Proceedings of the International Multiconference on
Computer Science and Information Technology pp. 397–401
ISBN 978-83-60810-27-9
ISSN 1896-7094

SyMGiza++: A Tool for Parallel Computation of Symmetrized Word Alignment Models

Marcin Junczys-Dowmunt
Adam Mickiewicz University
Faculty of Mathematics and Computer Science
ul. Umultowska 87, 61-614 Poznań, Poland
Email: junczys@amu.edu.pl

Arkadiusz Szał
Adam Mickiewicz University
Faculty of Mathematics and Computer Science
ul. Umultowska 87, 61-614 Poznań, Poland
Email: arekszal@amu.edu.pl

***Abstract*—SyMGiza++ — a tool that computes symmetric word alignment models with the capability to take advantage of multi-processor systems — is presented. A series of fairly simple modifications to the original IBM/Giza++ word alignment models allows to update the symmetrized models between each iteration of the original training algorithms. We achieve a relative alignment quality improvement of more than 17% compared to Giza++ and MGiza++ on the standard Canadian Hansards task, while maintaining the speed improvements provided by MGiza++'s capability of parallel computations.**

I. Introduction

WORD alignment is a key component of the training procedure for statistical machine translation systems. The classic tool used for this task is Giza++ [1] which is an implementation of the so-called IBM Models 1-5 [2], the HMM model by [3] and its extension by [1], and Model 6 [1].

All these models are asymmetric, i.e. for a chosen translation direction, they allow for many-to-one alignments, but not for one-to-many alignments. Training two models in opposite directions and symmetrizing the resulting word alignments is commonly employed to improve alignment quality and to allow for more natural alignments. The two alignment models are trained fully independently from each other. Symmetrization is then performed as a post-processing step. Previous work [4], [5] has shown that the introduction of symmetry during training results in better alignment quality than post-training symmetrization.

The approaches from [4], [5] as well as our method still require the computation of two directed models which use common information during the training. Employing a multiprocessor system for the parallel computation of theses models is a natural choice. However, Giza++ was designed to be single-process and single-thread. MGiza++ [6] is an extension of Giza++ which allows to start multiple threads on a single computer.

We therefore choose to extend MGiza++ with the capability to symmetrized word alignments models to tackle both problems in one stroke. The resulting tool SyMGiza++ is described in this work. The paper will be organized as follows: Section 2 provides a short overview of Giza++ and MGiza++ and the above mentioned methods of symmetrized alignment model training. In Sec. 3 we give a formal description of our modifications introduced into the classical word alignment models implemented in Giza++ and MGiza++. The evaluation methodology and results are provided in Sec. 4. Finally, conclusions are presented in Sec. 5.

II. Previous Work

A. Giza++ and MGiza++

Giza++ implements maximum likelihood estimators for several statistical alignment models, including Model 1 through 5 described by [2], a HMM alignment model by [3] and Model 6 from [1]. The EM [7] algorithm is employed for the estimation of the parameters of the models. During the EM algorithm two steps are applied in each iteration: in the first step, the E-step, the previously computed model or a model with initial values is applied to the data. The expected counts for specific parameters are collected using the probabilities of this model. In the second step, the M-step, these expected counts are taken as fact and used to estimate the probabilities of the next model. A correct implementation of the E-step requires to sum over all possible alignments for one sentence pair. This can be done efficiently for Model 1 and 2, and using the Baum-Welch algorithm also for the HMM alignment model [1].

For Models 3 through 6, a complete enumeration of alignments cannot be accomplished in a reasonable time. This can be approximated by using only a subset of highly scored alignments. In [2] it has been suggested to use only the alignment with the maximum probability, the so-called Viterbi alignment. Another approach resorts to the generation of a set of high probability alignments obtained by making small changes to the Viterbi alignment. [8] proposed to use the neighbour alignments of the Viterbi alignment.

MGiza++ [6] is a multi-threaded word alignment tool that utilizes multiple threads to speed up the time-consuming word alignment process. The implementation of the word alignment models is based on Giza++ and shares large portions of source code with Giza++. The main differences rely on multiple thread management and the synchronization of the counts collecting process. Similarly, our tool in turn incorporates large portions of the MGiza++ source code extending MGiza++'s capabilities of using multiple processors with the ability to compute symmetrized word alignment models in a multiprocessor environment. Since the multiprocessing aspect is mainly

a feature of the original MGiza++, we will not discuss it in this paper and refer the reader to the original paper on MGiza++ [6].

