A Systematic Comparison of Various Statistical Alignment Models

Franz Josef Och*
University of Southern California

Hermann Ney†
RWTH Aachen

We present and compare various methods for computing word alignments using statistical or heuristic models. We consider the five alignment models presented in Brown, Della Pietra, Della Pietra, and Mercer (1993), the hidden Markov alignment model, smoothing techniques, and refinements. These statistical models are compared with two heuristic models based on the Dice coefficient. We present different methods for combining word alignments to perform a symmetrization of directed statistical alignment models. As evaluation criterion, we use the quality of the resulting Viterbi alignment compared to a manually produced reference alignment. We evaluate the models on the German-English Verbmobil task and the French-English Hansards task. We perform a detailed analysis of various design decisions of our statistical alignment system and evaluate these on training corpora of various sizes. An important result is that refined alignment models with a first-order dependence and a fertility model yield significantly better results than simple heuristic models. In the Appendix, we present an efficient training algorithm for the alignment models presented.

1. Introduction

We address in this article the problem of finding the word alignment of a bilingual sentence-aligned corpus by using language-independent statistical methods. There is a vast literature on this topic, and many different systems have been suggested to solve this problem. Our work follows and extends the methods introduced by Brown, Della Pietra, Della Pietra, and Mercer (1993) by using refined statistical models for the translation process. The basic idea of this approach is to develop a model of the translation process with the word alignment as a hidden variable of this process, to apply statistical estimation theory to compute the "optimal" model parameters, and to perform alignment search to compute the best word alignment.

So far, refined statistical alignment models have in general been rarely used. One reason for this is the high complexity of these models, which makes them difficult to understand, implement, and tune. Instead, heuristic models are usually used. In heuristic models, the word alignments are computed by analyzing some association score metric of a link between a source language word and a target language word. These models are relatively easy to implement.

In this article, we focus on consistent statistical alignment models suggested in the literature, but we also describe a heuristic association metric. By providing a detailed description and a systematic evaluation of these alignment models, we give the reader various criteria for deciding which model to use for a given task.

* Information Science Institute (USC/ISI), 4029 Via Marina, Suite 1001, Marina del Rey, CA 90292.
† Lehrstuhl für Informatik VI, Computer Science Department, RWTH Aachen–University of Technology, D-52056 Aachen, Germany.

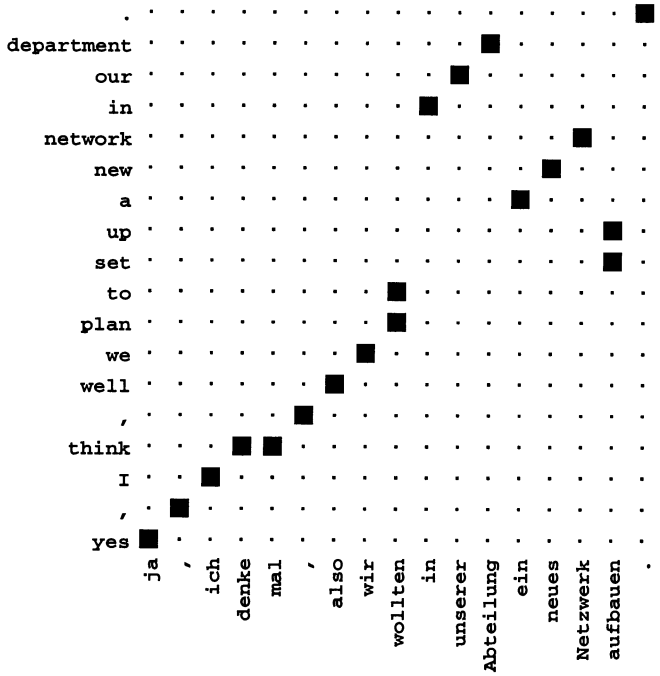

Figure 1
Example of a word alignment (VERBMOBIL task).

We propose to measure the quality of an alignment model by comparing the quality of the most probable alignment, the Viterbi alignment, with a manually produced reference alignment. This has the advantage of enabling an automatic evaluation to be performed. In addition, we shall show that this quality measure is a precise and reliable evaluation criterion that is well suited to guide designing and training statistical alignment models.

The software used to train the statistical alignment models described in this article is publicly available (Och 2000).

1.1 Problem Definition

We follow Brown, Della Pietra, Della Pietra, and Mercer (1993) to define alignment as an object for indicating the corresponding words in a parallel text. Figure 1 shows an example. Very often, it is difficult for a human to judge which words in a given target string correspond to which words in its source string. Especially problematic is the alignment of words within idiomatic expressions, free translations, and missing function words. The problem is that the notion of "correspondence" between words is subjective. It is important to keep this in mind in the evaluation of word alignment quality. We shall deal with this problem in Section 5.

The alignment between two word strings can be quite complicated. Often, an alignment includes effects such as reorderings, omissions, insertions, and word-to-phrase alignments. Therefore, we need a very general representation of alignment. Formally, we use the following definition for alignment in this article. We are given a source (French) string $f_1^J = f_1, \ldots, f_j, \ldots, f_J$ and a target language (English) string $e_1^I = e_1, \ldots, e_i, \ldots, e_I$ that have to be aligned. We define an alignment between the two word strings as a subset of the Cartesian product of the word positions; that is, an

alignment $\mathcal{A}$ is defined as

$$\mathcal{A} \subseteq \{(j, i):\ j = 1, \ldots, J; i = 1, \ldots, I\} \tag{1}$$

Modeling the alignment as an arbitrary relation between source and target language positions is quite general. The development of alignment models that are able to deal with this general representation, however, is hard. Typically, the alignment models presented in the literature impose additional constraints on the alignment representation.

Typically, the alignment representation is restricted in a way such that each source word is assigned to *exactly one* target word. Alignment models restricted in this way are similar to the concept of hidden Markov models (HMMs) in speech recognition. The alignment mapping in such models consists of associations $j \rightarrow i = a_j$ from source position j to target position $i = a_j$. The alignment $a_1^J = a_1, \ldots, a_j, \ldots, a_J$ may contain alignments $a_j = 0$ with the "empty" word e_0 to account for source words that are not aligned with any target word. Constructed in such a way, the alignment is not a relation between source and target language positions, but only a mapping from source to target language positions.

In Melamed (2000), a further simplification is performed that enforces a one-to-one alignment for nonempty words. This means that the alignment mapping a_1^J must be injective for all word positions $a_j > 0$. Note that many translation phenomena cannot be handled using restricted alignment representations such as this one. Especially, methods such as Melamed's are in principle not able to achieve a 100% recall. The problem can be reduced through corpus preprocessing steps that perform grouping and splitting of words.

Some papers report improvements in the alignment quality of statistical methods when linguistic knowledge is used (Ker and Chang 1997; Huang and Choi 2000). In these methods, the linguistic knowledge is used mainly to filter out incorrect alignments. In this work, we shall avoid making explicit assumptions concerning the language used. By avoiding these assumptions, we expect our approach to be applicable to almost every language pair. The only assumptions we make are that the parallel text is segmented into aligned sentences and that the sentences are segmented into words. Obviously, there are additional implicit assumptions in the models that are needed to obtain a good alignment quality. For example, in languages with a very rich morphology, such as Finnish, a trivial segmentation produces a high number of words that occur only once, and every learning method suffers from a significant data sparseness problem.

1.2 Applications

There are numerous applications for word alignments in natural language processing. These applications crucially depend on the quality of the word alignment (Och and Ney 2000; Yarowsky and Wicentowski 2000). An obvious application for word alignment methods is the automatic extraction of bilingual lexica and terminology from corpora (Smadja, McKeown, and Hatzivassiloglou 1996; Melamed 2000).

