Training a Statistical Machine Translation System without GIZA++

Arne Mauser, Evgeny Matusov, Hermann Ney

Lehrstuhl für Informatik VI - Computer Science Department
RWTH Aachen University
Aachen, Germany
{mauser, matusov, ney}@i6.informatik.rwth-aachen.de

Abstract

The IBM Models (Brown et al., 1993) enjoy great popularity in the machine translation community because they offer high quality word alignments and a free implementation is available with the GIZA++ Toolkit (Och and Ney, 2003). Several methods have been developed to overcome the asymmetry of the alignment generated by the IBM Models. A remaining disadvantage, however, is the high model complexity. This paper describes a word alignment training procedure for statistical machine translation that uses a simple and clear statistical model, different from the IBM models. The main idea of the algorithm is to generate a *symmetric* and *monotonic* alignment between the target sentence and a *permutation graph* representing different reorderings of the words in the source sentence. The quality of the generated alignment is shown to be comparable to the standard GIZA++ training in an SMT setup.

1 Introduction

Currently, the majority of statistical machine translation systems is trained using word alignments of parallel corpora generated by the complex IBM Models (Models 3, 4 or 5, see (Brown et al., 1993)) with the GIZA++ Toolkit (Och and Ney, 2003). This paper describes a word alignment training procedure for statistical machine translation that uses a simple and clear statistical model, different from the IBM models.

The novel training method described here produces word alignments for statistical machine translation while simultaneously reordering each source sentence to match the word order in the corresponding target sentence. This reordering has shown to improve translation quality (Kanthak et al., 2005), (Crego et al., 2005) when using reordering in search.

The main idea of the algorithm is to generate a *monotonic* alignment between the target sentence and a *permutation lattice* representing different reorderings of the words in the source sentence. In contrast to GIZA++ alignments, this alignment is *symmetric*, i.e. it allows for many-to-one and one-to-many connections simultaneously. Furthermore, full *coverage* is ensured for source and target sentence. The alignment is determined with a dynamic programming algorithm similar to the Levenshtein algorithm. In addition, the best permutation of the source sentence is selected from the given permutation lattice in the global decision process. This distinguishes the approach presented here from the methods presented in literature, where static word alignment information is used for reordering.

In the following section, we will give a review of the most common statistical alignment models. Section 3 gives a short description of the translation framework using weighted Finite State Transducers (WFST). A new lexicon-based method for reordering in training is introduced in Section 4 and the alignment procedure is described in detail in Section 5. We will present some experimental results in Section 6.2.

2 Statistical Alignment Models

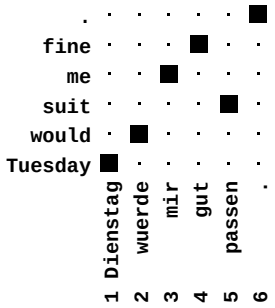
The task of statistical machine translation is to translate an input word sequence $f_1^J = f_1, \ldots, f_J$ in the source language into a target language word sequence $e_1^I = e_1, \ldots, e_I$. Given the source language sequence, we select the target language sequence that maximizes the product of the language model probability $Pr(e_1^I)$ and the translation model probability $Pr(f_1^J|e_1^I)$. The translation model describes the correspondence between the words in the source and the target sequence whereas the language model describes well-formedness of a target word sequence. Introducing a hidden variable, the translation model can be written as:

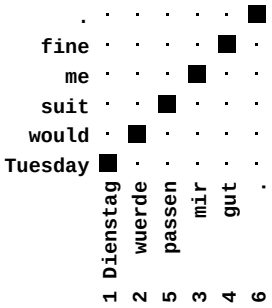
$$Pr(f_1^J|e_1^I) = \sum_{a_1^J} Pr(f_1^J, a_1^J|e_1^I)$$

where a_1^J are called alignments and represent mappings from the source word position j to the target word position $i = a_j$. Alignments are introduced into translation model as a hidden variable, similar to the concept of Hidden Markov Models (HMM) in speech recognition.
The translation probability $Pr(f_1^J, a_1^J|e_1^I)$ can be factorized as follows:

$$\begin{aligned} Pr(f_1^J, a_1^J|e_1^I) &= \prod_{j=1}^{J} Pr(f_j, a_j|f_1^{j-1}, a_1^{j-1}, e_1^I) \\ &= \prod_{j=1}^{J} Pr(a_j|f_1^{j-1}, a_1^{j-1}, e_1^I) \cdot \\ &\quad \cdot Pr(f_j|f_1^{j-1}, a_1^j, e_1^I) \end{aligned}$$

where $Pr(a_j|f_1^{j-1}, a_1^{j-1}, e_1^I)$ is alignment probability and $Pr(f_j|f_1^{j-1}, a_1^j, e_1^I)$ is the lexicon probability.
In all popular translation models IBM-1 to IBM-5 as well as in the HMM translation model, the lexicon probability $Pr(f_j|f_1^{j-1}, a_1^j, e_1^I)$ is approximated with the single-word lexicon probability $p(f_j|e_{a_j})$ which takes into account solely the aligned words f_j and e_{a_j}. The models differ in their definition of the alignment model






```
Dienstag|Tuesday wuerde|would mir|$
gut|$ passen|suit_me_fine .|.
```

```
Dienstag|Tuesday wuerde|would
passen|suit mir|me gut|fine .|.
```

Figure 1: Example for reordering in training. From a given alignment (top) we reorder the source words to generate a monotonic alignment (bottom). Below the alignment matrices the bilanguage representation is shown. It can be seen that by reordering the source sentence, the assignment in the intermediate language improves.

$Pr(a_j|f_1^{j-1}, a_1^{j-1}, e_1^I)$. A detailed description can be found in (Och and Ney, 2003).
Other methods for word alignment often use heuristic models based on cooccurrence counts, such as (Melamed, 2000) or recently (Moore, 2005).

3 Statistical Translation with Finite State Transducers

3.1 Approach

In statistical machine translation, we want to find the target sentence e_1^I with the highest probability for a given source language sentence f_1^J. Here, we formulate the decision rule for maximizing the joint probability of source and target sentence $Pr(f_1^J, e_1^I)$. As in equation (1), the alignment is introduced as a hidden variable $\mathcal{A}$.

$$\hat{e}_1^{\hat{I}} = \operatorname*{argmax}_{I,e_1^I} Pr(f_1^J, e_1^I) \quad (1)$$

$$\cong \operatorname*{argmax}_{I,e_1^I} \max_{\mathcal{A}} Pr(\mathcal{A}) \cdot Pr(f_1^J, e_1^I|\mathcal{A}) \quad (2)$$

For representing the joint events of source words and target words occurring together, we use given alignment information to create *biwords*, i.e. from a pair of aligned source and target word we create a new word separated by a delimiting symbol. Each token in the bilanguage represents the event of the source words $\tilde{f}$ and the target words $\tilde{e}$ being aligned in the training data. For these events, we want to model the joint probability $Pr(f, e)$. The transformation of the whole training corpus in such a way results in a *bilanguage* representation of the training corpus.
On this new corpus, we apply standard language modeling techniques to train smoothed m-gram models.
An m-gram language model can also be represented by a FSA. The histories can be interpreted as the states of the FSA (Allauzen et al., 2003).
In the RWTH system as described in (Kanthak and Ney, 2004), the source language side of the tuples $\tilde{f}$ is always a single source word. The corresponding target tuple $\tilde{e}$ is a sequence of 0 or more target words. For the unique mapping, we require an alignment that is a function of the target words.
In experimental trials it turned out that a 4-gram model yields the best performance for most translation tasks. For better generalization we applied absolute discounting with leaving-one-out parameter estimation.

3.2 Reordering in Training

While most approaches only use the initial alignments to extract bilingual tuples without reordering, the RWTH system first uses alignments which are functions of the source words to reorder the source corpus to reflect the target sentence order. The tuples are then extracted using the reordered corpus.
Given an initial (non-monotonic) alignment, we reorder the source words to form a monotonic alignment. The effect of reordering on the bilanguage representation of the alignment is shown in Figure 1. Reordering helps to extract smaller tuples, leading to a better generalization on unseen data.
In most approaches, alignment and reordering are treated as separate problems. First the best alignment is generated and then the source corpus is reordered given this alignment. The obtained reordering is static and cannot be changed. It is not clear, if the alignment is still optimal with the given reordering. In contrast, with the novel algorithm presented in this work, alignment and reordering are determined in a global decision process.

