PBML

The Prague Bulletin of Mathematical Linguistics

NUMBER 111 OCTOBER 2018 87–96

An Easily Extensible HMM Word Aligner

Jetic Gū,[a b] Anahita Mansouri Bigvand,[a] Anoop Sarkar[a]

[a] Simon Fraser University, Burnaby, Canada
[b] Zhejiang University, Hangzhou, China

Abstract

In this paper, we present a new word aligner with built-in support for alignment types, as well as comparisons between various models and existing aligner systems. It is an open source software that can be easily extended to use models of users' own design. We expect it to suffice the academics as well as scientists working in the industry to do word alignment, as well as experimenting on their own new models. Here in the present paper, the basic designs and structures will be introduced. Examples and demos of the system are also provided.

1. Introduction

Word alignment is a very important component in a machine translation system. Classical approaches include the IBM Models 1–5 (Brown et al., 1993) and the Hidden Markov Model (Vogel et al., 1996; Och and Ney, 2000a), which usually work quite well but insufficient for specific language pairs (Chinese-English for example, as shown in Section 4.1). As more human annotated data became available, a lot of supervised and semi-supervised algorithms were also proposed and had shown improvements, as demonstrated in Mansouri Bigvand et al. (2017).

In this paper we present a new open source HMM Aligner. The HMM Aligner (from this point forward will simply be referred to as The Aligner) not only contains built-in classic models, but also some of the supervised learning models as well (Section 3.3), while providing an extensible API. Users can easily combine existing models to form new learning sequences. For those that are familiar with the BaumWelch algorithm of HMM, it would only take a matter of minutes to implement their own extensions to the HMM model using The Aligner. The Aligner is written in Python using libraries such as Numpy so that it is efficient and also highly readable.

Corresponding author: jeticg@sfu.ca

The comparisons of The Aligner and current systems will be provided in Section 2, while the design and models of The Aligner are in Section 3. In Section 4 experiment results are presented to demonstrate the capability of The Aligner.

Current version of The Aligner is available at `https://github.com/sfu-natlang/HMM-Aligner` under the MIT license.

2. Existing Systems

GIZA++ (Och and Ney, 2003) has long been the de facto software to use and compare to when doing word alignment. Although generally considered as reliable, it is highly sophisticated, making it very difficult to extend.

BerkeleyAligner (Liang et al., 2006) is a word aligner developed by The Berkeley NLP Group. Comparing to GIZA++ it is simpler to setup and use. It claims higher accuracy than GIZA++.

FastAlign (Dyer et al., 2013) is a very simple and fast word aligner. It claims better performance both in terms of speed and accuracy than GIZA++.

The aligners above are all made production ready, and there are not plenty of resources on how to build extensions. It would be relatively difficult if one wishes to develop their own HMM models using these software.

The presence of The Aligner aims at solving that exact problem. It provides an easily extendable interface, plenty of example models and it is written in a very friendly and flexible language: Python. Also it has built-in support for alignment type, which is a truly valuable resource in NLP with huge potential.

3. Design and Structure of The Aligner API

This section provides a brief introduction to how The Aligner works and an overview of its API design.

3.1. Workflow

The workflows of the training process and decoding process are presented in Figure 1.

In the beginning of a training sequence, the dataset will first be lexicalised, during which the original text in the dataset are converted into numbers using dictionary created with the training corpus. Each unique word from the source language and the target language will receive a unique index on which making it easier to do operations.

Then, during the training process the specified model will be loaded. Each model has its own training sequence, which performs the EM algorithm on different parts of the dataset in the order of ones choosing. The example in Figure 1 is the training sequence of HMM model, in which the translation probability is initialised using IBM model 1, and then the HMM model is trained. The details of this model will be introduced later in Section 3.3.

