0:0 2-4:2e-7:8e-7`.[2]

Phrases of length not covered by any `lossy-counter` are not extracted at all, effectively setting also the *max-phrase-length*.

No defaults for `lossy-counters` are provided, because reasonable settings heavily depend on the corpus. *Eppex* at its end and also a faster single-purpose tool *counter* report the total number of extracted phrases of all lengths, allowing to set the thresholds.

3. Experiments

We compared our tool against two other methods of phrase table construction that were already introduced in the beginning of the paper:

1. the *extract* component of Moses toolkit, i.e. the baseline,
2. the *sigfilter* program, which is a re-implementation of significance testing phrase table filter described by Johnson et al. (2007) and is also bundled with Moses.

The baseline scenario has no options to adjust: all phrase pairs extracted from the corpus are included in the final phrase table. When applying *sigfilter*, at least one of two options has to be set: the *cutoff threshold* or the *pruning threshold*. By setting the *cutoff threshold* to n, all but the top n phrase pairs, sorted by the forward probability $P(t|s)$, will be removed. Johnson et al. (2007) recommend the cutoff of 30. The *pruning threshold* determines the minimum level of negative-log-p-value that a particular phrase pair (s, t) has to reach under Fisher's exact test that calculates probability of observed two by two contingency table based on frequency counts $C(s)$, $C(t)$ and $C(s, t)$. A particularly interesting settings for this threshold are values $\alpha - \epsilon$ and $\alpha + \epsilon$, where $\alpha = \log(N)$ and ϵ is appropriately small positive number. The former is the largest

[1] Phrase pair length is defined as the length of the longer of the phrases.

[2] All phrases of length 2–4 are stored together in one counter. To treat them separately, the counter has to be split in three: `2:2e-7:8e-7 3:2e-7:8e-7 4:2e-7:8e-7`.

threshold that results in keeping all the 1-1-1 phrase pairs[3] in the table, whereas the latter is the smallest threshold that results in all such phrase pairs being removed.

With *eppex*, a separate lossy counting may be instantiated for each phrase pair length, allowing to filter them with different values of sN (positive) and $(s - \epsilon)N$ (negative) thresholds. Depending on the corpus, the values of *support* and *error* have to be adjusted. The effect of *sigfilter* with $\alpha \pm \epsilon$ pruning may be approximated: setting *support* s such that $sN < 1$ will preserve all of the 1-1-1 phrases (like in $\alpha - \epsilon$), while setting *support* s and *error* ϵ to satisfy $(s - \epsilon)N > 1$ will result in their complete removal (like in $\alpha + \epsilon$).

3.1. Data

We evaluate the extraction methods on English-Czech translation trained on the corpus CzEng (Bojar and Žabokrtský, 2009) with a few additions and a large Czech language model. See Mareček et al. (2011) for the exact setup of the system "cu-bojar".

The complete parallel corpus for our experiments with phrase extraction is 8.4M sentence pairs; 107.2M English and 93.2M Czech tokens.

Our tuning and testing data come from the WMT 2011 Translation Task[4].

3.2. Benchmarking

The training process of Moses SMT takes place in nine steps.[5] The steps cover the whole training pipeline including word alignment, lexical table construction, phrase table construction and more. The phrase table construction itself is done in two steps, phrase extraction and phrase scoring, which might be even further split into following substeps: (1) phrase extraction that produces direct and reverse phrase table halves (without scores yet); (2) gzipping, (3) sorting and (4) scoring of the direct table; (5) gzipping, (6) sorting and (7) scoring of the reverse table; (8) sorting of the scored reverse table; (9) consolidation of the scored direct and reverse tables; (10) gzipping of the consolidated phrase table.

The default implementation in Moses training script *train-model.perl* does not use any parallelization of the sequence (steps 2–4 and 5–7 could be run in parallel). The gzipping in steps 2 and 5 is somewhat dubious but everybody seems to use it.[6]

We benchmark phrase table construction by measuring CPU time, wall clock time and memory requirements of all the substeps. To measure the memory with a reasonable precision, we save a copy of *stat* and *status* files from */proc/[pid]/* directory of the measured process(es) every second.

[3] A phrase pair (s,t) is called 1-1-1, if $C(s) = 1$, $C(t) = 1$ and $C(s,t) = 1$.

[4] `http://www.statmt.org/wmt11/translation-task.html`

[5] `http://www.statmt.org/moses/?n=FactoredTraining.HomePage`

[6] We discovered that the option *–dont-zip* of *train-moses.perl* has been broken since it was introduced.

We run all the steps on a single machine for comparability of the results. Although the machine is a standard node in a cluster, we keep jobs of other users away by reserving all memory of the machine for our job. The machine runs the 64-bit version of Ubuntu 10.04 server edition on 2 Core4 AMD Opteron 2.8 Ghz processors with 32 GB of RAM in total. All the input and output files were read and written to a locally mounted hard disk.

4. Results

Table 1 presents all the experiments and their settings. We compare the baseline, the recommended default settings for significance filtering and two *eppex* runs: one with mild pruning that left in all shorter phrase pairs (denoted as *eppex 1-in*) and one harsher that filters out all phrase pairs with single occurrence only (*eppex 1-out*).

Name	Description
baseline	standard Moses pipeline with *extract* component
eppex 1-in	the pipeline with *eppex*: all phrase pairs of length 1–3 kept in, longer phrase pairs pruned with max. positive threshold of 8
eppex 1-out	the pipeline with *eppex*: all single-occurring phrase pairs removed, phrase pairs pruned with max. positive threshold of 8
sigfilter a-e	baseline followed by *sigfilter* with pruning threshold $\alpha - \epsilon$
sigfilter a+e	baseline followed by *sigfilter* with pruning threshold $\alpha + \epsilon$
sigfilter 30	baseline followed by *sigfilter* with cutoff threshold of 30

Table 1. List of the experiments and their settings

4.1. Translation quality

We evaluate translation quality automatically using BLEU. The complete SMT system includes also a language model (always identical) and a distortion model. For *eppex* setups, the distortion model was always built from the same (pruned) set of phrase pairs as the phrase table. In each setup separately, model weights are optimized using Moses MERT.

Table 2 presents BLEU scores obtained in the experiments and the respective phrase table sizes. For both of the test sets, the top three results were obtained in *baseline*, *sigfilter 30* and *eppex 1-in* experiments, but all the differences are rather small and could be also attributed to the randomness of MERT.[7] The *baseline* scenario ranked best on wmt11 set (BLEU score 18.22), while the *eppex 1-in* scenario topped on wmt10 set

[7] We did not invest the computing resources necessary to estimate the confidence bounds covering optimizer instability (Clark et al., 2011).

	Final phrase table size		BLEU score	
Experiment	phrase pairs	.gz file size	wmt10	wmt11
baseline	153.6 M	3.68 GB	17.36	**18.22**
sigfilter 30	137.0 M	3.36 GB	17.48	18.13
sigfilter a-e	92.4 M	2.39 GB	17.23	17.87
eppex 1-in	57.1 M	1.28 GB	**17.60**	18.10
sigfilter a+e	35.0 M	0.86 GB	17.31	17.99
eppex 1-out	14.4 M	0.33 GB	17.23	17.94

Table 2. Phrase table sizes and BLEU scores for all experiments

(BLEU score 17.60). However, the phrase table extracted in *eppex 1-in* occupied only 1.28 GB space on disk, being less than half of the size of the *baseline* (3.68 GB) and *sigfilter 30* (3.36 GB) phrase tables.