3.3 Reordering in Translation

In search, alignment information is not available. Reordering, however, is necessary since the training examples were extracted on a reordered source corpus and word order in tuple sequences does not match the original word order. Therefore, we consider permutations of the source sentence in translation. Since arbitrary reorderings are infeasible for larger sentences, we use constrained reordering, for example with the IBM constraints (Berger et al., 1996) or local constraints as described in (Kanthak et al., 2005). They use linear automata to represent sentences and efficiently compute permutation automata on demand.
We follow the formalism from (Matusov et al., 2005), where permutations of the source sentence are represented in a permutation graph. The unpermuted sentence is a lin-

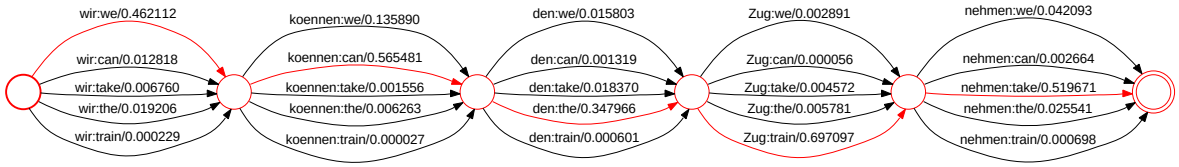


Figure 2: Automaton constructed by composing the linear automaton of the source sentence with the lexicon transducer for the sentence pair. The weights are lexical probabilities.

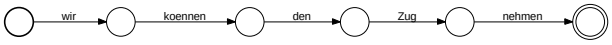


Figure 3: Linear automaton of the source sentence.

ear automaton, having the words as arc labels as shown in Figure 3. From this linear automaton with words as arc labels, permutations can be computed e.g. as described in (Knight and Al-Onaizan, 1998). By using coverage vectors in the state descriptions, we keep track of the covered source words.

4 N-Best Reordering

A static alignment for reordering in training is one way to address the reordering problem for machine translation. The disadvantage is that the reordering is not integrated into the iterative alignment training algorithm. A more flexible approach is to provide multiple reordering hypotheses when determining the alignment.

In training, the source sentence and the corresponding target sentence translation are available and are used to find the best ways of reordering the source sentence to match the target sentence word order. To determine the n-best reorderings for a given source sentence and target sentence we first generate a weighted finite state transducer whose arcs are labeled with the translation probabilities from a given lexicon (a *lexicon transducer*). In this transducer, every source word can be associated with every target word. In the next step, we create a linear unweighted automaton of the source sentence (Figure 3).

A first approach would be to compose the target sentence with the lexicon transducer and extract the best source word sequence. This, however, can result in sequences with incomplete source sentence coverage. Therefore, we follow a different approach. In order to ensure that all source words are covered, we first compose the source sentence with the lexicon transducer.

This composition results in an automaton as shown in Figure 2. In this automaton every path from the initial state to the final state represents a mapping of source words to target words. In this graph, we determine the n-best distinct paths and reorder each path according to the target sentence word order. Of the resulting transducer, we just keep the input labels that now represent the source sentence reorderings.

The resulting n-best list can be condensed in such a way, that reorderings with identical prefixes are combined into one path. With this procedure we can create a permutation graph as shown in Figure 4.

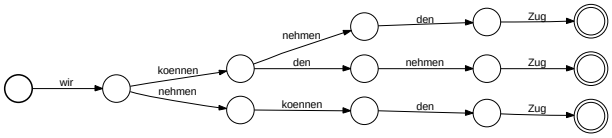


Figure 4: Permutation automaton representing the n-best reorderings. Identical prefixes of different reorderings are joined in order to reduce redundancy.