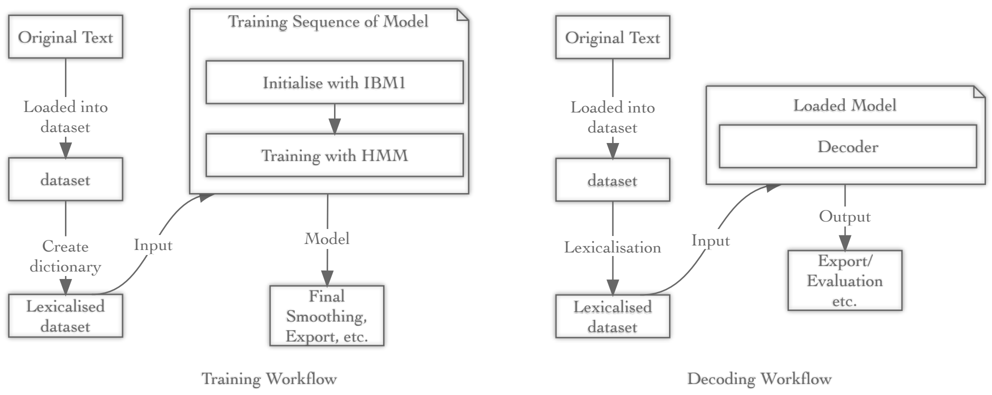

Figure 1. Workflow

During the evaluation process, lexicalised dataset is fed into the decoder to get the alignment.

3.2. Base Models

The Aligner provides two base models on which all other built-in models are built: the IBM model 1 (Brown et al., 1993) (Eq 1) and the HMM model (Och and Ney, 2000a) (Eq 2). Here $\mathbf{f}$ is the source sentence, and $\mathbf{e}$ the target sentence. $\mathbf{a}$ is the alignment from source to target: the jth source word is aligned to the a_jth target word. I is the length of the source sentence.

$$\Pr_{IBM1}(\mathbf{a}|\mathbf{f},\mathbf{e}) = \prod_{j=1}^{I} \mathbf{P}(f_j|e_{a_j}) \tag{1}$$

$$\mathbf{Pr}_{HMM}(\mathbf{a}|\mathbf{f},\mathbf{e}) = \prod_{j=1}^{I} [\mathbf{P}(a_j|a_{j-1}, I)\mathbf{P}(f_j|e_{a_j})] \tag{2}$$

Both base models come with highly customisable API for training. Users using The Aligner can focus on modifying the models, applying smoothing, and designing their own training sequences on a higher level without having to worry about data structure and other lower level detail. All of the computational parts are optimised using Numpy and Cython to ensure good performance.

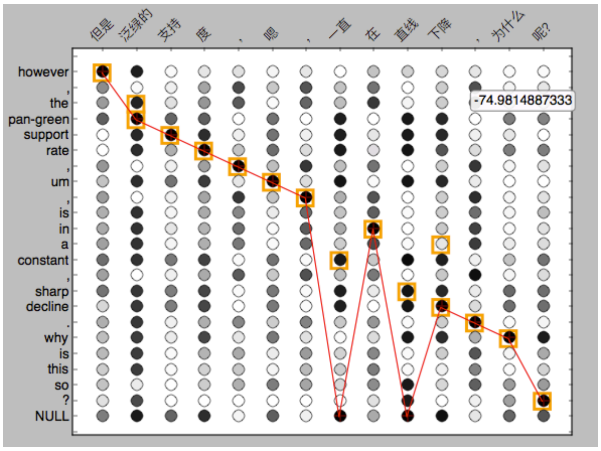

Figure 2. Screenshot of the figure of a pair of sentences. The gold alignment is represented with square boxes.

3.3. Alignment Type

Alignment of words between different languages can reflect more than just semantic meaning, it could also have been others including but not limited to function link, clausal link, modifier link, and also language specific links (Li et al., 2010). This information has proven to be quite useful for doing both supervised and semi-supervised aligner training as demonstrated in Mansouri Bigvand et al. (2017).

The Aligner provides built-in model that utilises alignment types (Eq 3) presented in Mansouri Bigvand et al. (2017). It uses both alignment type and POS tag information to enhance the baseline model to achieve better results. $\mathbf{h}$ represents the alignment type of the alignment: h_j is the alignment type of jth source word and a_jth target word.

$$\mathbf{Pr}_{HMMType}(\mathbf{a}, \mathbf{h}|\mathbf{f}, \mathbf{e}) = \prod_{j=1}^{I}[\mathbf{P}(a_j|a_{j-1}, I)\mathbf{P}(f_j|e_{a_j})\mathbf{P}(h_j|f_j, e_{a_j})] \tag{3}$$