Statistical alignment models are often the basis of single-word-based statistical machine translation systems (Berger et al. 1994; Wu 1996; Wang and Waibel 1998; Nießen et al. 1998; García-Varea, Casacuberta, and Ney 1998; Och, Ueffing, and Ney 2001; Germann et al. 2001). In addition, these models are the starting point for refined phrase-based statistical (Och and Weber 1998; Och, Tillmann, and Ney 1999) or example-based translation systems (Brown 1997). In such systems, the quality of the machine translation output directly depends on the quality of the initial word alignment (Och and Ney 2000).

Another application of word alignments is in the field of word sense disambiguation (Diab 2000). In Yarowsky, Ngai, and Wicentowski (2001), word alignment is used to transfer text analysis tools such as morphologic analyzers or part-of-speech taggers from a language, such as English, for which many tools already exist to languages for which such resources are scarce.

1.3 Overview

In Section 2, we review various statistical alignment models and heuristic models. We present a new statistical alignment model, a log-linear combination of the best models of Vogel, Ney, and Tillmann (1996) and Brown, Della Pietra, Della Pietra, and Mercer (1993). In Section 3, we describe the training of the alignment models and present a new training schedule that yields significantly better results. In addition, we describe how to deal with overfitting, deficient models, and very small or very large training corpora. In Section 4, we present some heuristic methods for improving alignment quality by performing a symmetrization of word alignments. In Section 5, we describe an evaluation methodology for word alignment methods dealing with the ambiguities associated with the word alignment annotation based on generalized precision and recall measures. In Section 6, we present a systematic comparison of the various statistical alignment models with regard to alignment quality and translation quality. We assess the effect of training corpora of various sizes and the use of a conventional bilingual dictionary. In the literature, it is often claimed that the refined alignment models of Brown, Della Pietra, Della Pietra, and Mercer (1993) are not suitable for small corpora because of data sparseness problems. We show that this is not the case if these models are parametrized suitably. In the Appendix, we describe some methods for efficient training of fertility-based alignment models.

2. Review of Alignment Models

2.1 General Approaches

We distinguish between two general approaches to computing word alignments: statistical alignment models and heuristic models. In the following, we describe both types of models and compare them from a theoretical viewpoint.

The notational convention we employ is as follows. We use the symbol $Pr(\cdot)$ to denote general probability distributions with (almost) no specific assumptions. In contrast, for model-based probability distributions, we use the generic symbol $p(\cdot)$.

2.1.1 Statistical Alignment Models. In statistical machine translation, we try to model the translation probability $Pr(f_1^J \mid e_1^I)$, which describes the relationship between a source language string f_1^J and a target language string e_1^I. In (statistical) alignment models $Pr(f_1^J, a_1^J \mid e_1^I)$, a "hidden" alignment a_1^J is introduced that describes a mapping from a source position j to a target position a_j. The relationship between the translation model and the alignment model is given by

$$Pr(f_1^J \mid e_1^I) = \sum_{a_1^J} Pr(f_1^J, a_1^J \mid e_1^I) \tag{2}$$

The alignment a_1^J may contain alignments $a_j = 0$ with the empty word e_0 to account for source words that are not aligned with any target word.

In general, the statistical model depends on a set of unknown parameters θ that is learned from training data. To express the dependence of the model on the parameter

set, we use the following notation:

$$Pr(f_1^J, a_1^J \mid e_1^I) = p_\theta(f_1^J, a_1^J \mid e_1^I) \tag{3}$$

The art of statistical modeling is to develop specific statistical models that capture the relevant properties of the considered problem domain. In our case, the statistical alignment model has to describe the relationship between a source language string and a target language string adequately.

To train the unknown parameters θ, we are given a parallel training corpus consisting of S sentence pairs $\{(\mathbf{f}_s, \mathbf{e}_s) : s = 1, \ldots, S\}$. For each sentence pair $(\mathbf{f}_s, \mathbf{e}_s)$, the alignment variable is denoted by $\mathbf{a} = a_1^J$. The unknown parameters θ are determined by maximizing the likelihood on the parallel training corpus:

$$\hat{\theta} = \operatorname*{argmax}_\theta \prod_{s=1}^{S} \sum_{\mathbf{a}} p_\theta(\mathbf{f}_s, \mathbf{a} \mid \mathbf{e}_s) \tag{4}$$

Typically, for the kinds of models we describe here, the expectation maximization (EM) algorithm (Dempster, Laird, and Rubin 1977) or some approximate EM algorithm is used to perform this maximization. To avoid a common misunderstanding, however, note that the use of the EM algorithm is not essential for the statistical approach, but only a useful tool for solving this parameter estimation problem.

Although for a given sentence pair there is a large number of alignments, we can always find a best alignment:

$$\hat{a}_1^J = \operatorname*{argmax}_{a_1^J} p_{\hat{\theta}}(f_1^J, a_1^J \mid e_1^I) \tag{5}$$

The alignment $\hat{a}_1^J$ is also called the **Viterbi alignment** of the sentence pair (f_1^J, e_1^I). (For the sake of simplicity, we shall drop the index θ if it is not explicitly needed.)

Later in the article, we evaluate the quality of this Viterbi alignment by comparing it to a manually produced reference alignment. The parameters of the statistical alignment models are optimized with respect to a maximum-likelihood criterion, which is not necessarily directly related to alignment quality. Such an approach, however, requires training with manually defined alignments, which is not done in the research presented in this article. Experimental evidence shows (Section 6) that the statistical alignment models using this parameter estimation technique do indeed obtain a good alignment quality.

In this paper, we use Models 1 through 5 described in Brown, Della Pietra, Della Pietra, and Mercer (1993), the hidden Markov alignment model described in Vogel, Ney, and Tillmann (1996) and Och and Ney (2000), and a new alignment model, which we call Model 6. All these models use a different decomposition of the probability $Pr(f_1^J, a_1^J \mid e_1^I)$.

2.1.2 Heuristic Models. Considerably simpler methods for obtaining word alignments use a function of the similarity between the types of the two languages (Smadja, McKeown, and Hatzivassiloglou 1996; Ker and Chang 1997; Melamed 2000). Frequently, variations of the Dice coefficient (Dice 1945) are used as this similarity function. For each sentence pair, a matrix including the association scores between every word at every position is then obtained:

$$\text{dice}(i, j) = \frac{2 \cdot C(e_i, f_j)}{C(e_i) \cdot C(f_j)} \tag{6}$$

$C(e,f)$ denotes the co-occurrence count of e and f in the parallel training corpus. $C(e)$ and $C(f)$ denote the count of e in the target sentences and the count of f in the source sentences, respectively. From this association score matrix, the word alignment is then obtained by applying suitable heuristics. One method is to choose as alignment $a_j = i$ for position j the word with the largest association score:

$$a_j = \underset{i}{\text{argmax}}\{\text{dice}(i,j)\} \tag{7}$$

A refinement of this method is the **competitive linking algorithm** (Melamed 2000). In a first step, the highest-ranking word position (i,j) is aligned. Then, the corresponding row and column are removed from the association score matrix. This procedure is iteratively repeated until every source or target language word is aligned. The advantage of this approach is that **indirect associations** (i.e., words that co-occur often but are not translations of each other) occur less often. The resulting alignment contains only one-to-one alignments and typically has a higher precision than the heuristic model defined in equation (7).

2.1.3 A Comparison of Statistical Models and Heuristic Models. The main advantage of the heuristic models is their simplicity. They are very easy to implement and understand. Therefore, variants of the heuristic models described above are widely used in the word alignment literature.