In general, it can be expected that larger n-best lists yield better results, since they offer more possibilities for reorderings. Adjusting the size of the n-best list allows for a convenient way to balance runtime with permutation space coverage. Unfortunately, as we need to know the target sentence in the process of generating the reorderings, this approach is only applicable in the training phase.

5 Details of the Algorithm

The alignment produced by the algorithm is monotone and symmetric. It consists of a sequence of aligned words, represented by K aligned pairs of source and target word $\mathcal{A} = a_1, a_2, \ldots, a_K$ where $a_k = (i_k, j_k)$ and i_k is the index of the target word aligned to the source word with index j_k. For monotony, we need to ensure that $i_k \leq i_{k+1}$ and $j_k <= j_{k+1}$. The alignment is also contiguous, i.e. no word is left out. Therefore, we have the additional constraints that $i_k \leq 1 + i_{k-1}$ and $j_k \leq 1 + j_{k-1}$. The index pairs $(1, 1)$ and ending in (I, J) are always part of the alignment. The alignment that maximizes the model probability then is:

$$\hat{\mathcal{A}} = \underset{a_k, k=1,\ldots,K}{\operatorname{argmax}} \prod_{k=1}^{K} Pr(f_{j_k}, e_{i_k} | f_{j_1}^{j_k}, e_{i_1}^{i_k})$$

Reading the arc labels of an alignment path gives us the reordering of the source sentence if we just read the source symbols or the bilanguage representation if we join source and target words on each arc. The training procedure involves the following steps:

- For each sentence pair, an alignment lattice L is generated. The alignment lattice contains all possible alignments between the words of the source sentence and the words of the target sentence. The individual alignments are weighted with the corresponding model (e.g. lexical) probability.
- The alignment lattice is composed with the permutation graph of the source sentence. This ensures source

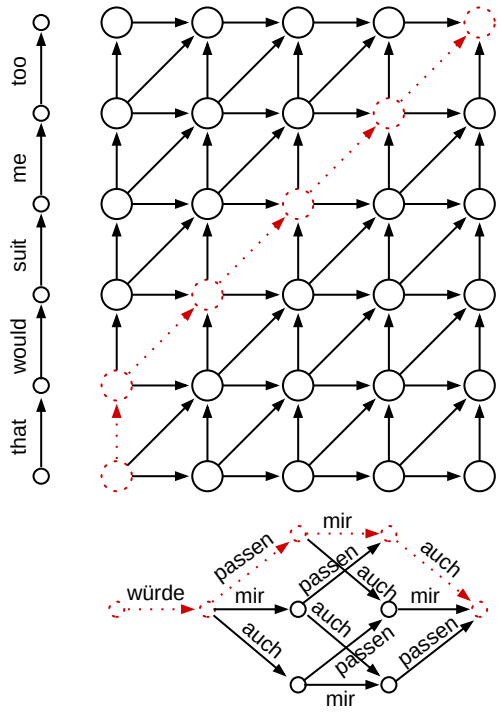


Figure 5: Illustration of the alignment process using a permutation graph and a monotonic alignment lattice.

sentence coverage and restricts the alignments permitted in the alignment lattice. The target sentence is left unpermuted.

- From the composition of permutation graph and alignment lattice, the best path is extracted. This can be seen as simultaneously finding the best path through the alignment lattice and the permutation graph.

Finding the best monotonic alignment between source and target sentence corresponds to finding the best sequence of monotonic transitions. A monotonic path consists of three different types of transitions: A diagonal transition takes place when a source and a target word are assumed to be directly associated with each other. It corresponds to a match/substitution in the Levenshtein algorithm. A horizontal or a vertical movement can be interpreted as an insertion of a source word or a target word, respectively. The probabilities each movement are given by a statistical bilingual lexicon.

Each of the three types of phrasal tuples can be seen as a different movement in the alignment lattice. The regular one-to-one tuple (f, e) corresponds to a diagonal transition in the alignment lattice. The tuples (f, ε) and (ε, e) correspond to a horizontal and vertical movement, respectively. The states in the alignment lattice indicate which source and target words have already been covered.

For permutation, it is possible to use either the constrained reorderings or the n-best reorderings presented in Section 3.3. The general process is illustrated in Figure 5.