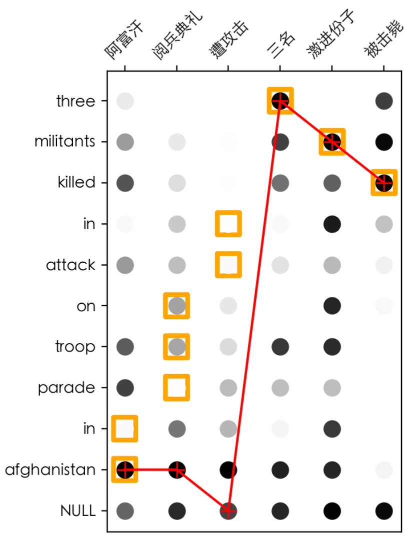

Figure 3. HMM Baseline

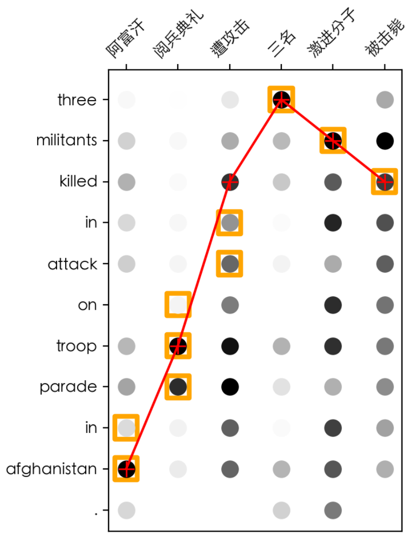

Figure 4. HMM With Alignment Type

3.4. Extended HMM models

Example implementation of extensions to the HMM model, as presented in K. Toutanova's 2002 paper is also provided. In the experiment section, we will be comparing the following extension (Eq 4. $\mathbf{P}^*(a_j|a_{j-1}, e_{a_{j-1}}, I)$ is defined in Eq 5):

$$\mathbf{Pr}_{\mathrm{HMMType}}(\mathbf{a}, \mathbf{h}|\mathbf{f}, \mathbf{e}) = \prod_{j=1}^{I}[\mathbf{P}^*(a_j|a_{j-1}, e_{a_{j-1}}, I)\mathbf{P}(f_j|e_{a_j})\mathbf{P}(fTag_j|eTag_{a_j})], \tag{4}$$

$$\mathbf{P}^*(a_j|a_{j-1}, e_{a_{j-1}}, I) = \delta(a_j, a_{j-1})\mathbf{P}(\mathbf{stay}|e_{a_{j-1}}) + \left\{\begin{array}{r} (1-\delta(a_j, a_{j-1})) \\ (1-\mathbf{P}(\mathbf{stay}|e_{a_{j-1}})) \\ (\mathbf{P}(a_j, a_{j-1}, I)) \end{array}\right\} \tag{5}$$

$\delta(a_j, a_{j-1})$ in Eq 5 is the Kronecker delta function. $\mathbf{P}(\mathbf{stay}|e_{a_{j-1}})$ is the probability of the alignment of the next source word being the same target word, given the aligned target word of the previous source word.

3.5. Displaying Alignment Scores

The Aligner also includes built-in feature of displaying the alignment score of each pair of words of sentences during decoding (Figure 2). The score of each pair is presented with grey-scaled colours. In any column, if the colours of two circles appear close then the score of these two are very close, v.v.. If one moves the cursor on any circle, the score of that corresponding pair of words will appear next to the cursor, making it easier to do debugging and present results. The gold alignment is also displayed to make doing comparisons easier.

This feature is very useful for debugging and also comparing results of different models.

3.6. Loading And Saving Models

The Aligner allows users to save the trained models for reuse. Not only are built-in models savable but also user customised models as well. It does not require any extra code to make it work, the API will automatically clean up the data structures, make saving and loading the least of ones concern when it comes to model design.

3.7. Intersection

Intersecting the alignment results from source-target training and target-source training usually gives better results, as demonstrated by Liang et al. (2006). The Aligner has built-in support for such intersection for all models without the need of extra code. Also, since intersection is done in parallel, it will not require extra time to train and decode.

4. Experiments

The experiments will focus on the quality of alignments produced by The Aligner's built-in models, including the models with alignment types and POS tags, by comparing Alignment Error Rates (Och and Ney, 2000b) and F1 Scores.