B. Symmetrized Word Alignment Models

The posteriori symmetrization of word alignments has been introduced by [1]. This method does not compute symmetrized word alignment models during the training procedure, but uses heuristic combination methods after the training. We described it in more detail in Sec. III-E. The best results of [1] for the Hansards task are 9.4% AER (using Model 4 in the last training iterations) and 8.7% AER (using the more sophisticated Model 6).

[4] improve the IBM alignment models, as well as the Hidden-Markov alignment model using a symmetric lexicon model. This symmetrization takes not only the standard translation direction from source to target into account, but also the inverse translation direction from target to source. In addition to the symmetrization, a smoothed lexicon model is used. The performance of the models is evaluated for Canadian Hansards task, where they achieve an improvement of more than 30% relative to unidirectional training with Giza++ (7.5% AER) is achieved.

In [9], the symmetrization is performed after training IBM and HMM alignment models in both directions. Using these models, local costs of aligning a source word and a target word in each sentence pair are estimated and graph algorithms are used to determine the symmetric alignment with minimal total costs. The automatic alignments created in this way are evaluated on the German–English Verbmobil task and the French–English Canadian Hansards task (6.6% AER).

Another unsupervised approach to symmetric word alignment is presented by [5]. Two simple asymmetric models are trained jointly to maximize a combination of data likelihood and agreement between the models. The authors restrict their experiments to IBM Models 1 and 2 and a new jointly trained HMM alignment model. They report an AER of 4.9% — a 29% reduction over symmetrized IBM model 4 predictions — for the Canadian Hansards task.

III. SyMGiza++ — Symmetrized MGiza++

In this section we will describe our modifications to the well known alignment models from [2] and [1].

We do not introduce changes to the main parameter estimation procedure. Instead, we modify the counting phase of each model to adopt information provided by both directed models simultaneously. The parameter combination step is executed in the main thread. In the following subsections, the formal aspects of the parameter combination will be outlined separately for each model. The notation has been adopted from [2] and we refer the reader to this work for details on the original models that will not be repeated in this paper.

A. Model 1

Model 1 is the first of the IBM models described extensively by [2] which have been implemented accurately in Giza++ and MGiza++.

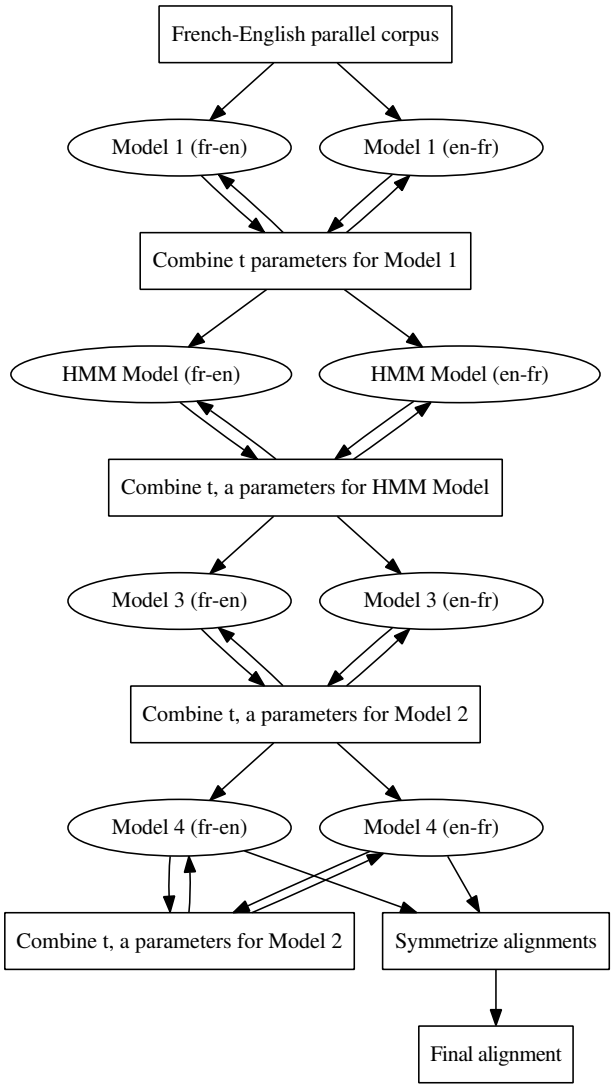

Fig. 1. General training scheme for SyMGiza++

In order to distinguish between the parameters of the two simultaneously computed alignment models we will use α and β as subscripts for the parameters of the first and second model respectively. For our English-French training corpus we compute the following two models:

$$Pr_\alpha(\mathbf{f}|\mathbf{e}) = \frac{\epsilon(m|l)}{(l+1)^m} \sum_{\mathbf{a}} \prod_{j=1}^{m} t_\alpha(f_j|e_{a_j}) \tag{1}$$

$$Pr_\beta(\mathbf{e}|\mathbf{f}) = \frac{\epsilon(l|m)}{(m+1)^l} \sum_{\mathbf{b}} \prod_{i=1}^{l} t_\beta(e_i|f_{b_i}) \tag{2}$$

where l and m are the lengths of the French sentence $\mathbf{f}$ and the English sentence $\mathbf{e}$ respectively, $\mathbf{a}$ and $\mathbf{b}$ are the directed alignments between the sentences and t_α and t_β the directed *translation probabilities* between the French and English words f and e. Due to the simplicity of this model, it is straightforward to introduce our changes in the counting method used during the E-step of the EM-algorithm. The only

free parameters of Model 1 are the translation probabilities t_α and t_β which are estimated by:

$$t_\alpha(f|e) = \frac{\sum_{s=1}^{S} c(f|e;\mathbf{f}^{(s)},\mathbf{e}^{(s)})}{\sum_{f'}\sum_{s=1}^{S} c(f'|e;\mathbf{f}^{(s)},\mathbf{e}^{(s)})}, \tag{3}$$

where S is the number of sentences in the parallel training corpus. $c(f|e;\mathbf{f},\mathbf{e})$ is the expected count of times the words f and e form translations in the given sentences $\mathbf{f}$ and $\mathbf{e}$, in the inverted model $c(e|f;\mathbf{e},\mathbf{f})$ is used.

In the original model, the expected counts $c(f|e;\mathbf{f},\mathbf{e})$ are calculated from the t values of the preceding iteration with the help of the following two formulas:

$$c(f|e;\mathbf{f},\mathbf{e}) = \sum_{\mathbf{a}} Pr_\alpha(\mathbf{a}|\mathbf{f},\mathbf{e}) \sum_{i,j} \delta(f,f_j)\delta(e,e_i), \tag{4}$$

and

$$Pr_\alpha(\mathbf{a}|\mathbf{f},\mathbf{e}) = \frac{\prod_{j=1}^{m} t_\alpha(f_j|e_{a_j})}{\sum_{\mathbf{a}}\prod_{j=1}^{m} t_\alpha(f_j|e_{a_j})}, \tag{5}$$

where δ is the Kronecker function[1]. Equations (3) and (4) are common for all models discussed in this section. Our modifications are restricted to (5) which is replaced by

$$\begin{aligned} Pr_\alpha(\mathbf{a}|\mathbf{f},\mathbf{e}) &= \frac{\prod_{j=1}^{m} \bar{t}(f_j,e_{a_j})}{\sum_{\mathbf{a}}\prod_{j=1}^{m} \bar{t}(f_j,e_{a_j})} \\ &= \frac{\prod_{j=1}^{m} \left(t_\alpha(f_j|e_{a_j}) + t_\beta(e_{a_j}|f_j)\right)}{\sum_{\mathbf{a}}\prod_{j=1}^{m} \left(t_\alpha(f_j|e_{a_j}) + t_\beta(e_{a_j}|f_j)\right)} \end{aligned} \tag{6}$$

Here we see the only difference between the standard Model 1 and our symmetrized version. By taking into account the translation probabilities from the previous iteration of both directed models we inform each model about the estimates of its counterparts. The following intuition applies: a French word is a good translation of an English word, if the English word is a good translation of the French word as well. This cannot be easily captured in the directed models without breaking up its sound probabilistic interpretation, as it happens here. However, since we modify only the way expected counts are obtained, the requirement imposed by [2] that

$$\sum_f t(f|e) = 1$$

still applies. Our modifications do not interfere with the EM procedure. The parameters for the inverted model are obtained analogously.

It should be noted that most of the time — despite the symmetry of the sum $\tilde{t}_\alpha(f|e) + \tilde{t}_\beta(e|f)$ occurring in both counts — $c(f|e;\mathbf{f},\mathbf{e})$ and $c(e|f;\mathbf{e},\mathbf{f})$ will have different values for the same words and sentences. This is due to the differences in the alignment direction. Therefore $t_\alpha(f|e) \neq t_\beta(e|f)$ in the general case.