The alignment lattice is constructed in such a way, that arcs are labeled with the source and target words associated at the current position. For the target words, this is unambiguous. Since we want to allow for source sequence reorderings any possible source word has to be considered at the horizontal and diagonal transitions in the lattice.

The individual states in the lattice represent the current position in the target sentence (bottom to top) and the number of source words read (left to right).

In the resulting automaton, every path corresponds to an alignment of source sequence and target sequence, where source words are in the order of their aligned target words.

In order to determine the best alignment between source and target sequence, we determine the best path though the composition of alignment lattice and permutation automaton using a general search algorithm.

The lexical probabilities can be modeled taking different context lengths into account. In analogy to the Levenshtein alignment, we first decided to use a zero-order model, where only the current transition is taken into account. This is then extended to a first-order model by the context of the previously aligned words. The models are described in more detail in the following sections.

5.1 Zero-Order Model

For the diagonal transition, the probability is given by the *joint* lexicon probability $p_d(f, e)$ of the source word f and the target word e involved. The horizontal transitions have no direct association with a target word and are weighted by the probability $p_h(f|\varepsilon)$. Similarly, the probabilities for the vertical transitions are $p_v(e|\varepsilon)$.

For initializing the lexicon probabilities of the zero-order model, we decided to use the probabilities obtained from the IBM Model 1. Being simple and easy to obtain, IBM Model 1 also has the advantage of making no assumptions on the word ordering in the languages. This allows us to decide independently of the lexicon, which reordering constraints are suitable for the given task.

Having the named advantages, IBM Model 1 offers a good starting point for the training of other alignment models. For our experiments, we rely on the symmetrized version of the translation lexicon as described in (Zens et al., 2004). Lexica for the source-to-target and target-to-source translation direction are combined log-linearly. This has shown to improve alignment quality. Furthermore, with the models proposed in this section, symmetric alignments will be generated. Therefore, using a symmetric lexicon model for initialization seems to be advisable.

The concept for the implementation of the training uses a recursive problem formulation. The auxiliary quantity $Q(i, j)$ defines the maximum probability for aligning the first i target words and j source words. The recursive formulation of $Q(i, j)$ is

$$\begin{aligned} Q(i,j) = \max \big\{ & Q(i-1,j-1) \cdot p(f_j, e_i), \\ & Q(i,j-1) \quad \cdot p(f_j, \varepsilon), \\ & Q(i-1,j) \quad \cdot p(\varepsilon, e_i) \big\} \end{aligned} \tag{3}$$

In this simple way, the recursive function does not allow for reordering and still assumes a monotonic alignment. To extend the algorithm, we change the source index parameter j of the auxiliary function Q to the coverage vector c. To determine the previous states of coverage vector c, we define

a predecessor set $Pred(c)$, that contains all coverage vectors that are predecessors of c in the permutation graph. If $c' \in Pred(c)$, j is the index of the source word in which c and c differ. The maximization has to be extended to the set of predecessors. Additionally, each arc in the alignment lattice can be associated with the transition probability $p(d)$, $p(h)$ or $p(v)$. This can be used to favor diagonal transitions in the alignment. The complete auxiliary quantity $Q(i, c)$ then is

$$\begin{aligned} Q(i,c) = \max \{ & \\ & \max_{c' \in Pred(c)} Q(i-1, c') \cdot p_d(f_j, e_i) \cdot p(d), \\ & \max_{c' \in Pred(c)} Q(i, c') \quad \cdot p_h(f_j, \varepsilon) \cdot p(h), \\ & Q(i-1, c) \cdot p_v(\varepsilon, e_i) \cdot p(v) \} \end{aligned}$$

Initialization of $Q(i, j)$ is done as in the Levenshtein algorithm. The generated alignment is symmetric, since we keep track of the correspondence of source and target word and allow for one-to-many and many-to-one alignments.

5.2 First-Order Lexicon Model

As a refinement, the lexical probabilities for the non-diagonal transitions can include *first-order* context dependency on the previous source word f' and the previous target word e', i. e. $p_h(f|f', e')$ and $p_v(e|f', e')$, respectively. The previously joint event of a source or target word without a direct association with a word in the other language is now written as a conditional probability of the word, given the previously aligned words.
For initialization, the probabilities are obtained from an alignment generated by the zero-order model.