Harsher pruning in both *eppex* and *sigfilter* can reduce the phrase table size up to one tenth of the baseline with only negligible loss in BLEU.

4.2. Memory and time requirements

Table 3 presents in detail physical memory peaks for all experiments. In *eppex* scenarios, it is the epochal extraction itself that is the most demanding part of the pipeline, consuming 19.2 GB (*1-in*) and 16.7 GB (*1-out*) of memory. In all the other experiments the memory consumption is considerably lower and the *scorer* component is responsible for the peak, except for the cases when $\alpha \pm \epsilon$ significance filtering is applied.

Experiment	VM peak	in step
baseline	1.1 GB	scoring-e2f
sigfilter 30	1.1 GB	scoring-e2f
sigfilter a-e	5.4 GB	sigfilter
eppex 1-in	19.2 GB	phr-ext
sigfilter a+e	5.4 GB	sigfilter
eppex 1-out	16.7 GB	phr-ext

Table 3. Virtual memory peaks for all experiments

Table 4 compares CPU usage and wallclock times of all the substeps of the pipeline in *baseline* and *eppex* scenarios. While the initial extraction of phrase pairs takes much longer with *eppex* than with *extract*, subsequent steps finish much quicker in the *eppex* scenario: total time required is less than half in case of *1-in* pruning and less than 1/4

	baseline		eppex 1-in		eppex 1-out	
step	CPU	wallclock	CPU	wallclock	CPU	wallclock
phr-ext	1145	**1152**	4346	**4360**	4349	**4361**
gzip-f2e	590	662	183	226	86	114
gzip-e2f	593	641	185	276	87	132
sort-f2e	653	2257	304	822	196	564
sort-e2f	628	2844	315	810	199	567
score-f2e	10516	10795	4493	4531	371	372
score-e2f	9400	9622	2867	2902	340	340
sort-inv	358	1569	129	129	21	22
cons	754	1361	269	269	65	66
pt-gzip	791	881	258	259	65	65
TOTAL	25428	31784	13349	14584	5779	6603
hh:mm:ss	7:03:48	**8:49:44**	3:42:29	**4:03:04**	1:36:19	**1:50:03**

Table 4. CPU and wallclock times (in seconds) of the phrase table construction.

Sigf. settings	-l a+e		-l a-e		-n 30	
	CPU	wallclock	CPU	wallclock	CPU	wallclock
Sigfilter alone	18635	18248	19252	18449	2105	1141
TOTAL	44063	50032	44680	50233	27533	31784
hh:mm:ss	12:14:23	13:53:52	12:24:40	13:57:13	7:38:53	9:08:45

Table 5. CPU and wallclock times (in seconds) of significance filtering. The total includes the time of baseline extraction: 7:03:48 (CPU) and 8:49:44 (wallclock).

in case of *1-out* pruning compared to the baseline. Partial parallelization of the baseline extraction as suggested above decreases the gains but *eppex* still remains safely faster, esp. if *eppex* used the same optimization.

Significance filtering requires an additional amount of time (see Table 5) on top of the baseline extraction. The most striking difference is between *sigfilter* $\alpha+\epsilon$ and *eppex 1-out*. They are comparable in terms of BLEU score and phrase table size but *sigfilter* took almost 14 hours while *eppex 1-out* finished in less than 2 hours.

5. Related and Future Work

We point out that the idea of using approximate frequency counting algorithms in the field of NLP is not new. Goyal et al. (2009) used approximate n-gram frequency counts to build language models, which they then applied successfully in SMT achieving no significant loss in BLEU.

A recent feature of Moses is *incremental training* (probably related to the experiments by Levenberg et al. (2010)) aiming at the reduction of time required to incorporate recent data into already trained models. The entire source and target corpora are indexed in suffix arrays along with their alignments. Phrase extraction and scoring happens on the fly when phrases are needed in translation, completely eliminating the expensive batch retraining. Because the reduction of training time is the main advantage of *eppex*, we intend to perform a comprehensive comparison of this feature with batch retraining utilizing *eppex*.

The epochal extraction in *eppex* also lends itself to incremental extraction: only the counts in the current epoch have to be stored and reloaded when the model is to be extended by new data. The setting of the thresholds, however, would require a fairly large amount of training data to be processed in the first batch to estimate the values that will lead to sufficient saturation of phrase table. Our initial experiment suggests that for short phrase pair lengths it is beneficient to use no pruning at all.

Hardmeier (2010) presented *memscore*, an open-source tool to score phrases in memory that acts as a faster drop-in replacement for the sorting and the *scorer* in the pipeline. By combining *memscore* and *eppex* into a single phrase extraction tool a further speed up of phrase table construction process might be achieved.

Furthermore, we expect the benefits of *eppex* to be even more significant when confronted with larger training corpora, therefore we are in the process of its evaluation on the 10^9 French–English corpus available as part of WMT training data.

6. Conclusions

We presented *eppex*, a tool for extraction of phrase pairs from word-aligned parallel corpus capable of phrase pairs filtering on the fly based on a user-defined threshold. *Eppex* is a drop-in alternative of *extract* component in Moses training toolkit. Our tool is ready to use and it is available in Moses SVN trunk (in `scripts/training/eppex`).

We compared our tool against the baseline extraction and another common approach to phrase table filtration. By using *eppex* for phrase extraction we were able to obtain translation quality comparable to the baseline, while at the same time both the (wallclock) training time and phrase table size have been reduced by more than a half or up to one tenth with harsher pruning. Although memory requirements are significantly increased, they still lie within manageable limits.

7. Acknowledgment

This work has been supported by the grants EuroMatrixPlus (FP7-ICT-2007-3-231720 of the EU and 7E09003 of the Czech Republic), P406/10/P259, and MSM 0021620838.

Bibliography

Bojar, Ondřej and Zdeněk Žabokrtský. CzEng0.9: Large Parallel Treebank with Rich Annotation. *Prague Bulletin of Mathematical Linguistics*, 92:63–83, 2009. ISSN 0032-6585.

Clark, Jonathan H., Chris Dyer, Alon Lavie, and Noah A. Smith. Better hypothesis testing for statistical machine translation: Controlling for optimizer instability. In *Proc. of ACL/HLT*, pages 176–181, Portland, Oregon, USA, June 2011. URL http://www.aclweb.org/anthology/P11-2031.

Goyal, Amit, Hal Daumé, III, and Suresh Venkatasubramanian. Streaming for large scale NLP: language modeling. In *Proc. of HTL/NAACL*, pages 512–520, Boulder, Colorado, 2009. URL http://portal.acm.org/citation.cfm?id=1620754.1620829.

Hardmeier, Christian. Fast and Extensible Phrase Scoring for Statistical Machine Translation. *The Prague Bulletin of Mathematical Linguistics*, 93:79–88, 2010.

Johnson, J Howard, Joel Martin, George Foster, and Roland Kuhn. Improving Translation Quality by Discarding Most of the Phrasetable. In *Proc. of EMNLP and Computational Natural Language Learning*, pages 967–975, 2007.