[1] $\delta(i,j) = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{otherwise} \end{cases}$.

B. Model 2

Although it is common practice to replace Model 2 during the training procedure with the HMM Model described in the next subsection, we need to modify its counting procedure as well. Model 2 is used to score a subset of alignments during the training procedure of the more sophisticated Models 3 and 4 which — in contrast to the lower models — cannot efficiently enumerate all possible alignments.

Model 2 introduces a second type of free parameters: the *alignment probabilities* a. These a parameters capture the probability that given the lengths of both sentences, a French word at position j is aligned with an English word at position a_j. The complete model is given by [2] as:

$$Pr_\alpha(\mathbf{f}|\mathbf{e}) = \epsilon(m|l) \sum_{\mathbf{a}} \prod_{j=1}^{m} \left(t_\alpha(f_j|e_{a_j}) a_\alpha(a_j|j,m,l)\right) \tag{7}$$

The general scheme described in (3) and (4) for the estimation of t values is the same for Model 2 as for Model 1. The alignment probabilities are estimated similarly:

$$a_\alpha(i|j,m,l) = \frac{\sum_{s=1}^{S} c(i|j,m,l;\mathbf{f}^{(s)},\mathbf{e}^{(s)})}{\sum_{i'}\sum_{s=1}^{S} c(i'|j,m,l;\mathbf{f}^{(s)},\mathbf{e}^{(s)})}, \tag{8}$$

$$c(i|j,m,l;\mathbf{f},\mathbf{e}) = \sum_{\mathbf{a}} Pr_\alpha(\mathbf{a}|\mathbf{f},\mathbf{e})\delta(i,a_j). \tag{9}$$

Again, we only modify $Pr(\mathbf{a}|\mathbf{f},\mathbf{e})$ in (4) and (9) to obtain our symmetrized version of the alignment models:

$$Pr_\alpha(\mathbf{a}|\mathbf{f},\mathbf{e}) = \frac{\prod_{i=1}^{m} \left(\bar{t}(f_j,e_{a_j})\bar{a}(a_j,j,m,l)\right)}{\sum_{\mathbf{a}}\prod_{j=1}^{m} \left(\bar{t}(f_j,e_{a_j})\bar{a}(a_j,j,m,l)\right)} \tag{10}$$

where $\bar{t}(f,e)$ is defined as before for Model 1 and $\bar{a}(i,j,m,l) = a_\alpha(i|j,m,l) + a_\beta(j|i,l,m)$. The effect of information sharing between the two inverted models Pr_α and Pr_β is even increased for Model 2 since translation and alignment probabilities interact during the estimation of both types of parameters for the next iteration.

C. HMM Model

The HMM Alignment Model has been introduced by [3] and is used in the GIZA++ family of alignment tools as a replacement for the less effective Model 2. The HMM alignment model is given by the following formula which at first looks very similar to (7):

$$P_\alpha(\mathbf{f}|\mathbf{e}) = \epsilon(m|l) \sum_{\mathbf{a}} \prod_{j=1}^{m} \left(t_\alpha(f_j|e_{a_j}) a_\alpha(a_j|a_{j-1},l)\right) \tag{11}$$

The alignment probabilities from Model 2, however, are replaced by a different type of alignment probabilities. Here the probability of alignment a_j for position j has a dependence on the previous alignment a_{j-1} which turns the alignment model into a first order Markov model. The counts for the new a parameter are defined as follows:

$$a_\alpha(i|i',l) = \frac{\sum_{s=1}^{S} c(i|i',l;\mathbf{f}^{(s)},\mathbf{e}^{(s)})}{\sum_{i''}\sum_{s=1}^{S} c(i''|i',l;\mathbf{f}^{(s)},\mathbf{e}^{(s)})}, \tag{12}$$

$$c(i|i',l;\mathbf{f},\mathbf{e}) = \sum_{\mathbf{a}} Pr_\alpha(\mathbf{a}|\mathbf{f},\mathbf{e}) \sum_j \delta(i',a_{j-1})\delta(i,a_j) \quad (13)$$