One problem with heuristic models is that the use of a specific similarity function seems to be completely arbitrary. The literature contains a large variety of different scoring functions, some including empirically adjusted parameters. As we show in Section 6, the Dice coefficient results in a worse alignment quality than the statistical models.

In our view, the approach of using statistical alignment models is more coherent. The general principle for coming up with an association score between words results from statistical estimation theory, and the parameters of the models are adjusted such that the likelihood of the models on the training corpus is maximized.

2.2 Statistical Alignment Models

2.2.1 Hidden Markov Alignment Model. The alignment model $Pr(f_1^J, a_1^J \mid e_1^I)$ can be structured without loss of generality as follows:

$$Pr(f_1^J, a_1^J \mid e_1^I) = Pr(J \mid e_1^I) \cdot \prod_{j=1}^{J} Pr(f_j, a_j \mid f_1^{j-1}, a_1^{j-1}, e_1^I) \tag{8}$$

$$= Pr(J \mid e_1^I) \cdot \prod_{j=1}^{J} Pr(a_j \mid f_1^{j-1}, a_1^{j-1}, e_1^I) \cdot Pr(f_j \mid f_1^{j-1}, a_1^{j}, e_1^I) \tag{9}$$

Using this decomposition, we obtain three different probabilities: a length probability $Pr(J \mid e_1^I)$, an alignment probability $Pr(a_j \mid f_1^{j-1}, a_1^{j-1}, e_1^I)$ and a lexicon probability $Pr(f_j \mid f_1^{j-1}, a_1^{j}, e_1^I)$. In the hidden Markov alignment model, we assume a first-order dependence for the alignments a_j and that the lexicon probability depends only on the word at position a_j:

$$Pr(a_j \mid f_1^{j-1}, a_1^{j-1}, e_1^I) = p(a_j \mid a_{j-1}, I) \tag{10}$$

$$Pr(f_j \mid f_1^{j-1}, a_1^{j}, e_1^I) = p(f_j \mid e_{a_j}) \tag{11}$$

Later in the article, we describe a refinement with a dependence on $e_{a_{j-1}}$ in the alignment model. Putting everything together and assuming a simple length model $Pr(J \mid e_1^I) = p(J \mid I)$, we obtain the following basic HMM-based decomposition of $p(f_1^J \mid e_1^I)$:

$$p(f_1^J \mid e_1^I) = p(J \mid I) \cdot \sum_{a_1^J} \prod_{j=1}^{J} [p(a_j \mid a_{j-1}, I) \cdot p(f_j \mid e_{a_j})] \tag{12}$$

with the alignment probability $p(i \mid i', I)$ and the translation probability $p(f \mid e)$.

To make the alignment parameters independent of absolute word positions, we assume that the alignment probabilities $p(i \mid i', I)$ depend only on the jump width $(i - i')$. Using a set of non-negative parameters $\{c(i - i')\}$, we can write the alignment probabilities in the form

$$p(i \mid i', I) = \frac{c(i - i')}{\sum_{i''=1}^{I} c(i'' - i')} \tag{13}$$

This form ensures that the alignment probabilities satisfy the normalization constraint for each conditioning word position i', $i' = 1, \ldots, I$. This model is also referred to as a **homogeneous HMM** (Vogel, Ney, and Tillmann 1996). A similar idea was suggested by Dagan, Church, and Gale (1993).

In the original formulation of the hidden Markov alignment model, there is no empty word that generates source words having no directly aligned target word. We introduce the empty word by extending the HMM network by I empty words e_{I+1}^{2I}. The target word e_i has a corresponding empty word e_{i+I} (i.e., the position of the empty word encodes the previously visited target word). We enforce the following constraints on the transitions in the HMM network ($i \leq I$, $i' \leq I$) involving the empty word e_0:[1]

$$p(i + I \mid i', I) = p_0 \cdot \delta(i, i') \tag{14}$$

$$p(i + I \mid i' + I, I) = p_0 \cdot \delta(i, i') \tag{15}$$

$$p(i \mid i' + I, I) = p(i \mid i', I) \tag{16}$$

The parameter p_0 is the probability of a transition to the empty word, which has to be optimized on held-out data. In our experiments, we set $p_0 = 0.2$.

Whereas the HMM is based on first-order dependencies $p(i = a_j \mid a_{j-1}, I)$ for the alignment distribution, Models 1 and 2 use zero-order dependencies $p(i = a_j \mid j, I, J)$:

- Model 1 uses a uniform distribution $p(i \mid j, I, J) = 1/(I + 1)$:

$$Pr(f_1^J, a_1^J \mid e_1^I) = \frac{p(J \mid I)}{(I + 1)^J} \cdot \prod_{j=1}^{J} p(f_j \mid e_{a_j}) \tag{17}$$

  Hence, the word order does not affect the alignment probability.

- In Model 2, we obtain

$$Pr(f_1^J, a_1^J \mid e_1^I) = p(J \mid I) \cdot \prod_{j=1}^{J} [p(a_j \mid j, I, J) \cdot p(f_j \mid e_{a_j})] \tag{18}$$

1 $\delta(i, i')$ is the Kronecker function, which is one if $i = i'$ and zero otherwise.

To reduce the number of alignment parameters, we ignore the dependence on J in the alignment model and use a distribution $p(a_j \mid j, I)$ instead of $p(a_j \mid j, I, J)$.

2.3 Fertility-Based Alignment Models

In the following, we give a short description of the fertility-based alignment models of Brown, Della Pietra, Della Pietra, and Mercer (1993). A gentle introduction can be found in Knight (1999b).

The fertility-based alignment models (Models 3, 4, and 5) (Brown, Della Pietra, Della Pietra, and Mercer 1993) have a significantly more complicated structure than the simple Models 1 and 2. The fertility ϕ_i of a word e_i in position i is defined as the number of aligned source words:

$$\phi_i = \sum_j \delta(a_j, i) \tag{19}$$

The fertility-based alignment models contain a probability $p(\phi \mid e)$ that the target word e is aligned to ϕ words. By including this probability, it is possible to explicitly describe the fact that for instance the German word `übermorgen` produces four English words (`the day after tomorrow`). In particular, the fertility $\phi = 0$ is used for prepositions or articles that have no direct counterpart in the other language.

To describe the fertility-based alignment models in more detail, we introduce, as an alternative alignment representation, the **inverted** alignments, which define a mapping from *target* to *source* positions rather than the other way around. We allow *several* positions in the source language to be covered; that is, we consider alignments B of the form

$$B\colon\ i \rightarrow B_i \subset \{1, \ldots, j, \ldots, J\}. \tag{20}$$

An important constraint for the inverted alignment is that *all* positions of the source sentence must be covered exactly once; that is, the B_i have to form a partition of the set $\{1, \ldots, j, \ldots, J\}$. The number of words $\phi_i = |B_i|$ is the fertility of the word e_i. In the following, B_{ik} refers to the kth element of B_i in ascending order.