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 *Proc. of ACL (Demonstration Session)*, pages 177–180, 2007.

Levenberg, Abby, Chris Callison-Burch, and Miles Osborne. Stream-based translation models for statistical machine translation. In *Proc. of HTL/NAACL*, pages 394–402, Los Angeles, California, 2010. URL http://portal.acm.org/citation.cfm?id=1857999.1858061.

Manku, Gurmeet Singh and Rajeev Motwani. Approximate Frequency Counts over Data Streams. In *Proceedings of the 28th International Conference on Very Large Data Bases*, 2002.

Mareček, David, Rudolf Rosa, Petra Galuščáková, and Ondřej Bojar. Two-step translation with grammatical post-processing. In *Proc. of WMT*, Edinburgh, UK, July 2011.

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, Philadelphia, Pennsylvania, 2002. URL http://dx.doi.org/10.3115/1073083.1073135.

Address for correspondence:
Ondřej Bojar
bojar@ufal.mff.cuni.cz
UK MFF ÚFAL
Malostranské náměstí 25
118 00 Praha 1, Czech Republic

PBML

The Prague Bulletin of Mathematical Linguistics
NUMBER 96 OCTOBER 2011 99–108

Multi-Task Minimum Error Rate Training for SMT

Patrick Simianer, Katharina Wäschle, Stefan Riezler

Department of Computational Linguistics, University of Heidelberg, Germany

Abstract

We present experiments on multi-task learning for discriminative training in statistical machine translation (SMT), extending standard minimum-error-rate training (MERT) by techniques that take advantage of the similarity of related tasks. We apply our techniques to German-to-English translation of patents from 8 tasks according to the International Patent Classification (IPC) system. Our experiments show statistically significant gains over task-specific training by techniques that model commonalities through shared parameters. However, more fine-grained combinations of shared parameters with task-specific ones could not be brought to bear on models with a small number of dense features. The software used in the experiments is released as open-source tool.

1. Introduction

Multi-task learning aims at learning several different tasks simultaneously, addressing commonalities through shared parameters and modeling differences through task-specific parameters. This learning framework is advantageous if the tasks are not completely independent of each other, which would advocate to train a separate model for each task. Instead, they should be related and share some commonalities, yet be different enough to counter a simple pooling of training data.

A predestined application for multi-task learning in the area of statistical machine translation (SMT) is patent translation over several different classes of patents according to the International Patent Classification (IPC)[1]. Table 1 shows the eight top level

[1] http://www.wipo.int/classifications/ipc/en/

© 2011 PBML. All rights reserved.
Corresponding author: riezler@cl.uni-heidelberg.de
Cite as: Patrick Simianer, Katharina Wäschle, Stefan Riezler. Multi-Task Minimum Error Rate Training for SMT. The Prague Bulletin of Mathematical Linguistics No. 96, 2011, pp. 99–108.
doi: 10.2478/v10108-011-0015-0.

sections of the IPC categorization. They aim at a distinction of main technological fields, each of which is characterized by its own technological terminology.

A Human Necessities
B Performing Operations; Transporting
C Chemistry; Metallurgy
D Textiles; Paper
E Fixed Constructions
F Mechanical Engineering; Lighting; Heating; Weapons; Blasting
G Physics
H Electricity

Table 1. Patent sections according to the IPC classification.

On the other hand, patents exhibit strong commonalities across IPC sections in sharing a highly specialized vocabulary, consisting of a legal jargon not found in everyday language, and a rigid textual structure including highly formulaic language. The goal of multi-task learning for SMT is thus to learn a translation system that performs well across several different patent sections, thus benefits from shared information, and yet is able to address the specifics of each patent section.

The machine learning community has developed several different formalizations of the central idea of trading off optimality of parameter vectors for each task-specific model and closeness of these model parameters to the average parameter vector across models. For example, Evgeniou and Pontil (2004) develop this idea in the framework of support vector machines (SVM) as finding a tradeoff between each task-specific SVM having a large margin and having each SVM close to the average SVM. They formalize this tradeoff via regularization of the task-specific parameter vectors and of the distance to the average parameter vector. The starting point of all this and related algorithms is a linear classifier (or a non-linear kernelized variant) with a fixed feature vector (or kernel) whose associated parameters are adjusted in multi-task learning.

(Multi-)domain adaptation[2] for SMT has so far been seen as a challenge of out-of-vocabulary (OOV) terms. Adaptation techniques thus have focused on gathering OOV information from various sources in order to feed the standard generative SMT pipeline of translation and language model with it. A recent approach is Daumé and Jagarlamudi (2011) who mine translations for OOV terms from comparable corpora.

Patent translation exhibits an even more severe OOV problem because of very specialized terminology in different IPC patent sections. Multi-task learning or domain

[2]We consider domain adaptation as a special case of multi-task learning for two tasks, and multi-domain adaptation as equivalent to multi-task learning. Other definitions are possible (see, e.g., Dredze et al. (2010)).

adaptation efforts in patent translation have so far been restricted to experimental combinations of translation and language models from different sets of IPC sections (Utiyama and Isahara, 2007; Tinsley et al., 2010; Ceauşu et al., 2011).

In this paper, we consider the specific setting in which the generative SMT pipeline is not adaptable. Such situations arise if there are not enough parallel data to train generative models on the new tasks. However, we assume that there are enough parallel data available to perform discriminative training (Och, 2003) for each specific task. Our goal is to investigate how state-of-the-art multi-task learning techniques for linear classifiers can be applied to standard discriminative training for SMT. In other words, we would like to know how much gain there is in extending the standard tuning technique of minimum error rate training (MERT) to multi-task MERT for SMT. To this aim, we present a generic new algorithm to model commonalities by regularized parameter averaging, building upon Evgeniou and Pontil (2004), and apply it to multi-task MERT for SMT . Furthermore, we present a distributed implementation of MERT for multiple tasks that allows us to apply techniques for parameter averaging from distributed learning (Zinkevich et al., 2010) to a version of averaged MERT. Our experimental results show that averaged and multi-task MERT achieve statistically significant gains over training separate task-specific models. However, multi-task MERT's fine-grained combination of shared parameters with task-specific ones did not improve upon parameter averaging in our experiments on models with a small number of dense features.

2. Related Work

A central idea to learn common behaviors across related task is to learn task-specific models and to minimize their deviation from an average model. Starting from a separate SVM for each task, Evgeniou and Pontil (2004) present a regularization method that trades off optimization of the task-specific parameter vectors and the distance of each SVM to the average SVM. Equivalent formalizations replace parameter regularization by Bayesian prior distributions on the parameters (Finkel and Manning, 2009) or by augmentation of the feature space with domain independent features (Daumé, 2007). Besides SVMs, several learning algorithms have been extended to the multi-task scenario in a parameter regularization setting, e.g., perceptron-type algorithms (Dredze et al., 2010) or boosting (Chapelle et al., 2011). Further variants include different formalizations of norms for parameter regularization, e.g., $\ell_{1,2}$ regularization (Obozinski et al., 2010) or $\ell_{1,\infty}$ regularization (Quattoni et al., 2009), where only the features that are most important across all tasks are kept in the model.