The definition of the t parameter and corresponding counts remains the same as for Model 1 and 2. Like before we only have to modify the definition of $Pr(\mathbf{a}|\mathbf{f},\mathbf{e})$:

$$Pr_\alpha(\mathbf{a}|\mathbf{f},\mathbf{e}) = \frac{\prod_{j=1}^m t_\alpha(f_j|e_{a_j})a_\alpha(a_j|a_{j-1},l)}{\sum_{\mathbf{a}}\prod_{j=1}^m t_\alpha(f_j|e_{a_j})a_\alpha(a_j|a_{j-1},l)} \quad (14)$$

is replaced by

$$Pr_\alpha(\mathbf{a}|\mathbf{f},\mathbf{e}) = \frac{\prod_{i=1}^m \left(\bar{t}(f_j,e_{a_j})a_\alpha(a_j|a_{j-1},l)\right)}{\sum_{\mathbf{a}}\prod_{j=1}^m \left(\bar{t}(f_j,e_{a_j})a_\alpha(a_j|a_{j-1},l)\right)}. \quad (15)$$

$\bar{t}$ is defined as before for Model 1 and 2.

Here, the alignment probabilities a remain unchanged. For Model 2 we are able to find the symmetrically calculated a parameters just by swapping source and target values. Doing the same for the Markov model would change the interpretation of the alignment probabilities. We would require neighbouring source language words to be aligned only with neighbouring target language words which is to strong an assumption. Nevertheless, their values are still influenced by both models due to the appearance of $\bar{t}$ in the re-estimation.

D. Model 3 and 4

We already mentioned that the parameters specific for Models 3 and 4 are calculated from fractional counts collected over a subset of alignments that have been identified with the help of the Viterbi alignments calculated by Model 2. Therefore it is not necessary to revise the parameter estimation formulas for Model 3 and 4, instead we simply adopt the previous changes made for Model 2. This influences the parameters of Model 3 and 4 indirectly by choosing better informed Viterbi alignments during each iteration.

E. Final Symmetrization

Although the two directed models influence each other between each iteration, the two final alignments produced at the end of the training procedure differ due the restrictions imposed by the models. Alignments are directed and since alignments are functions, there are no one-to-many or many-to-many alignments for the respective directions. There are, however, many-to-one alignments. [1] have proposed a method for the symmetrization of alignment models, which they call *refined symmetrization* and which is reported to have a positive effect on alignment quality.

They first map each directed alignment into a set of alignment points and create a new alignment as the intersection of these two sets. Next, they iteratively add alignment points (i,j) from the union of the two sets to the newly created alignment occurring only in the first alignment or in the second alignment if neither f_j nor e_i has an alignment in the new alignment, or if both of the following conditions hold:

- The alignment (i,j) has a horizontal neighbour$(i-1,j)$, $(i+1,j)$ or a vertical neighbour $(i,j-1)$, $(i,j+1)$ that is already in the new alignment.
- Adding (i,j) to the new alignment does not created alignments with both horizontal and vertical neighbours.

This method is applied as the final step of our computation and will also be applied to the directed alignments created by Giza++ and MGiza++, our baseline systems.

TABLE I
RESULTS FOR THE HLT/NAACL 2003 TEST SET

Alignment Method	Time [m]	Prec [%]	Rec [%]	AER [%]
GIZA++ EN-FR	–	91.19	92.20	8.39
GIZA++ FR-EN	–	91.82	87.96	9.79
GIZA++ REFINED	457	93.24	92.59	7.02
MGIZA++ EN-FR	–	91.19	92.22	8.40
MGIZA++ FR-EN	–	91.84	87.96	9.78
MGIZA++ REFINED	306	93.25	92.60	7.01
SYMGIZA++	332	94.34	94.08	**5.76**

IV. EVALUATION

We compare three systems on the same training and test data: Giza++, MGiza++, and SyMGiza++. For the Giza++ and MGiza++ we run both directed models separately and in parallel and recombine the resulting final alignments with the refined method described in III-E. The tools from the Giza++ family are all run with the following training scheme: $5\times$Model 1, $5\times$HMM Model, $5\times$Model 3 and $5\times$Model 4. All experiments were performed on a test system with 4 CPUs and 8 GB RAM, we plan to increase the number CPUs in the future. Apart from Giza++ all tools make use of all available CPUs, the parallel computation of the alignment model with Giza++ can employ at most two CPUs.