The inverted alignments B_0^I are a different way to represent normal alignments a_1^J. The set B_0 contains the positions of all source words that are aligned with the empty word. Fertility-based alignment models use the following decomposition and assumptions:[2]

$$\begin{aligned} Pr(f_1^J, a_1^J \mid e_1^I) &= Pr(f_1^J, B_0^I \mid e_1^I) && (21)\\ &= Pr(B_0 \mid B_1^I) \cdot \prod_{i=1}^{I} Pr(B_i \mid B_1^{i-1}, e_1^I) \cdot Pr(f_1^J \mid B_0^I, e_1^I) && (22)\\ &= p(B_0 \mid B_1^I) \cdot \prod_{i=1}^{I} p(B_i \mid B_{i-1}, e_i) \cdot \prod_{i=0}^{I} \prod_{j \in B_i} p(f_j \mid e_i) && (23) \end{aligned}$$

As might be seen from this equation, we have tacitly assumed that the set B_0 of words aligned with the empty word is generated only after the nonempty positions have

2 The original description of the fertility-based alignment models in Brown, Della Pietra, Della Pietra, and Mercer (1993) includes a more refined derivation of the fertility-based alignment models.

been covered. The distribution $p(B_i \mid B_{i-1}, e_i)$ is different for Models 3, 4, and 5:

- In Model 3, the dependence of B_i on its predecessor B_{i-1} is ignored:

$$p(B_i \mid B_{i-1}, e_i) = p(\phi_i \mid e_i)\ \phi_i! \prod_{j \in B_i} p(j \mid i, J) \tag{24}$$

  We obtain an (inverted) zero-order alignment model $p(j \mid i, J)$.

- In Model 4, every word is dependent on the previous aligned word and on the word classes of the surrounding words. First, we describe the dependence of alignment positions. (The dependence on word classes is for now ignored and will be introduced later.) We have two (inverted) first-order alignment models: $p_{=1}(\Delta j \mid \cdots)$ and $p_{>1}(\Delta j \mid \cdots)$. The difference between this model and the first-order alignment model in the HMM lies in the fact that here we now have a dependence along the j-axis instead of a dependence along the i-axis. The model $p_{=1}(\Delta j \mid \cdots)$ is used to position the first word of a set B_i, and the model $p_{>1}(\Delta j \mid \cdots)$ is used to position the remaining words from left to right:

$$p(B_i \mid B_{i-1}, e_i) = p(\phi_i \mid e_i) \cdot p_{=1}(B_{i1} - \overline{B_{\rho(i)}} \mid \cdots) \prod_{k=2}^{\phi_i} p_{>1}(B_{ik} - B_{i,k-1} \mid \cdots) \tag{25}$$

  The function $i \to i' = \rho(i)$ gives the largest value $i' < i$ for which $|B_{i'}| > 0$. The symbol $\overline{B_{\rho(i)}}$ denotes the average of all elements in $B_{\rho(i)}$.

- Both Model 3 and Model 4 ignore whether or not a source position has been chosen. In addition, probability mass is reserved for source positions outside the sentence boundaries. For both of these reasons, the probabilities of all valid alignments do not sum to unity in these two models. Such models are called **deficient** (Brown, Della Pietra, Della Pietra, and Mercer 1993). Model 5 is a reformulation of Model 4 with a suitably refined alignment model to avoid deficiency. (We omit the specific formula. We note only that the number of alignment parameters for Model 5 is significantly larger than for Model 4.)

Models 3, 4, and 5 define the probability $p(B_0 \mid B_1^I)$ as uniformly distributed for the $\phi_0!$ possibilities given the number of words aligned with the empty word $\phi_0 = |B_0|$. Assuming a binomial distribution for the number of words aligned with the empty word, we obtain the following distribution for B_0:

$$p(B_0 \mid B_1^I) = p\left(\phi_0 \mid \sum_{i=1}^{I} \phi_i\right) \cdot \frac{1}{\phi_0!} \tag{26}$$

$$= \binom{J - \phi_0}{\phi_0} (1 - p_1)^{J - 2\phi_0} p_1^{\phi_0} \cdot \frac{1}{\phi_0!} \tag{27}$$

The free parameter p_1 is associated with the number of words that are aligned with the empty word. There are $\phi_0!$ ways to order the ϕ_0 words produced by the empty word, and hence, the alignment model of the empty word is nondeficient. As we will

see in Section 3.2, this creates problems for Models 3 and 4. Therefore, we modify Models 3 and 4 slightly by replacing $\phi_0!$ in equation (27) with J^{ϕ_0}:

$$p(B_0 \mid B_1^I) = \binom{J-\phi_0}{\phi_0} (1-p_1)^{J-2\phi_0} p_1^{\phi_0} \cdot \frac{1}{J^{\phi_0}} \tag{28}$$

As a result of this modification, the alignment models for both nonempty words and the empty word are deficient.

2.3.1 Model 6. As we shall see, the alignment models with a first-order dependence (HMM, Models 4 and 5) produce significantly better results than the other alignment models. The HMM predicts the distance between subsequent source language positions, whereas Model 4 predicts the distance between subsequent target language positions. This implies that the HMM makes use of locality in the source language, whereas Model 4 makes use of locality in the target language. We expect to achieve better alignment quality by using a model that takes into account both types of dependencies. Therefore, we combine HMM and Model 4 in a log-linear way and call the resulting model Model 6:

$$p_6(\mathbf{f}, \mathbf{a} \mid \mathbf{e}) = \frac{p_4(\mathbf{f}, \mathbf{a} \mid \mathbf{e})^{\alpha} \cdot p_{\mathrm{HMM}}(\mathbf{f}, \mathbf{a} \mid \mathbf{e})}{\sum_{\mathbf{a}', \mathbf{f}'} p_4(\mathbf{f}', \mathbf{a}' \mid \mathbf{e})^{\alpha} \cdot p_{\mathrm{HMM}}(\mathbf{f}', \mathbf{a}' \mid \mathbf{e})} \tag{29}$$

Here, the interpolation parameter α is employed to weigh Model 4 relative to the hidden Markov alignment model. In our experiments, we use Model 4 instead of Model 5, as it is significantly more efficient in training and obtains better results.

In general, we can perform a log-linear combination of several models $p_k(\mathbf{f}, \mathbf{a} \mid \mathbf{e})$, $k = 1, \ldots, K$ by

$$p_6(\mathbf{f}, \mathbf{a} \mid \mathbf{e}) = \frac{\prod_{k=1}^{K} p_k(\mathbf{f}, \mathbf{a} \mid \mathbf{e})^{\alpha_k}}{\sum_{\mathbf{a}', \mathbf{f}'} \prod_{k=1}^{K} p_k(\mathbf{f}', \mathbf{a}' \mid \mathbf{e})^{\alpha_k}} \tag{30}$$

The interpolation parameters α_k are determined in such a way that the alignment quality on held-out data is optimized.

We use a log-linear combination instead of the simpler linear combination because the values of $Pr(\mathbf{f}, \mathbf{a} \mid \mathbf{e})$ typically differ by orders of magnitude for HMM and Model 4. In such a case, we expect the log-linear combination to be better than a linear combination.

2.3.2 Alignment Models Depending on Word Classes. For HMM and Models 4 and 5, it is straightforward to extend the alignment parameters to include a dependence on the word classes of the surrounding words (Och and Ney 2000). In the hidden Markov alignment model, we allow for a dependence of the position a_j on the class of the preceding target word $C(e_{a_{j-1}})$: $p(a_j \mid a_{j-1}, I, C(e_{a_{j-1}}))$. Similarly, we can include dependencies on source and target word classes in Models 4 and 5 (Brown, Della Pietra, Della Pietra, and Mercer 1993). The categorization of the words into classes (here: 50 classes) is performed automatically by using the statistical learning procedure described in Kneser and Ney (1993).

2.3.3 Overview of Models. The main differences among the statistical alignment models lie in the alignment model they employ (zero-order or first-order), the fertility model they employ, and the presence or absence of deficiency. In addition, the models differ with regard to the efficiency of the E-step in the EM algorithm (Section 3.1). Table 1 offers an overview of the properties of the various alignment models.

Table 1
Overview of the alignment models.