While the standard machine learning approaches to multi-task learning are based on linear classifiers (or non-linear kernelized versions), SMT approaches to multi-task learning have concentrated on adapting unsupervised generative modules such as translation models or language models to new tasks. For example, transductive approaches have used automatic translations of monolingual corpora for self-training

modules of the generative SMT pipeline (Ueffing et al., 2007; Schwenk, 2008; Bertoldi and Federico, 2009). Other approaches have extracted parallel data from similar or comparable corpora (Zhao et al., 2004; Snover et al., 2008). Several approaches have been presented to train separate translation and language models on task-specific subsets of the data and combine them in different mixture models (Foster and Kuhn, 2007; Koehn and Schroeder, 2007).

Multi-task learning efforts in patent translation have so far been restricted to experimental combinations of translation and language models from different sets of IPC sections. For example, Utiyama and Isahara (2007) and Tinsley et al. (2010) investigate translation and language models trained on different sets of patent sections, with larger pools of parallel data improving results. Ceauşu et al. (2011) find that language models always and translation model mostly benefit from larger pools of data from different sections.[3]

3. Parallel Data from Patent Classes for Patent Translation

Our work on patent translation is based on the MAREC[4] patent data corpus. MAREC contains over 19 million patent applications and granted patents in a standardized format from four patent organizations (European Patent Office (EP), World Intellectual Property Organisation (WO), United States Patent and Trademark Office (US), Japan Patent Office (JP)), from 1976 to 2008.

Patent text is organized in 4 document sections, the patent title, abstract, description and claims. The patent title is usually a short noun phrase. The abstract contains a short summary of the invention. The description is a detailed explanation of the patent. The claims are a list of sentences that define the scope of protection granted by the patent with a standardized sentence structure. MAREC contains comparable text sections, mainly in English, French, and German. Patent titles are automatically parallel, since they only consist of one sentence and there is one title per document. Text in abstracts and claims must be split into sentences and aligned. There are no parallel descriptions.

For our experiments, we extracted bilingual abstract and claims sections from the EP and WO parts for German-to-English translation. The distribution over the sections mirrors the overall distribution of IPC sections in the corpus (see Table 2). For sentence splitting and tokenizing we used the Europarl tools[5]. Sentence alignment was done with Gargantua 1.0b[6]. The training data for the de-en language pair contains 1,000,000 sentences extracted from all top-level IPC sections from abstracts (5%)

[3] Ceauşu et al. (2011) report that including data from IPC section C (chemistry) in pooled training data is detrimental for translation models.

[4] http://www.ir-facility.org/prototypes/marec

[5] http://www.statmt.org/europarl/

[6] http://sourceforge.net/projects/gargantua/

A	266,521	21.81%
B	384,517	31.47%
C	372,903	30.52%
D	50,579	4.14%

E	54,396	4.45%
F	149,370	12.22%
G	291,671	23.87%
H	228,147	18.67%

Table 2. Distribution of IPC sections for comparable de-en abstracts and claims.

and claims (95%) from 1993 to 1995. Furthermore, we extracted for each top-level IPC section 2,000 randomly sampled sentences from abstracts (5%) and claims (95%) from 2007 (development) and 2008 (development-testing and final-testing). Table 3 gives an overview over the processed data.

	train	dev	devtest	test
# parallel sents	1M	2K	2K	2K
avg. # tokens de	32,329,745	59,376	60,061	59,930
avg. # tokens en	36,005,763	69,584	70,700	70,331
year	1993-1995	2007	2008	2008

Table 3. Statistics on parallel de-en data extracted from MAREC patent corpus.

4. Distributed Multi-Task Parameter Regularization

Multi-task learning assumes learning tasks or domains $d = 1, \ldots, D$, each coming with a separate sample of $n(d)$ training points from the same space. Evgeniou and Pontil (2004)'s idea of trading off optimal parameter weights for each task-specific model and closeness to an average parameter vector can be stated in a more general form as follows. We aim at minimization of task-specific loss functions l_d under a regularization of task-specific parameter vectors w_d towards an average parameter vector w_{avg}.

$$\min_{w_1,\ldots,w_D} \sum_{d=1}^{D} l_d(w_d) + \lambda \sum_{d=1}^{D} |w_d - w_{avg}|_p^p \quad (1)$$

For prediction, one can use task-specific weight vectors $w_d \in \{w_1, \ldots, w_D\}$ that have been adjusted to trade off task-specificity (small λ) and commonality (large λ), or the average weight vector w_{avg} as a global model.

An average or global model can be estimated directly by applying ideas from distributed learning (Zinkevich et al., 2010). The idea is to base the distribution strategy on task-specific partitions of data. An algorithm for distributed average learning will

take a loss function $c_d(w_d)$ for data and weights specific to task d, parameter initializations $w^{(0)}$, and return an averaged weight vector w_{avg}, for D tasks. An instantiation of such an algorithm to our problem, called `AvgMERT`, calls one iteration of a MERT implementation, denoted by `MERT`, that continues from parameter vector $w_d^{(t-1)}$ and optimizes translation loss $c_d(w_d)$ on the data from task d.

```
AvgMERT(w^(0), D, {c_d}_{d=1}^D):
for d = 1,...,D parallel do
  for t = 1,...,T do
    w_d^(t) = MERT(w_d^(t-1), c_d(w_d))
  end for
end for
return w_avg = (1/D) Σ_{d=1}^D w_d^(T)
```

For multi-task learning, we set p=1 to obtain an ℓ_1 regularizer, and apply the penalty term λ to the parameter weights the extent that they do not cross the average weights. That is, the weight vector w_d is moved towards the average weight vector w_{avg} by adding or subtracting the penalty λ for each weight component $w_d[k]$, and clipped when it crosses the average. This strategy can be motivated in a stochastic gradient descent framework (Tsuruoka et al., 2009), however, we apply it to regularized loss minimization in general, and to regularized MERT in specific. As stopping criterion we used a threshold on the maximal change in the average parameter vector.

```
MMERT(w^(0), D, {c_d}_{d=1}^D):
for t = 1,...,T do
  w_avg^(t) = (1/D) Σ_{d=1}^D w_d^(t-1)
  for d = 1,...,D parallel do
    w_d^(t) = MERT(w_d^(t-1), c_d(w_d))
    for k = 1,...,K do
      if w[k]_d^(t) − w_avg^(t)[k] > 0 then
        w_d^(t)[k] = max(w_avg^(t)[k], w_d^(t)[k] − λ)
      else if w_d^(t)[k] − w_avg^(t)[k] < 0 then
        w_d^(t)[k] = min(w_avg^(t)[k], w_d^(t)[k] + λ)
      end if
    end for
  end for
end for
return w_1^(T),...,w_D^(T), w_avg^(T)
```

The code described in this section is written as script wrapper around the MERT implementation of Bertoldi et al. (2009). The code is licensed unter the LGPL and can be found online[7].

5. Experiments

For training a German-to-English baseline model on the 1 million parallel patent data described in Section 3, we used the open-source Moses[8] SMT system. Parallel sentences were filtered to sentences of at most 80 tokens. For development, development-testing and final-testing data we additionally ensured that the random sample contained no duplicates.