5.3 Alignment Procedure

The probability distributions are initialized with a simple lexicon model – IBM Model 1. Then, the alignment and reordering is improved in an iterative training procedure. The lexicon probabilities in the next iteration are reestimated using relative frequencies based on the alignments for the whole training corpus in the previous iteration.
The reordering of source words is achieved by considering the permutation lattice. The lattice is processed from left to right. Having processed j source positions, all lattice word alternatives for the position $j + 1$ in the (reordered) source sentence are hypothesized with every diagonal or horizontal transition.

5.4 Implementation

The permutation lattice is computed on-demand under the reordering constraints described in Section 3.3, or we use the n-best reorderings as described in Section 4.
The algorithm was implemented using weighted finite state transducers (WFSTs). This allows for a convenient incorporation of the permutations described above. In contrast to GIZA++, a distributed implementation of this algorithm is straightforward.

6 Experiments

6.1 Experimental Setting

The word alignment and reordering methods presented here were evaluated on three different machine translation tasks with respect to translation quality (as measured by well-established objective error criteria). The proposed training procedure fits especially well with a joint-probability WFST translation system (Kanthak et al., 2005). This system searches for the best translation by composing a lattice that represents constrained reorderings of the source sentence with a WFST representation of a bilingual m-gram model.

Table 3: Corpus Statistics for the German-English Verbmobil Corpus and the Spanish-English Xerox Corpus. The average number of reference translations for the Verbmobil corpus is 3.9 for the test set while the Xerox corpus only had single references

		German	English	Spanish	English
train	Sent.	58073		55761	
	R. Words	519523	549921	752606	665399
	Voc.	7939	4672	11050	7956
test	Sent.	251		1125	
	R. Words	2628	2871	10106	8370
	Voc.	429	402	1215	1132

Experiments were carried out on the German-English Verbmobil task and the Spanish-English Xerox task with the training, development, and test data as shown in Table 3. The Verbmobil task contains dialogues about appointment scheduling and hotel reservation. The Xerox task is the translation of instruction manuals for technical devices.
Despite many approaches, there is still no generally accepted criterion for the evaluation of machine translation output. Different available measures capture different aspects of the translation quality, therefore we will provide several evaluation measures: The word error rate (WER), the position-independent word error rate (PER), the BLEU criterion and the NIST precision measure.
For each of these evaluation measures, multiple references were used for the Verbmobil, but not for the Xerox task.

6.2 Experimental Results

The overall performance of the proposed alignment procedure compares well with the GIZA++ IBM Model 4 training while using a significantly simpler model.
Table 1 shows the translation quality for the German-English Verbmobil task when using this WFST system. The error measures were computed with respect to multiple references. The table shows, that n-best reordering has a slightly worse performance than the IBM constraints. However, for large corpora, alignment with these constraints is not feasible due to the computational complexity. Using the n-best reorderings results in a major speedup of the alignment procedure. For the Verbmobil corpus, the first-order model performs worse than the zero-order model. This can be attributed to data sparseness problems. In fact, most of the bigrams of bilingual tuples are only seen once in training, making probability estimates very unreliable. This problem does not occur to this degree for the zero-order model.
Table 2 shows the evaluation scores of the translation for the Spanish-English Xerox task test corpus. N-best re-

Table 1: Translation performance on the German-English Verbmobil task using the proposed training algorithm. Reordering in translation is performed under the IBM constraints, window size 4.

training model	reordering in training	WER [%]	PER [%]	BLEU [%]	NIST
GIZA++ IBM model 4	static	36.2	27.4	49.1	8.00
zero-order	IBM constraints (window size 4)	35.9	22.8	49.0	7.77
zero-order	N-best reordering ($N = 1000$)	36.5	22.8	47.5	7.75
first-order	IBM constraints (window size 4)	36.1	22.1	48.9	7.82
first-order	N-best reordering ($N = 1000$)	36.9	23.1	47.8	7.74