Model	Alignment model	Fertility model	E-step	Deficient
Model 1	uniform	no	exact	no
Model 2	zero-order	no	exact	no
HMM	first-order	no	exact	no
Model 3	zero-order	yes	approximative	yes
Model 4	first-order	yes	approximative	yes
Model 5	first-order	yes	approximative	no
Model 6	first-order	yes	approximative	yes

2.4 Computation of the Viterbi Alignment

We now develop an algorithm to compute the Viterbi alignment for each alignment model. Although there exist simple polynomial algorithms for the baseline Models 1 and 2, we are unaware of any efficient algorithm for computing the Viterbi alignment for the fertility-based alignment models.

For Model 2 (also for Model 1 as a special case), we obtain

$$\hat{a}_1^J = \operatorname*{argmax}_{a_1^J} Pr(f_1^J, a_1^J \mid e_1^I) \tag{31}$$

$$= \operatorname*{argmax}_{a_1^J} \left\{ p(J \mid I) \cdot \prod_{j=1}^{J} [p(a_j \mid j, I) \cdot p(f_j \mid e_{a_j})] \right\} \tag{32}$$

$$= \left[\operatorname*{argmax}_{a_j} \{ p(a_j \mid j, I) \cdot p(f_j \mid e_{a_j}) \} \right]_{j=1}^{J} \tag{33}$$

Hence, the maximization over the $(I+1)^J$ different alignments decomposes into J maximizations of $(I+1)$ lexicon probabilities. Similarly, the Viterbi alignment for Model 2 can be computed with a complexity of $O(I \cdot J)$.

Finding the optimal alignment for the HMM is more complicated than for Model 1 or Model 2. Using a dynamic programming approach, it is possible to obtain the Viterbi alignment for the HMM with a complexity of $O(I^2 \cdot J)$ (Vogel, Ney, and Tillmann 1996).

For the refined alignment models, however, namely, Models 3, 4, 5, and 6, maximization over all alignments cannot be efficiently carried out. The corresponding search problem is NP-complete (Knight 1990a). For short sentences, a possible solution could be an A* search algorithm (Och, Ueffing, and Ney 2001). In the work presented here, we use a more efficient greedy search algorithm for the best alignment, as suggested in Brown, Della Pietra, Della Pietra, and Mercer (1993). The basic idea is to compute the Viterbi alignment of a simple model (such as Model 2 or HMM). This alignment is then iteratively improved with respect to the alignment probability of the refined alignment model. (For further details on the greedy search algorithm, see Brown, Della Pietra, Della Pietra, and Mercer [1993].) In the Appendix, we present methods for performing an efficient computation of this pseudo-Viterbi alignment.

3. Training

3.1 EM Algorithm

In this section, we describe our approach to determining the model parameters θ. Every model has a specific set of free parameters. For example, the parameters θ for

Model 4 consist of lexicon, alignment, and fertility parameters:

$$\theta = \{\{p(f \mid e)\}, \{p_{=1}(\Delta j \mid \cdots)\}, \{p_{>1}(\Delta j \mid \cdots)\}, \{p(\phi \mid e)\}, p_1\} \tag{34}$$

To train the model parameters θ, we use a maximum-likelihood approach, as described in equation (4), by applying the EM algorithm (Baum 1972). The different models are trained in succession on the same data; the final parameter values of a simpler model serve as the starting point for a more complex model.

In the E-step of Model 1, the lexicon parameter counts for one sentence pair $(\mathbf{e}, \mathbf{f})$ are calculated:

$$c(f \mid e; \mathbf{e}, \mathbf{f}) = \sum_{\mathbf{e},\mathbf{f}} N(\mathbf{e}, \mathbf{f}) \sum_{\mathbf{a}} Pr(\mathbf{a} \mid \mathbf{e}, \mathbf{f}) \sum_{j} \delta(f, f_j) \delta(e, e_{a_j}) \tag{35}$$

Here, $N(\mathbf{e}, \mathbf{f})$ is the training corpus count of the sentence pair $(\mathbf{f}, \mathbf{e})$. In the M-step, the lexicon parameters are computed:

$$p(f \mid e) = \frac{\sum_s c(f \mid e; \mathbf{f}_s, \mathbf{e}_s)}{\sum_{s,f} c(f \mid e; \mathbf{f}_s, \mathbf{e}_s)} \tag{36}$$

Similarly, the alignment and fertility probabilities can be estimated for all other alignment models (Brown, Della Pietra, Della Pietra, and Mercer 1993). When bootstrapping from a simpler model to a more complex model, the simpler model is used to weigh the alignments, and the counts are accumulated for the parameters of the more complex model.

In principle, the sum over all $(I+1)^J$ alignments has to be calculated in the E-step. Evaluating this sum by explicitly enumerating all alignments would be infeasible. Fortunately, Models 1 and 2 and HMM have a particularly simple mathematical form such that the EM algorithm can be implemented efficiently (i.e., in the E-step, it is possible to efficiently evaluate all alignments). For the HMM, this is referred to as the Baum-Welch algorithm (Baum 1972).

Since we know of no efficient way to avoid the explicit summation over all alignments in the EM algorithm in the fertility-based alignment models, the counts are collected only over a subset of promising alignments. For Models 3 to 6, we perform the count collection only over a small number of good alignments. To keep the training fast, we consider only a small fraction of all alignments. We compare three different methods for using subsets of varying sizes:

- The simplest method is to perform Viterbi training using only the best alignment found. As the Viterbi alignment computation itself is very time consuming for Models 3 to 6, the Viterbi alignment is computed only approximately, using the method described in Brown, Della Pietra, Della Pietra, and Mercer (1993).
- Al-Onaizan et al. (1999) suggest using as well the neighboring alignments of the best alignment found. (For an exact definition of the neighborhood of an alignment, the reader is referred to the Appendix.)
- Brown, Della Pietra, Della Pietra, and Mercer (1993) use an even larger set of alignments, including also the **pegged** alignments, a large set of alignments with a high probability $Pr(f_1^J, a_1^J \mid e_1^I)$. The method for constructing these alignments (Brown, Della Pietra, Della Pietra, and Mercer 1993) guarantees that for each lexical relationship in every sentence pair, at least one alignment is considered.

In Section 6, we show that by using the HMM instead of Model 2 in bootstrapping the fertility-based alignment models, the alignment quality can be significantly improved. In the Appendix, we present an efficient training algorithm of the fertility-based alignment models.

3.2 Is Deficiency a Problem?

When using the EM algorithm on the standard versions of Models 3 and 4, we observe that during the EM iterations more and more words are aligned with the empty word. This results in a poor alignment quality, because too many words are aligned to the empty word. This progressive increase in the number of words aligned with the empty word does not occur when the other alignment models are used. We believe that this is due to the deficiency of Model 3 and Model 4.

The use of the EM algorithm guarantees that the likelihood increases for each iteration. This holds for both deficient and nondeficient models. For deficient models, however, as the amount of deficiency in the model is reduced, the likelihood increases. In Models 3 and 4 as defined in Brown, Della Pietra, Della Pietra, and Mercer (1993), the alignment model for nonempty words is deficient, but the alignment model for the empty word is nondeficient. Hence, the EM algorithm can increase likelihood by simply aligning more and more words with the empty word.[3]

Therefore, we modify Models 3 and 4 slightly, such that the empty word also has a deficient alignment model. The alignment probability is set to $p(j \mid i, J) = 1/J$ for each source word aligned with the empty word. Another remedy, adopted in Och and Ney (2000), is to choose a value for the parameter p_1 of the empty-word fertility and keep it fixed.