BLEU scores on test set are shown on Table 4. Columns show an evaluation on test sets consisting of 2,000 parallel sentences from each of IPC sections A-H. All systems use the same phrase-table and language model trained on 1 million parallel data from all IPC sections. Different rows show results for systems that differ only in the approaches to discriminative optimization of the BLEU metric (Papineni et al., 2001). All models use the MERT implementation of (Bertoldi et al., 2009) for the 14 standard features of the Moses system. Best results are indicated by **bold face** type.

The baseline systems perform individual tuning for each IPC section, and tuning on a development set pooled from all sections. All MERT runs start from default hand-tuned weight vectors for each model. The first column (*ind.*) shows results for a system that is tuned on each individual IPC section separately, i.e., each system is tuned on a development set of 2,000 sentences from section X and evaluated on a test set of 2,000 sentences from the same section X. BLEU scores for this baseline system are already quite high, due to the repetitive nature of patents where many long and specific sub-sentential expressions are reused. The second column(*pooled*) shows a system that is tuned on a development set consisting of 2,000 sentences pooled from 250 sentences from each patent section. Result differences to *ind.* are not statistically significant.[9]

The distributed average learner AvgMERT produces some small, but statistically significant improvements over *ind.* (indicated by $*$) and *pooled* (indicated by $+$). The multi-task learner MMERT and the global model w_{avg} produced as by-product in multi-task learning show some improvements over *ind.* and AvgMERT (indicated by #). Meta-parameters for multi-task learning were set to a regularization parameter of λ=0.0001 and a convergence threshold of 0.001, resulting in convergence after 13 MERT iterations. The average weight vector w_{avg} was initialized to the zero vector.

[7] http://www.cl.uni-heidelberg.de/statnlpgroup/mmert/

[8] http://www.statmt.org/moses/

[9] Statistical significance of pairwise result differences is assessed by p-values smaller than 0.05 using an Approximate Randomization test (Riezler and Maxwell, 2005).

section	*ind.*	*pooled*	AvgMERT	MMERT	w_{avg}
A	0.5187	0.5199	**0.5213***	0.5195$^{\#}$	0.5196$^{\#}$
B	0.4877	0.4885	0.4908^{*+}	0.4911*	**0.4921**$^{*\#}$
C	0.5214	0.5175	0.5199^{*+}	**0.5218**$^{\#}$	0.5162$^{*\#}$
D	0.4724	0.4730	0.4733	**0.4736**	0.4734
E	0.4666	0.4661	0.4679^{*+}	0.4669	**0.4685***
F	0.4794	0.4801	0.4811*	0.4821*	**0.4830**$^{*\#}$
G	0.4596	0.4576	0.4607^{+}	0.4606	**0.4610***
H	0.4573	0.4560	0.4578	**0.4581**	**0.4581**

Table 4. BLEU scores on 2K parallel sentences for each of 8 patent sections.

6. Discussion

An interpretation of the results presented in Section 5 can be given as follows. The distributed average learner AvgMERT shows small, but statistically significant improvements over individual tuning for most IPC sections. This is consistent with theoretical and empirical results on distributed weight averaging for linear models (see, e.g., Zinkevich et al. (2010)). The evaluation on section C ("chemistry") shows a significant degradation. This confirms the intuition that averaging parameter weights over sections with commonalities is helpful, but not so for exceptional domains containing complex chemical formulae and compound names. Furthermore, this result is consistent with Ceauşu et al. (2011) who find that section C is best omitted when extracting a phrase table pooled across sections.

Similar results are found for the global model w_{avg} produced as by-product of multi-task learning. The multi-task learner MMERT is the only system that is able to improve results for section C over the results of individual tuning.

Clearly, all presented results have to be interpreted with a grain of salt because of the small, even if statistically significant, result differences. We conjecture that this is due to the small number of features deployed in MERT training, and can be overcome by moving to discriminative training with millions of sparse, lexicalized features. We believe that especially the fine-tuning between task-specific and average feature weights addressed by multi-task learning can be brought to bear on large-scale lexicalized models. This is due to future work.

Bibliography

Bertoldi, Nicola and Marcello Federico. Domain adaptation for statistical machine translation with monolingual resources. In *Proceedings of the 4th EACL Workshop on Statistical Machine Translation*, Athens, Greece, 2009.

Bertoldi, Nicola, Barry Haddow, and Jean-Baptiste Fouet. Improved minimum error rate training in moses. *The Prague Bulletin of Mathematical Linguistics*, (91):7–16, 2009.

Ceauşu, Alexandru, John Tinsley, Jian Zhang, and Andy Way. Experiments on domain adaptation for patent machine translation in the PLuTO project. In *Proceedings of the 15th Conference of the European Assocation for Machine Translation (EAMT 2011)*, Leuven, Belgium, 2011.

Chapelle, Olivier, Pannagadatta Shivaswamy, Srinivas Vadrevu, Kilian Weinberger, Ya Zhang, and Belle Tseng. Boosted multi-task learning. *Machine Learning*, 2011.

Daumé, Hal. Frustratingly easy domain adaptation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics (ACL'07)*, Prague, Czech Republic, 2007.

Daumé, Hal and Jagadeesh Jagarlamudi. Domain adaptation for machine translation by mining unseen words. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL-HLT 2011)*, Portland, OR, 2011.

Dredze, Mark, Alex Kulesza, and Koby Crammer. Multi-domain learning by confidence-weighted parameter combination. *Machine Learning*, 79:123–149, 2010.

Evgeniou, Theodoros and Massimiliano Pontil. Regularized multi-task learning. In *Proceedings of the 10th ACM SIGKDD conference on knowledge discovery and data mining (KDD'04)*, Seattle, WA, 2004.

Finkel, Jenny Rose and Christopher D. Manning. Hierarchical bayesian domain adaptation. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics - Human Language Technologies (NAACL-HLT'09)*, Boulder, CO, 2009.

Foster, George and Roland Kuhn. Mixture-model adaptation for SMT. In *Proceedings of the Second Workshop on Statistical Machine Translation*, Prague, Czech Republic, 2007.

Koehn, Philipp and Josh Schroeder. Experiments in domain adaptation for statistical machine translation. In *Proceedings of the Second Workshop on Statistical Machine Translation*, Prague, Czech Republic, 2007.

Obozinski, Guillaume, Ben Taskar, and Michael I. Jordan. Joint covariate selection and joint subspace selection for multiple classification problems. *Statistics and Computing*, 20:231–252, 2010.

Och, Franz Josef. Minimum error rate training in statistical machine translation. In *Proceedings of the Human Language Technology Conference and the 3rd Meeting of the North American Chapter of the Association for Computational Linguistics (HLT-NAACL'03)*, Edmonton, Cananda, 2003.

Papineni, Kishore, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. Technical Report IBM Research Division Technical Report, RC22176 (W0190-022), Yorktown Heights, N.Y., 2001.

Quattoni, Ariadna, Xavier Carreras, Michael Collins, and Trevor Darrell. An efficient projection for $l_{1,\infty}$ regularization. In *Proceedings of the 26th International Conference on Machine Learning (ICML'09)*, Montreal, Canada, 2009.

Riezler, Stefan and John Maxwell. On some pitfalls in automatic evaluation and significance testing for MT. In *Proceedings of the ACL-05 Workshop on Intrinsic and Extrinsic Evaluation Measures for MT and/or Summarization*, Ann Arbor, MI, 2005.