Table 2: Translation performance on the Spanish-English Xerox task using the proposed training algorithm. Reordering in translation is performed under local constraints, using the same window size as in training.

training model	reordering in training	WER [%]	PER [%]	BLEU [%]	NIST
GIZA IBM model 4	static	35.7	19.7	54.0	8.69
zero-order	local constraints (window size 3)	29.6	21.9	57.3	8.71
first-order	local constraints (window size 3)	29.3	21.6	57.5	8.76

ordering was not performed here, since the language pair only requires local reorderings, which are sufficiently fast to evaluate. In contrast to the Verbmobil corpus, the first-order model gives a slight improvement over the zero-order model. In the domain of the Xerox corpus, technical manuals, expressions are more standardized and less free as in spontaneous speech. Therefore, more reliable bigram counts on the level of bilingual tuples can be obtained.

7 Conclusions

In summary, the main contributions of this work compared to previous approaches are:

- a unified training algorithm for word alignment and source sentence reordering which is much simpler than the algorithms implemented in GIZA++,
- the use of context information in lexical probabilities,
- and a novel approach for reordering in training using the N-best reordering hypotheses.

The translation systems trained with the proposed method perform at least as well or better than the same systems trained with the GIZA++ word alignment toolkit.
Future directions for research might be the use of smoothing methods in the alignment procedure to attenuate data sparseness effects. This would also allow for using higher m-gram models in training.

8 Acknowledgement

This work was in part funded by the European Union under the integrated project TC-STAR – Technology and Corpora for Speech to Speech Translation (IST-2002-FP6-506738).

9 References

C. Allauzen, M. Mohri, and B. Roark. 2003. Generalized algorithms for constructing statistical language models. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 40–47.

A. L. Berger, P. F. Brown, S. A. Della Pietra, V. J. Della Pietra, A. Kehler, and R. L. Mercer. 1996. Language translation apparatus and method using context-based translation models. U.S. patent #5,510,981.

Peter F. Brown, Vincent J. Della Pietra, Stephen A. Della Pietra, and R. L. Mercer. 1993. The mathematics of statistical machine translation: parameter estimation. *Computational Linguistics*, 19(2):263–312, June.

J. M. Crego, J. B. Mari no, and A. de Gispert. 2005. Reordered search, and tuple unfolding for Ngram-based SMT. In *Proceedings of the MT Summit X*, pages 283–289, Phuket, Thailand, September.

S. Kanthak and H. Ney. 2004. FSA: an efficient and flexible c++ toolkit for finite state automata using on-demand computation. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL 2004)*, pages 510–517, Barcelona, Spain, July.

S. Kanthak, D. Vilar, E. Matusov, R. Zens, and H. Ney. 2005. Novel reordering approaches in phrase-based statistical machine translation. In *Proceedings of the ACL Workshop on Building and Using Parallel Texts*, pages 167–174, Ann Arbor, Michigan, June.

K. Knight and Y. Al-Onaizan. 1998. Translation with finite-state devices. In *AMTA '98: Proceedings of the Third Conference of the Association for Machine Translation in the Americas on Machine Translation and the Information Soup*, pages 421–437, London, UK.

E. Matusov, S. Kanthak, and H. Ney. 2005. Efficient statistical machine translation with constrained reordering. In *Proceedings of the 10th annual Conference of European Association of Machine Translation (EAMT 2005)*, pages 181–188, Budapest, Hungary, May.

I. D. Melamed. 2000. Models of translational equivalence among words. *Computational Linguistics*, 26(2):221–249, June.

R. C. Moore. 2005. Association-based bilingual word alignment. In *Proceedings, Workshop on Building and Using Parallel Texts: Data-Driven Machine Translation and Beyond*, pages 1–8, Ann Arbor, Michigan, June.

F. J. Och and H. Ney. 2003. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1), March.

R. Zens, E. Matusov, and H. Ney. 2004. Improved word alignment using a symmetric lexicon model. In *Proceedings of the 20th International Conference on Computational Linguistics (CoLing 2004)*, pages 36–42, Geneva, Switzerland, August.