3.3 Smoothing

To overcome the problem of overfitting on the training data and to enable the models to cope better with rare words, we smooth the alignment and fertility probabilities. For the alignment probabilities of the HMM (and similarly for Models 4 and 5), we perform an interpolation with a uniform distribution $p(i \mid j, I) = 1/I$ using an interpolation parameter α:

$$p'(a_j \mid a_{j-1}, I) = (1 - \alpha) \cdot p(a_j \mid a_{j-1}, I) + \alpha \cdot \frac{1}{I} \tag{37}$$

For the fertility probabilities, we assume that there is a dependence on the number of letters $g(e)$ of e and estimate a fertility distribution $p(\phi \mid g)$ using the EM algorithm. Typically, longer words have a higher fertility. By making this assumption, the model can learn that the longer words usually have a higher fertility than shorter words.

Using an interpolation parameter β, the fertility distribution is then computed as

$$p'(\phi \mid e) = \left(1 - \frac{\beta}{\beta + n(e)}\right) \cdot p(\phi \mid e) + \frac{\beta}{\beta + n(e)} \cdot p(\phi \mid g(e)) \tag{38}$$

Here, $n(e)$ denotes the frequency of e in the training corpus. This linear interpolation ensures that for frequent words (i.e., $n(e) \gg \beta$), the specific distribution $p(\phi \mid e)$ dominates, and that for rare words (i.e., $n(e) \ll \beta$), the general distribution $p(\phi \mid g(e))$ dominates.

The interpolation parameters α and β are determined in such a way that the alignment quality on held-out data is optimized.

3 This effect did not occur in Brown, Della Pietra, Della Pietra, and Mercer (1993), as Models 3 and 4 were not trained directly.

3.4 Bilingual Dictionary

A conventional bilingual dictionary can be considered an additional knowledge source that can be used in training. We assume that the dictionary is a list of word strings $(\mathbf{e}, \mathbf{f})$. The entries for each language can be a single word or an entire phrase.

To integrate a dictionary into the EM algorithm, we compare two different methods:

- Brown, Della Pietra, Della Pietra, Goldsmith, et al. (1993) developed a multinomial model for the process of constructing a dictionary (by a human lexicographer). By applying suitable simplifications, the method boils down to adding every dictionary entry $(\mathbf{e}, \mathbf{f})$ to the training corpus with an entry-specific count called **effective multiplicity**, expressed as $\mu(\mathbf{e}, \mathbf{f})$:

$$\mu(\mathbf{e}, \mathbf{f}) = \frac{\lambda(\mathbf{e}) \cdot p(\mathbf{f} \mid \mathbf{e})}{1 - e^{\lambda(\mathbf{e}) \cdot p(\mathbf{f}|\mathbf{e})}} \tag{39}$$

  In this section, $\lambda(\mathbf{e})$ is an additional parameter describing the size of the sample that is used to estimate the model $p(\mathbf{f} \mid \mathbf{e})$. This count is then used instead of $N(\mathbf{e}, \mathbf{f})$ in the EM algorithm as shown in equation (35).

- Och and Ney (2000) suggest that the effective multiplicity of a dictionary entry be set to a large value $\mu^+ \gg 1$ if the lexicon entry actually occurs in one of the sentence pairs of the bilingual corpus and to a low value otherwise:

$$\mu(\mathbf{e}, \mathbf{f}) = \begin{cases} \mu^+ & \text{if } \mathbf{e} \text{ and } \mathbf{f} \text{ co-occur} \\ \mu^- & \text{otherwise} \end{cases} \tag{40}$$

  As a result, only dictionary entries that indeed occur in the training corpus have a large effect in training. The motivation behind this is to avoid a deterioration of the alignment as a result of out-of-domain dictionary entries. Every entry in the dictionary that does co-occur in the training corpus can be assumed correct and should therefore obtain a high count. We set $\mu^- = 0$.

4. Symmetrization

In this section, we describe various methods for performing a symmetrization of our directed statistical alignment models by applying a heuristic postprocessing step that combines the alignments in both translation directions (source to target, target to source).

The baseline alignment model does not allow a source word to be aligned with more than one target word. Therefore, lexical correspondences like that of the German compound word *Zahnarzttermin* with the English *dentist's appointment* cause problems, because a single source word must be mapped to two or more target words. Therefore, the resulting Viterbi alignment of the standard alignment models has a systematic loss in recall.

To solve this problem, we perform training in both translation directions (source to target, target to source). As a result, we obtain two alignments a_1^J and b_1^I for each pair of sentences in the training corpus. Let $A_1 = \{(a_j, j) \mid a_j > 0\}$ and $A_2 = \{(i, b_i) \mid b_i > 0\}$ denote the sets of alignments in the two Viterbi alignments. To increase the quality of the alignments, we combine A_1 and A_2 into one alignment matrix A using the

following combination methods:

- Intersection: $A = A_1 \cap A_2$.
- Union: $A = A_1 \cup A_2$.
- Refined method: In a first step, the intersection $A = A_1 \cap A_2$ is determined. The elements of this intersection result from both Viterbi alignments and are therefore very reliable. Then, we extend the alignment A iteratively by adding alignments (i, j) occurring only in the alignment A_1 or in the alignment A_2 if neither f_j nor e_i has an alignment in A, or if both of the following conditions hold:
  - The alignment (i, j) has a horizontal neighbor $(i-1, j)$, $(i+1, j)$ or a vertical neighbor $(i, j-1)$, $(i, j+1)$ that is already in A.
  - The set $A \cup \{(i, j)\}$ does not contain alignments with both horizontal and vertical neighbors.

Obviously, the intersection of the two alignments yields an alignment consisting of only one-to-one alignments with a higher precision and a lower recall than either one separately. The union of the two alignments yields a higher recall and a lower precision of the combined alignment than either one separately. Whether a higher precision or a higher recall is preferred depends on the final application for which the word alignment is intended. In applications such as statistical machine translation (Och, Tillmann, and Ney 1999), a higher recall is more important (Och and Ney 2000), so an alignment union would probably be chosen. In lexicography applications, we might be interested in alignments with a very high precision obtained by performing an alignment intersection.

5. Evaluation Methodology

In the following, we present an annotation scheme for single-word-based alignments and a corresponding evaluation criterion.

It is well known that manually performing a word alignment is a complicated and ambiguous task (Melamed 1998). Therefore, in performing the alignments for the research presented here, we use an annotation scheme that explicitly allows for ambiguous alignments. The persons conducting the annotation are asked to specify alignments of two different kinds: an S (sure) alignment, for alignments that are unambiguous, and a P (possible) alignment, for ambiguous alignments. The P label is used especially to align words within idiomatic expressions and free translations and missing function words ($S \subseteq P$).

The reference alignment thus obtained may contain many-to-one and one-to-many relationships. Figure 2 shows an example of a manually aligned sentence with S and P labels.