Schwenk, Holger. Investigations on large-scale lightly-supervised training for statistical machine translation. In *Proceedings of the International Workshop on Spoken Language Translation (IWSLT'08)*, Hawaii, 2008.

Snover, Matthew, Bonnie Dorr, and Richard Schwartz. Language and translation model adaptation using comparable corpora. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP'08)*, Honolulu, Hawaii, 2008.

Tinsley, John, Andy Way, and Paraic Sheridan. PLuTO: MT for online patent translation. In *Proceedings of the 9th Conference of the Association for Machine Translation in the Americas (AMTA 2010)*, Denver, CO, 2010.

Tsuruoka, Yoshimasa, Jun'ichi Tsujii, and Sophia Ananiadou. Stochastic gradient descent training for l1-regularized log-linear models with cumulative penalty. In *Proceedings of the 47th Annual Meeting of the Association for Computational Linguistics (ACL-IJCNLP'09)*, Singapore, 2009.

Ueffing, Nicola, Gholamreza Haffari, and Anoop Sarkar. Transductive learning for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics (ACL'07)*, Prague, Czech Republic, 2007.

Utiyama, Masao and Hitoshi Isahara. A japanese-english patent parallel corpus. In *Proceedings of MT Summit XI*, Copenhagen, Denmark, 2007.

Zhao, Bing, Matthias Eck, and Stephan Vogel. Language model adaptation for statistical machine translation with structured query models. In *Proceedings of the 20th International Conference on Computational Linguistics (COLING'04)*, Geneva, Switzerland, 2004.

Zinkevich, Martin A., Markus Weimer, Alex Smola, and Lihong Li. Parallelized stochastic gradient descent. In *Proceedings of the 24th Annual Conference on Neural Information Processing Sytems (NIPS'10)*, Vancouver, Canada, 2010.

Address for correspondence:
Stefan Riezler
riezler@cl.uni-heidelberg.de
Department of Computational Linguistics, University of Heidelberg,
Im Neuenheimer Feld 325, 69120 Heidelberg, Germany

PBML

The Prague Bulletin of Mathematical Linguistics
NUMBER 96 OCTOBER 2011 109–117

Optimising Multiple Metrics with MERT

Christophe Servan, Holger Schwenk

LIUM, University of Le Mans

Abstract

Optimisation in statistical machine translation is usually made toward the BLEU score, but this metric is questioned about its relevance to an human evaluation. Many other metrics exist but none of them are in perfect harmony with human evaluation. On the other hand, most evaluation campaigns use multiple metrics (BLEU, TER, METEOR, etc.). Statistical machine translation systems can be optimised for other metrics than BLEU, but usually the optimisation with other metrics tends to decrease the BLEU score, the main metric used in MT evaluation campaigns.

In this paper we extend the minimum error training tool of the popular Moses SMT toolkit with a scorer for the TER score, and any linear combination of the existing metrics. The TER scorer was reimplemented in C++ which results in a ten times faster execution than the reference java code.

We have performed experiments with two large-scale phrase-base SMT systems to show the benefit of the new options of the minimum error training in Moses. The first one translates from French into English (WMT 2011 evaluation). The second one was developed in the frame work of the DARPA Gale project to translate from Arabic to English in three different genres (news, web and transcribed broadcast news and conversations).

© 2011 PBML. All rights reserved. Corresponding author: servan@lium.univ-lemans.fr
Cite as: Christophe Servan, Holger Schwenk. Optimising Multiple Metrics with MERT. The Prague Bulletin of Mathematical Linguistics No. 96, 2011, pp. 109–117. doi: 10.2478/v10108-011-0016-z.

1. Introduction

It is today common practice to use a log-linear approach to combine the various models involved in statistical machine translation (SMT). This is summarised in the fundamental equation of SMT:

$$e^* = \arg\max_e \log \prod_i f_i(e,f)^{\lambda_i} = \arg\max_e \sum_i \lambda_i \log f_i(e,f) \quad (1)$$

$f_i(e,f)$ are functions of the source e and target word f sequences. Typical feature functions include the translation and distortion model, a language model on the target language and various penalties. Each feature function is weighted by a coefficient λ_i. These weights are usually optimised so that to maximise the translation performance on some development data. In the popular Moses toolkit (Koehn et al., 2007), this numerically optimisation is performed by a tool called `mert` (Bertoldi et al., 2009) which performs a simplex-style optimisation.

The provided mert tool uses the BLEU score as performance measure of the translation quality (Papineni et al., 2002). Despite the fact that the relevance of BLEU is often questioned, see for instance (Hammon, 2007), it is still a metric frequently used to evaluate machine translation, and more importantly to tune SMT systems. In fact, many metrics have been proposed to measure MT quality, for example TER (Snover et al., 2006), TERp (Snover et al., 2009) and METEOR (Banerjee and Lavie, 2005) just to mention some, and many of them are believed to correlate better with human judgements of translation quality. However, many of these metrics are not used to tune the SMT systems, at least they are not available for the Moses toolkit. It has also been observed several times that it is better to optimise towards the same metric that is later used to evaluate the SMT system.

Some previous works showed the interest of optimising toward BLEU and TER (Mauser et al., 2008; Cer et al., 2010b). Some of this work has been done with other MT systems like the Phrasal Machine Translation system (Cer et al., 2010a). Most of people use the Moses SMT system (Koehn et al., 2007) which uses MERT to optimise its parameters. We implemented a metric combination into MERT.

This kind of experiment is hard to reproduce by the fact that TER and metric combination is not directly implemented in MERT program (Bertoldi et al., 2009). That's why a MT Marathon project was suggested on this subject last year. We start over this project and we provide this tool to the Machine Translation community as open source.

In this paper we describe an extension of the mert optimiser provided in the Moses toolkit to optimise the translations performance with respect to the linear combination of multiple metrics. We have performed experiments for two well known large translations tasks: a French/English SMT system that was ranked among the best ones in the 2011 WMT evaluation and a state-of-the-art SMT system to translate from Arabic to English in the framework of the DARPA Gale project. As as special case we

will consider the frequently used combination $(\mathrm{TER} - \mathrm{BLEU})/2$, but we also report results on other combinations.

This paper is organised as follows. In the next section we will first give some details on the new scorers for the mert tool. We then present experimental results for the two tasks mentioned above. The paper concludes with a discussion of open issues.

2. A fast scorer for TER

We extended the mert tool in a flexible way so that to allow multiple metrics. The mert tool provided with Moses uses the notion of a `scorer`. This is basically an abstract C++ class that implements a particular metric, by default either the BLEU, WER and PER scores. It is not possible to use a combination of several metrics. Each scorer reads a file with n-best translations and produces a file with the corresponding scores.

We realised two additional scorers:

- the *TER scorer* which implements the translation edit rate (Snover et al., 2006) algorithm in MERT;
- the *merge scorer* which implements the combination of two or more metrics.

The TER score is usually calculated using the reference implementation of M. Snover in java. We reimplemented a TER scorer in C++ in order to have an easier interface with the mert software and to speed up the calculation of the TER score. In fact, it was observed that the calculation of the TER score with the java software can take some time, up to a minute on large development sets. This would result in a slow optimisation by MERT since the scorer is called on large n-best lists. Our implementation in C++ is roughly ten times faster than the java code.