A. Measures and Evaluation Data

The standard metric *Alignment Error Rate* (AER) proposed by [1] is used to evaluate the quality of the introduced input word alignments. AER is calculated as follows:

$$\text{Precision} = \frac{|A\cap P|}{|A|} \qquad \text{Recall} = \frac{|A\cap S|}{|S|}$$
$$\text{AER} = 1 - \frac{|A\cap S| + |A\cap P|}{|A| + |S|} \quad (16)$$

where P is the set of possible alignment points in the reference alignment, S is the set of sure alignments in the reference alignment ($S\subset P$), and A is the evaluated word alignment.

In order to obtain results that can be easily compared with the work summarized in II-B, we evaluated our system on the Canadian Hansards task made available during the HLT-NAACL 2003 workshop on "Building and Using Parallel Texts: Data Driven Machine Translation and Beyond" [10]. The training data comprises 1.1M sentences from the Canadian Hansards proceedings and a separate test set of 447 manually word-aligned sentences provided by [1].

B. Results

Our results — which comprise alignment quality and processing time — are summarized in Tab. I. Processing time is

measured from the beginning of processing till the end of the symmetrization process.

It is not surprising that there are no significant differences between Giza++ and MGiza++ when AER is considered. MGiza++, however, is about 33% faster than the two Giza++ processes run in parallel. MGiza++ is also slightly faster than SyMGiza++. This delay of SymGiza++ is caused by the parameter recombination executed between each model iteration and by the idle time if one directed model has to wait for its counterpart. SyMGiza++ achieves the best AER results with a relative improvement of more than 17% compared to Giza++ and MGiza++.

In Sec. II-B we gave the results for a number of other symmetrization approaches. Although we use the same test set our results are not yet fully comparable to the results of other works. We tried but failed to reproduce the results from [5] using the BerkeleyAligner, for which the authors reported an AER of 4.9%. The results reported by [5] for their base line alignments produced with Giza++, on the other hand, are more or less identical to our results. This requires further investigation and we will give a more comprehensive comparison of our results and the results in the literature in an extended paper to come.

V. Conclusions

We have presented SyMGiza++, a tool that computes symmetric word alignment models with the capability to take advantage of multi-processor systems. Our fairly simple modification to the well-known IBM Models implemented in Giza++ and MGiza++ achieves quite impressive improvements for AER on the standard Canadian Hansards task. Our symmetrized models outperform post-training symmetrization methods. On a four processor system, SyMGiza++ is slightly slower than MGiza++, but significantly faster than Giza++ executed in two parallel processes.

Acknowledgments

This paper is based on research funded by the Polish Ministry of Science and Higher Education (Grant No. 003/R/T00/2008/05).

References

[1] F. J. Och and H. Ney, "A systematic comparison of various statistical alignment models." *Computational Linguistics*, vol. 29, no. 1, pp. 19–51, 2003.

[2] P. F. Brown, V. J. D. Pietra, S. A. D. Pietra, and R. L. Mercer, "The mathematics of statistical machine translation: Parameter estimation," *Computational Linguistics*, vol. 19, no. 2, pp. 263–311, 1993.

[3] S. Vogel, H. Ney, and C. Tillmann, "Hmm-based word alignment in statistical translation," in *Proceedings of ACL*, 1996, pp. 836–841.

[4] R. Zens, E. Matusov, and H. Ney, "Improved word alignment using a symmetric lexicon model," in *Proceedings of ACL-COLING*, 2004, p. 36.

[5] P. Liang, B. Taskar, and D. Klein, "Alignment by agreement," in *Proceedings of ACL-COLING*, 2006, pp. 104–111.

[6] Q. Gao and S. Vogel, "Parallel implementations of word alignment tool," in *Proceedings of SETQA-NLP*, 2008, pp. 49–57.

[7] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *Journal of the Royal Statistcial Society, series B*, vol. 39, no. 1, pp. 1–38, 1977.

[8] Y. Al-Onaizan, J. Curin, M. Jahr, K. Knight, J. Lafferty, I. Melamed, F. Och, D. Purdy, N. Smith, and D. Yarowsky, "Statistical machine translation," JHU workshop, Tech. Rep., 1999. [Online]. Available: citeseer.ist.psu.edu/al-onaizan99statistical.html

[9] E. Matusov, R. Zens, and H. Ney, "Symmetric word alignments for statistical machine translation," in *Proceedings of ACL-COLING*, 2004, pp. 219–225.

[10] R. Mihalcea and T. Pedersen, "An evaluation exercise for word alignment," in *Proceedings of HLT-NAACL*, 2003, pp. 1–10.