The quality of an alignment $A = \{(j, a_j) \mid a_j > 0\}$ is then computed by appropriately redefined precision and recall measures:

$$\text{recall} = \frac{|A \cap S|}{|S|}, \quad \text{precision} = \frac{|A \cap P|}{|A|} \tag{41}$$

and the following alignment error rate (AER), which is derived from the well-known F-measure:

$$\text{AER}(S, P; A) = 1 - \frac{|A \cap S| + |A \cap P|}{|A| + |S|} \tag{42}$$

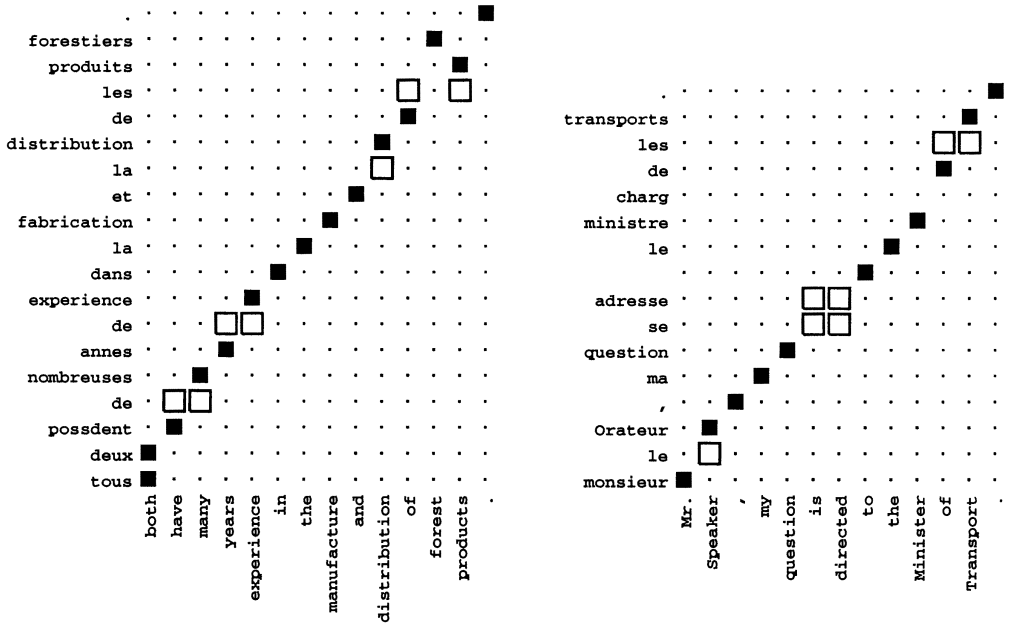

Figure 2
A manual alignment with S (filled squares) and P (unfilled squares) connections.

These definitions of precision, recall and the AER are based on the assumption that a recall error can occur only if an S alignment is not found and a precision error can occur only if the found alignment is not even P.

The set of sentence pairs for which the manual alignment is produced is randomly selected from the training corpus. It should be emphasized that all the training of the models is performed in a completely unsupervised way (i.e., no manual alignments are used). From this point of view, there is no need to have a test corpus separate from the training corpus.

Typically, the annotation is performed by two human annotators, producing sets S_1, P_1, S_2, P_2. To increase the quality of the resulting reference alignment, the annotators are presented with the mutual errors and asked to improve their alignments where possible. (Mutual errors of the two annotators A and B are the errors in the alignment of annotator A if we assume the alignment of annotator B as reference and the errors in the alignment of annotator B if we assume the alignment of annotator A as reference.) From these alignments, we finally generate a reference alignment that contains only those S connections on which both annotators agree and all P connections from both annotators. This can be accomplished by forming the intersection of the sure alignments ($S = S_1 \cap S_2$) and the union of the possible alignments ($P = P_1 \cup P_2$), respectively. By generating the reference alignment in this way, we obtain an alignment error rate of 0 percent when we compare the S alignments of every single annotator with the combined reference alignment.

6. Experiments

We present in this section results of experiments involving the Verbmobil and Hansards tasks. The Verbmobil task (Wahlster 2000) is a (German-English) speech translation task

Table 2
Corpus characteristics of the Verbmobil task.

		German	English
Training corpus	Sentences	34,446	≈ 34K
	Words	329,625	343,076
	Vocabulary	5,936	3,505
	Singletons	2,600	1,305
Bilingual dictionary	Entries	4,404	
	Words	4,758	5,543
Test corpus	Sentences	354	
	Words	3,233	3,109

Table 3
Corpus characteristics of the Hansards task.

		French	English
Training corpus	Sentences	1470K	
	Words	24.33M	22.16M
	Vocabulary	100,269	78,332
	Singletons	40,199	31,319
Bilingual dictionary	Entries	28,701	
	Words	28,702	30,186
Test corpus	Sentences	500	
	Words	8,749	7,946

in the domain of appointment scheduling, travel planning, and hotel reservation. The bilingual sentences used in training are correct transcriptions of spoken dialogues. However, they include spontaneous speech effects such as hesitations, false starts, and ungrammatical phrases. The French-English Hansards task consists of the debates in the Canadian parliament. This task has a very large vocabulary of about 100,000 French words and 80,000 English words.[4]

Statistics for the two corpora are shown in Tables 2 and 3. The number of running words and the vocabularies are based on full-form words and the punctuation marks. We produced smaller training corpora by randomly choosing 500, 2,000 and 8,000 sentences from the Verbmobil task and 500, 8,000, and 128,000 sentences from the Hansards task.

For both tasks, we manually aligned a randomly chosen subset of the training corpus. From this subset of the corpus, the first 100 sentences are used as the development corpus to optimize the model parameters that are not trained via the EM

4 We do not use the Blinker annotated corpus described in Melamed (1998), since the domain is very special (the Bible) and a different annotation methodology is used.

Table 4
Comparison of alignment error rate percentages for various training schemes (Verbmobil task; Dice+C: Dice coefficient with competitive linking).

		Size of training corpus			
Model	Training scheme	0.5K	2K	8K	34K
Dice		28.4	29.2	29.1	29.0
Dice+C		21.5	21.8	20.1	20.4
Model 1	1^5	19.3	19.0	17.8	17.0
Model 2	$1^5 2^5$	27.7	21.0	15.8	13.5
HMM	$1^5 H^5$	19.1	15.4	11.4	9.2
Model 3	$1^5 2^5 3^3$	25.8	18.4	13.4	10.3
	$1^5 H^5 3^3$	18.1	14.3	10.5	8.1
Model 4	$1^5 2^5 3^3 4^3$	23.4	14.6	10.0	7.7
	$1^5 H^5 4^3$	17.3	11.7	9.1	6.5
	$1^5 H^5 3^3 4^3$	16.8	11.7	8.4	6.3
Model 5	$1^5 H^5 4^3 5^3$	17.3	11.4	8.7	6.2
	$1^5 H^5 3^3 4^3 5^3$	16.9	11.8	8.5	5.8
Model 6	$1^5 H^5 4^3 6^3$	17.2	11.3	8.8	6.1
	$1^5 H^5 3^3 4^3 6^3$	16.4	11.7	8.0	5.7

algorithm (e.g., the smoothing parameters). The remaining sentences are used as the test corpus.

The sequence of models used and the number of training iterations used for each model is referred to in the following as the **training scheme**. Our standard training scheme on Verbmobil is $1^5 H^5 3^3 4^3 6^3$. This notation indicates that five iterations of Model 1, five iterations of HMM, three iterations of Model 3, three iterations of Model 4, and three iterations of Model 6 are performed. On Hansards, we use $1^5 H^{10} 3^3 4^3 6^3$. This training scheme typically gives very good results and does not lead to overfitting. We use the slightly modified versions of Model 3 and Model 4 described in Section 3.2 and smooth the fertility and the alignment parameters. In the E-step of the EM algorithm for the fertility-based alignment models, we use the Viterbi alignment and its neighborhood. Unless stated otherwise, no bilingual dictionary is used in training.

6.1 Models and Training Schemes

Tables 4 and 5 compare the alignment quality achieved using various models and training schemes. In general, we observe that the refined models (Models 4, 5, and 6) yield significantly better results than the simple Model 1 or Dice coefficient. Typically, the best results are obtained with Model 6. This holds across a wide range of sizes for the training corpus, from an extremely small training corpus of only 500 sentences up to a training corpus of 1.5 million sentences. The improvement that results from using a larger training corpus is more significant, however, if more refined models are used. Interestingly, even on a tiny corpus of only 500 sentences, alignment error rates under 30% are achieved for all models, and the best models have error rates somewhat under 20%.