In addition, we implemented a `merge` scorer that allows the linear combination of an arbitrary number of scorers. This allows in particular to minimise $(\mathrm{TER}-\mathrm{BLEU})/2$, but it also possible to attach a weight to each scorer, for instance $(\mathrm{TER} - 2 * \mathrm{BLEU})/2$. For this, a new switch has been added to the script *mert-moses.pl* `--sc-config`, *e.g.* `--sc-config=BLEU:2,TER:1`. All necessary configuration files are generated automatically by the script *mert-moses.pl*. A typical configuration file is shown in Table 1. This configuration file is used by the `merge` scorer in order to set weights associated with metrics. When this scorer is used, the extractor software, which is a part of the mert toolkit, succesively extract data (features and scores) for each metric. Then, the mert software is called by using the switch `--sctype MERGE`.

The mert tool always tries to maximise the returned scores, but TER is an error metric that should be minimised. Therefore, the *negTER* score is used and actually returns 1-TER.

Theses scorers are open source and released to the machine translation community with the moses SMT toolkit.

Metric	weight	feature file name	score file name
BLEU	2	BLEU_FEATURE_FILE	BLEU_SCORING_FILE
TER	1	TER_FEATURE_FILE	TER_SCORING_FILE

Table 1. Example of a configuration file for the metric combination 2xBLEU-TER.

3. Experiments

The TER scorer as well as the merge scorer were evaluated for two important tasks: the translation from French to English and the translation from Arabic to English. Both system are phrase-based, but the same procedure could also be applied to the hierarchical system `moses_chart`. We performed several experiments to asses the impact of different combinations of the scorers, keeping all other settings unchanged, in particular a fixed seed was used during the mert optimisation process. The beam search is set to 0.4 and the mert n-best is set to 100.

3.1. French/English system

Since several years, between several European languages. LIUM build this a system to translate between French and English. Our official systems are optimised using the default implementation of the mert tool which only optimises toward the BLEU score. After the evaluation, we have performed additional experiments with our new scorer. In this paper we only consider the translation from French to English. In our experiments with different metrics, we used exactly the same translation and language models than in our evaluation systems. The first model was trained on about 435M words of parallel data, while more than 7 billion words were used for the English language model. More details are given in (Schwenk et al., 2011). We report BLEU and TER scores on our development corpus (*newstest2009*), our internal test set (*newstest2010*) and the official test set of this year's evaluation (*newstest2011*). All metrics are case sensitive and include punctuations.

Table 2 summarises all the results. The first line, labelled BLEU corresponds to our official evaluation system. The second line, shows the results when using negTER as optimisation metric instead of BLEU. It is not surprising to see that this leads a decrease in the TER score, but unfortunately this comes at the cost of a worse BLEU score. BLEU is a precision metric which must be maximised while TER is an error measure which should be minimised. Therefore, it is common practice to look simultaneously at both metrics using the value $(\mathrm{TER} - \mathrm{BLEU})/2$ which must be of course minimised.

It can be clearly seen that we achieve best results by directly optimising the combined score $(\mathrm{TER} - \mathrm{BLEU})/2$. On the development data, this decreases the TER score from 53.98 to 53.58 without penalising the BLEU score. This results in an improvement of the combined score from 12.42 to 12.22. It is nice to see that the results are even

	newstest2009 (Dev)			newstest2010 (Internal test)			newstest2011 (Evaluation test)		
Optimisation	BLEU	TER	$\frac{\text{TER}-\text{BLEU}}{2}$	BLEU	TER	$\frac{\text{TER}-\text{BLEU}}{2}$	BLEU	TER	$\frac{\text{TER}-\text{BLEU}}{2}$
BLEU	29.14	53.98	12.42	29.65	52.78	11.57	30.19	51.61	10.71
TER	27.65	52.91	12.63	28.79	51.56	11.39	29.36	50.57	10.61
1xBLEU-TER	29.15	53.58	**12.22**	29.95	52.42	**11.24**	30.37	51.36	**10.50**
2xBLEU-TER	29.10	53.88	12.39	29.93	52.55	11.31	30.15	51.56	10.71
3xBLEU-TER	29.19	53.83	12.32	29.99	52.46	**11.24**	30.14	51.56	10.71
4xBLEU-TER	29.21	54.01	12.40	29.98	52.60	11.31	30.08	51.75	10.84
5xBLEU-TER	29.33	53.84	12.26	29.89	52.53	11.32	30.21	51.56	10.68

Table 2. Results for the French/English WMT 2011 translation task.

better on the internal and official test set: the BLEU and the TER score do improve when optimising on the combined criterion instead of BLEU itself. On *newstest2011* BLEU improves from 30.19 to 30.37 and TER from 51.61 to 51.36. Unfortunately, this improved system did not participate in the human evaluation. It would be very interesting to see how theses changes impact human judgements.

The merge scorer is able to perform arbitrary linear combinations of the two metrics. The corresponding results are shown in the subsequent lines of Table 2. For this task, this did not improve the overall combined performance.

3.2. GALE evaluation task

In 2005 DARPA lunched a new 5 year language technology program called Global Autonomous Language Exploitation, shortly GALE. The goal of this project was to build high performance machine translation systems from Arabic and Mandarin into English for text and speech and to prepare this information in various ways (*distillation*). Several genres were considered: news paper texts, WEB data and broadcast news and conversations automatically transcribed from speech to text. LIUM developed phrase-based systems to translate all these genres from Arabic to English in collaboration with IBM's Rosetta team.

DARPA organised yearly evaluations to measure the progress. The official metric of this evaluations was HTER which is a human judgement. In principle, this error measure corresponds to the minimal number of edit operations (insertion, deletion, substitution and block shift) a human operator has to perform to correct the errors of the automatic translation. Obviously, the human metric HTER is related to the automatic metric TER. In fact, HTER could be seen as TER with optimal references created on the fly for each sentence, or TER with respect to a pool of all possible reference translations.

We developed separate systems for the news, web and speech genre.[1] Statistics on the used parallel training data are given in Table 3. For each genre a development

[1] it is not possible to automatically separate broadcast news and broadcast conversations.

and internal test corpus was available. It consisted of about 50k English words for the news and web genre, and almost 100k words for the speech genre. Three reference translations are available for the web and broadcast conversation genres, while only one is available for the news and broadcast news genres.

Genre	# lines	# words AR	# words EN
news	3M	72.8M	76.9M
web	2.2M	46.6M	48.3M
speech	2.4M	54.4M	57.3M

Table 3. Size of the different bitexts used for our Arabic/English Gale systems.

All the experimental results are summarised in Table 4. Again, we give the BLEU score, TER and the combination $(\mathrm{TER}-\mathrm{BLEU})/2$ when optimising the systems for the different criteria.