The datasets used in the experiments are: first for the datasets with human annotated alignment types and POS tags, we used the GALE Chinese-English Parallel Aligned Treebank[1] which includes parallel text from news broadcasts, news articles, and online discussion panels; then, experiments were also carried out on the French-English Hansard Canadian Parliament dataset[2] and the German-English Dataset from the European Parliament Proceedings Parallel Corpus (Koehn, 2005). The latter two datasets are all parliament proceedings. Sentences in parliament proceedings appear

[1]Catalog numbers: LDC2014T25; LDC2015T04; LDC2015T06; LDC2015T18; LDC2016T19; LDC2017T05

[2]Catalog numbers: LDC95T20

more formal than daily conversations, are highly structured and sophisticated, and contain significant amount of legal-domain words. They also do not come with alignment types, so we only compared the quality of models that does not require alignment types, including the extended HMM model.

The German-English dataset and French-English dataset we used are all parliament proceedings, which are also highly popular dataset for NLP experiments. Although all in spoken language, the sentences often are highly structured and sophisticated, with significant amount of legal-domain words.

POS tags of datasets that originally do not contain POS tags are obtained using The Stanford POS Tagger (Toutanova et al., 2003).

Table 1 gives information on the sizes of the datasets. Table 2 contains information regarding percentages of words according to their occurrences in each dataset.

language pair	Unique Words		POS Tags		Sentences	
	source	target	source	target	train	test
Chinese-English	107 503	49 557	40	57	125 989	1956
French-English	92 280	82 074	18	45	951 462	447
German-English	395 952	133 835	27	55	1 890 489	150

Table 1. Information on Datasets

occurrences	Chinese-English		French-English		German-English	
	source	target	source	target	source	target
=1	48.00%	32.08%	35.77%	35.49%	50.32%	41.32%
≤3	70.18%	54.86%	55.98%	57.93%	70.16%	60.44%
≤5	78.15%	63.94%	64.01%	63.93%	76.95%	67.50%
≤10	86.20%	74.13%	72.90%	72.86%	83.98%	75.25%

Table 2. Percentage of words according to their occurrence

4.1. Experiments on Datasets with Alignment Types and POS Tags

Table 3 shows the performance of built-in models mentioned in Section 3.2 and also the result of Fast Aligner, GIZA++, and Berkeley Aligner on Chinese-English dataset. For all models, experiments were run on the same settings. It is apparent that models supporting alignment types and POS tags produce alignments with better quality.

Model	Precision	Recall	AER	F1-score
IBM1	0.5279	0.4204	0.5320	0.4680
IBM1 (Intersect)	0.8457	0.3686	0.4866	0.5134
HMM	0.7233	0.5063	0.4044	0.5956
HMM (Intersect)	0.9135	0.4431	0.4032	0.5968
HMM+Extensions	0.6865	0.5465	0.3915	0.6085
HMM+Extensions (Intersect)	0.8907	0.4758	0.3798	0.6202
HMM+Type	0.7257	**0.6189**	0.3319	0.6681
HMM+Type (Intersect)	0.9154	0.5690	**0.2982**	**0.7018**
GIZA++(Model 4)	0.6513	0.5825	0.3850	0.6150
GIZA++ (Manual Intersect)	**0.9481**	0.4049	0.4325	0.5675
Fast-Align	0.6263	0.6142	0.3798	0.6202
Fast-Align (Manual Intersect)	0.8674	0.5064	0.3605	0.6395
Berkeley-Aligner	0.7638	0.6116	0.3207	0.6793

Table 3. Chinese-English Dataset test results

4.2. Experiments on Datasets without Alignment Types

Experiments in this section are carried out on the French-English dataset (Table 4) and German-English dataset (Table 5).

Model	Precision	Recall	AER	F1-score
IBM1	0.5570	0.7038	0.3928	0.6218
HMM	0.7930	0.8462	0.1877	0.8187
HMM+Extension	0.7663	0.8834	0.1936	0.8207
HMM+Extension (Intersect)	0.9389	0.7979	0.1264	0.8627
GIZA++ (Model 4)	0.7848	0.7940	0.2115	0.7894
GIZA++ (Manual Intersect)	**0.9773**	0.7214	0.1548	0.8301
Fast-Align	0.7679	0.8294	0.2091	0.7975
Fast-Align (Manual Intersect)	0.7584	0.8826	0.1997	0.8158
Berkeley-Aligner	0.8677	**0.9113**	**0.1151**	**0.8899**