Corpus name	Optimisation	Dev			Test		
		BLEU	TER	$\frac{\mathrm{TER}-\mathrm{BLEU}}{2}$	BLEU	TER	$\frac{\mathrm{TER}-\mathrm{BLEU}}{2}$
news	BLEU	33.56	43.80	5.12	33.56	44.25	5.34
	TER	34.07	42.81	**4.37**	34.07	43.18	**4.55**
	1xBLEU-TER	33.55	43.67	5.06	33.55	44.00	5.22
	2xBLEU-TER	33.47	43.66	5.09	33.47	44.05	5.29
	3xBLEU-TER	33.66	43.45	4.89	33.66	43.91	5.12
	4xBLEU-TER	33.63	43.68	5.03	33.63	44.01	5.19
	5xBLEU-TER	33.47	43.69	5.11	33.47	44.15	5.34
web	BLEU	40.78	61.20	10.96	39.27	61.86	11.29
	TER	40.46	60.59	**10.68**	39.24	61.43	**11.10**
	1xBLEU-TER	40.76	61.09	10.79	39.52	61.72	11.10
	2xBLEU-TER	40.62	61.01	10.87	39.28	61.56	11.14
	3xBLEU-TER	40.72	*60.86*	10.72	39.42	61.56	*11.07*
	4xBLEU-TER	40.71	61.17	10.92	39.33	61.69	11.18
	5xBLEU-TER	40.63	61.55	11.24	39.06	62.04	11.49
speech	BLEU	33.73	58.03	12.15	33.94	58.03	12.04
	TER	33.30	55.92	**11.31**	33.39	56.34	**11.47**
	1xBLEU-TER	34.04	56.98	11.47	34.13	57.17	11.52
	2xBLEU-TER	33.97	57.21	11.62	34.12	57.28	11.58
	3xBLEU-TER	33.86	57.97	12.05	33.88	58.13	12.12
	4xBLEU-TER	33.85	58.02	12.09	33.79	58.37	12.29
	5xBLEU-TER	33.85	57.91	12.03	33.84	58.13	12.14

Table 4. Results for the Arabic/English Gale translation tasks.

The results for the news genre are somehow surprising. In fact, when we tune on negTER we get, as expected, an improvement of the TER score on the development and test corpus, but the BLEU score also improves by about 0.5 points, in comparison to tuning directly on the BLEU score. Overall, the combined score $(\mathrm{TER} - \mathrm{BLEU})/2$ is substantially improved. Tuning directly on the combined metric always produced worse combined scores than tuning on negTER only. We are currently investigating this effect. Note that it can't be explained by the BLEU brevity penalty since it is 1.0 for all the experiments.

The improvements are less important for the web genre: we achieve a smaller improvement in the TER score, with only minor changes in the BLEU score. Optimising on TER or $(\mathrm{TER} - \mathrm{BLEU})/2$ gives basically the same results: 10.68 versus 10.72 on the dev data, and 11.10 and 11.07 on the test data. The improvements in the TER score for the speech genre are quite substantial, up to 2 points, with a modest loss in the BLEU score.

Overall, it is always best to tune on negTER for all the genres of the Arabic/English Gale systems, although $(\mathrm{TER}-\mathrm{BLEU})/2$ is almost quite as good for the web and speech genres.

4. Conclusion

This paper addressed the important issue on which automatic measure one should optimise the weights of the feature functions in the log-linear model used in SMT. For this, we extended the mert optimisation software in the very popular Moses SMT toolkit with scorers for TER and a *merge* scorer which allows to optimise an arbitrary linear combination of other metrics. Since the TER scorer is implemented in C++ in performs roughly ten times faster than the reference java code. The whole software is open-source and available in Moses svn[2].

We have performed experiments with two large-scale phrase-base SMT systems. The first one translates from French into English (WMT 2011 evaluation). The second one was developed in the frame work of the DARPA Gale project to translate from Arabic to English in three different genres (news, web and transcribed broadcast news and conversations). For the WMT system we have observed, like many others before, that tuning on one metric, concretely BLEU or TER, obviously improves the performance measured in this metric, but usually worsens other metric. Best results were obtained when tuning directly on a linear combination of both, usually $(\mathrm{TER} - \mathrm{BLEU})/2$.

For the Arabic/English system, significant improvements of the combined score $(\mathrm{TER} - \mathrm{BLEU})/2$ were obtained, in particular for the news and speech genre. However, in contrast to the WMT task, this can be obtained by tuning on TER only. We are currently investigating the reasons for these effects: is the tuning affected by the speci-

[2] `https://mosesdecoder.svn.sourceforge.net/svnroot/mosesdecoder`

ficities of the language pair, i.e. French/English versus Arabic/English, the number of available reference translations, ... ?

In the future, we plan to add further metrics, namely TERp and METEOR, and we try to study which metric combination is best related to human judgements.

5. Acknowledgements

This work was partially supported under the GALE program of the Defense Advanced Research Projects Agency, Contract No. HR0011-06-C-0022, and by the European Commission under the project EUROMATRIXPLUS.

Bibliography

Banerjee, Satanjeev and Alon Lavie. Meteor: An automatic metric for MT evaluation with improved correlation with human judgments. In *Workshop on Intrinsic and Extrinsic Evaluation Measures for MT and/or Summarization at the 43th Annual Meeting of the Association of Computational Linguistics*, 2005.

Bertoldi, Nicola, Barry Haddow, and Jean-Baptiste Fouet. Improved minimum error rate training in Moses. *The Prague Bulletin of Mathematical Linguistics*, 91, 2009.

Cer, Daniel, Michel Galley, Daniel Jurafsky, and Christopher Manning. Phrasal: A toolkit for statistical machine translation with facilities for extraction and incorporation of arbitrary model features. In *North American Association of Computational Linguistics - Demo Session (NAACL-10)*, 2010a.

Cer, Daniel, Christopher D. Manning, and Daniel Jurafsky. The best lexical metric for phrase-based statistical mt system optimization. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, HLT '10, pages 555–563, Stroudsburg, PA, USA, 2010b. Association for Computational Linguistics. ISBN 1-932432-65-5. URL `http://portal.acm.org/citation.cfm?id=1857999.1858079`.

Hammon, Olivier. Rapport du projet CESTA : Campagne d'evaluation des systèmes de traduction automatique. Technical report, ELDA, 2007.

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, demonstration session*, 2007.

Mauser, Arne, Saša Hasan, and Hermann Ney. Automatic evaluation measures for statistical machine translation system optimization. In *LREC'08*, 2008.

Papineni, Kishore, Salim Roukos, Todd Ward, and Wei-Jing Zhu. BLEU: a method for automatic evaluation of machine translation. In *ACL*, 2002.

Schwenk, Holger, Patrik Lambert, Loïc Barrault, Christophe Servan, Haithem Afli, Sadaf Abdul-Rauf, and Kashif Shah. LIUM's SMT machine translation systems for WMT 2011. In *6th Workshop on statistical Machine Translation*, 2011.

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

Snover, Matthew, Nitin Madnani, Bonnie 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 at the 12th Meeting of the European Chapter of the Association for Computational Linguistics (EACL-2009)*, pages 259–268, 2009.

Address for correspondence:
Christophe Servan
servan@lium.univ-lemans.fr
LIUM, University of Le Mans
72085 Le Mans cedex 9, FRANCE

The Prague Bulletin of Mathematical Linguistics
NUMBER 96 OCTOBER 2011

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 two copies 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.html`. If there are any technical problems, please contact the editorial staff at `pbml@ufal.mff.cuni.cz`.