Table 4. French-English Dataset test results

Model	Precision	Recall	AER	F1-score
IBM1	0.5982	0.6388	0.3830	0.6178
HMM	0.7253	0.7513	0.2626	0.7381
HMM+Extension	0.7333	0.7693	0.2500	0.7509
HMM+Extension (Intersect)	0.9295	0.7066	0.1941	0.8029
GIZA++ (Model 4)	0.8611	0.7429	0.2006	0.7976
GIZA++ (Manual Intersect)	**0.9593**	0.6349	0.2334	0.7641
Fast-Align	0.7275	0.7587	0.2581	0.7428
Fast-Align (Manual Intersect)	0.8718	0.6240	0.2694	0.7274
Berkeley-Aligner	0.8898	**0.8227**	**0.1435**	**0.8549**

Table 5. German-English Dataset test results

As shown by the data, The Aligner performs fairly well despite its simplicity. It is able to produce competitive if not better results than the other Aligners. Of course these experiments do not represent the full potential of the built-in models. By applying various smoothing techniques for each individual language pair one can surely achieve better results. It is not this paper's intention to declare that The Aligner is a far better alternative to the other software available, but only that it is highly competitive while having the unique advantage of being easy to extend.

5. Conclusions

This project is maintained by the Simon Fraser University Natural Language Lab. We have plans to gradually include more sample models in the future.

The Aligner aims at providing an easy-to-extend interface to users wishing to do word alignment. The Aligner can also be used for educational purposes such as teaching students to do word alignment and learning HMM models and alignments. With its simplicity and flexibility, we believe it will be proven to be very useful for researchers as well as industrial productions.

Bibliography

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

Dyer, Chris, Victor Chahuneau, and Noah. A Smith. A Simple, Fast, and Effective Reparameterization of IBM Model 2. In *Proceedings of the Conference of NAACL: Human Language Technologies*, ACL 2013, pages 644–649. ACL, 2013. URL `http://repository.cmu.edu/cgi/viewcontent.cgi?article=1038&context=lti`.

Koehn, Philipp. Europarl: A Parallel Corpus for Statistical Machine Translation. In *Conference Proceedings: the tenth Machine Translation Summit*, pages 79–86, Phuket, Thailand, 2005. AAMT. URL `http://mt-archive.info/MTS-2005-Koehn.pdf`.

Li, Xuansong, Niyu Ge, Stephen Grimes, Stephanie M. Strassel, and Kazuaki Maeda. Enriching word alignment with linguistic tags. In *In Proceedings of the Seventh International Conference on Language Resources and Evaluation*, 2010.

Liang, Percy, Ben Taskar, and Dan Klein. Alignment by Agreement. In *Proceedings of the Main Conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, HLT-NAACL '06, pages 104–111, Stroudsburg, PA, USA, 2006. ACL. doi: 10.3115/1220835.1220849. URL `http://dx.doi.org/10.3115/1220835.1220849`.

Mansouri Bigvand, Anahita, Te Bu, and Anoop Sarkar. Joint prediction of word alignment with alignment types. In *Transactions of ACL*, TACL 2017. ACL, 2017.

Och, Franz Josef and Hermann Ney. A Comparison of Alignment Models for Statistical Machine Translation. In *Proceedings of the 18th Conference on Computational Linguistics - Volume 2*, COLING '00, pages 1086–1090, Stroudsburg, PA, USA, 2000a. ACL. doi: 10.3115/992730.992810. URL `http://dx.doi.org/10.3115/992730.992810`.

Och, Franz Josef and Hermann Ney. Improved Statistical Alignment Models. In *Proceedings of the 38th Annual Meeting of ACL*, pages 440–447, 2000b.

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

Toutanova, Kristina, Dan Klein, Christopher D. Manning, and Yoram Singer. Feature-rich Part-of-speech Tagging with a Cyclic Dependency Network. In *Proceedings of the 2003 Conference of NAACL on Human Language Technology - Volume 1*, NAACL '03, pages 173–180, Stroudsburg, PA, USA, 2003. ACL. doi: 10.3115/1073445.1073478. URL `http://dx.doi.org/10.3115/1073445.1073478`.

Vogel, Stephan, Hermann Ney, and Christoph Tillmann. HMM-based Word Alignment in Statistical Translation. In *Proceedings of the 16th Conference on Computational Linguistics - Volume 2*, COLING '96, pages 836–841, Stroudsburg, PA, USA, 1996. ACL. doi: 10.3115/993268.993313. URL `http://dx.doi.org/10.3115/993268.993313`.

Address for correspondence:
Jetic Gū
`jeticg@sfu.ca`
NATLANG Lab, Simon Fraser University
8888 University Dr, Burnaby
BC V5A1S6, Canada