We observe that the alignment quality obtained with a specific model heavily depends on the training scheme that is used to bootstrap the model.

Table 5
Comparison of alignment error rate percentages for various training schemes (Hansards task; Dice+C: Dice coefficient with competitive linking).

		Size of training corpus			
Model	Training scheme	0.5K	8K	128K	1.47M
Dice		50.9	43.4	39.6	38.9
Dice+C		46.3	37.6	35.0	34.0
Model 1	1^5	40.6	33.6	28.6	25.9
Model 2	$1^5 2^5$	46.7	29.3	22.0	19.5
HMM	$1^5 H^5$	26.3	23.3	15.0	10.8
Model 3	$1^5 2^5 3^3$	43.6	27.5	20.5	18.0
	$1^5 H^5 3^3$	27.5	22.5	16.6	13.2
Model 4	$1^5 2^5 3^3 4^3$	41.7	25.1	17.3	14.1
	$1^5 H^5 3^3 4^3$	26.1	20.2	13.1	9.4
	$1^5 H^5 4^3$	26.3	21.8	13.3	9.3
Model 5	$1^5 H^5 4^3 5^3$	26.5	21.5	13.7	9.6
	$1^5 H^5 3^3 4^3 5^3$	26.5	20.4	13.4	9.4
Model 6	$1^5 H^5 4^3 6^3$	26.0	21.6	12.8	8.8
	$1^5 H^5 3^3 4^3 6^3$	25.9	20.3	12.5	8.7

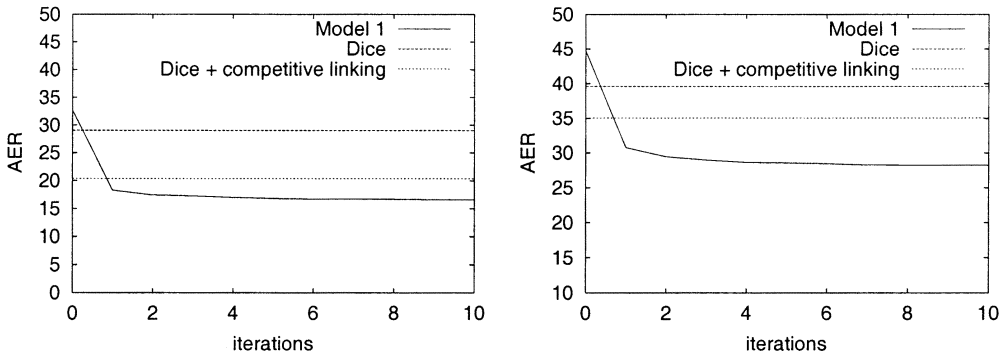

Figure 3
Comparison of alignment error rate (in percent) for Model 1 and Dice coefficient (left: 34K Verbmobil task, right: 128K Hansards task).

6.2 Heuristic Models versus Model 1

We pointed out in Section 2 that from a theoretical viewpoint, the main advantage of statistical alignment models in comparison to heuristic models is the well-founded mathematical theory that underlies their parameter estimation. Tables 4 and 5 show that the statistical alignment models significantly outperform the heuristic Dice coefficient and the heuristic Dice coefficient with competitive linking (Dice+C). Even the simple Model 1 achieves better results than the two Dice coefficient models.

It is instructive to analyze the alignment quality obtained in the EM training of Model 1. Figure 3 shows the alignment quality over the iteration numbers of Model 1. We see that the first iteration of Model 1 achieves significantly worse results than the Dice coefficient, but by only the second iteration, Model 1 gives better results than the Dice coefficient.

Table 6
Effect of using more alignments in training fertility models on alignment error rate (Verbmobil task). Body of table presents error rate percentages.

		Size of training corpus			
Training scheme	Alignment set	0.5K	2K	8K	34K
$1^5H^53^34^36^3$	Viterbi	17.8	12.6	8.6	6.6
	+neighbors	16.4	11.7	8.0	5.7
	+pegging	16.4	11.2	8.2	5.7
$1^52^53^34^35^3$	Viterbi	24.1	16.0	11.6	8.6
	+neighbors	22.9	14.2	9.8	7.6
	+pegging	22.0	13.3	9.7	6.9

Table 7
Effect of using more alignments in training fertility models on alignment error rate (Hansards task). Body of table presents error rate percentages.

		Size of training corpus		
Training scheme	Alignment set	0.5K	8K	128K
$1^5H^{10}3^34^36^3$	Viterbi	25.8	20.3	12.6
	+neighbors	25.9	20.3	12.5
	+pegging	25.8	19.9	12.6
$1^52^53^34^35^3$	Viterbi	41.9	25.1	17.6
	+neighbors	41.7	24.8	16.1
	+pegging	41.2	23.7	15.8

6.3 Model 2 versus HMM

An important result of these experiments is that the hidden Markov alignment model achieves significantly better results than Model 2. We attribute this to the fact that the HMM is a homogeneous first-order alignment model, and such models are able to better represent the locality and monotonicity properties of natural languages. Both models have the important property of allowing an efficient implementation of the EM algorithm (Section 3). On the largest Verbmobil task, the HMM achieves an improvement of 3.8% over Model 2. On the largest Hansards task, the improvement is 8.7%. Interestingly, this advantage continues to hold after bootstrapping more refined models. On Model 4, the improvement is 1.4% and 4.8%, respectively.

We conclude that it is important to bootstrap the refined alignment models with good initial parameters. Obviously, if we use Model 2 for bootstrapping, we eventually obtain a poor local optimum.

6.4 The Number of Alignments in Training

In Tables 6 and 7, we compare the results obtained by using different numbers of alignments in the training of the fertility-based alignment models. We compare the three different approaches described in Section 3: using only the Viterbi alignment, using in addition the neighborhood of the Viterbi alignment, and using the pegged alignments. To reduce the training time, we restrict the number of pegged alignments by using only those in which $Pr(\mathbf{f}, \mathbf{a} \mid \mathbf{e})$ is not much smaller than the probability of the Viterbi alignment. This reduces the training time drastically. For the large Hansards

Table 8
Computing time on the 34K Verbmobil task (on 600 MHz Pentium III machine).

	Seconds per iteration		
Alignment set	Model 3	Model 4	Model 5
Viterbi	48.0	251.0	248.0
+neighbors	101.0	283.0	276.0
+pegging	129.0	3,348.0	3,356.0

Table 9
Effect of smoothing on alignment error rate (Verbmobil task, Model 6). Body of table presents error rate percentages.

	Size of training corpus			
Smoothing method	0.5K	2K	8K	34K
None	19.7	14.9	10.9	8.3
Fertility	18.4	14.3	10.3	8.0
Alignment	16.8	13.2	9.1	6.4
Alignment and fertility	16.4	11.7	8.0	5.7

corpus, however, there still is an unacceptably large training time. Therefore, we report the results for only up to 128,000 training sentences.

The effect of pegging strongly depends on the quality of the starting point used for training the fertility-based alignment models. If we use Model 2 as the starting point, we observe a significant improvement when we use the neighborhood alignments and the pegged alignments. If we use only the Viterbi alignment, the results are significantly worse than using additionally the neighborhood of the Viterbi alignment. If we use HMM as the starting point, we observe a much smaller effect. We conclude that using more alignments in training is a way to avoid a poor local optimum.

Table 8 shows the computing time for performing one iteration of the EM algorithm. Using a larger set of alignments increases the training time for Model 4 and Model 5 significantly. Since using the pegging alignments yields only a moderate improvement in performance, all following results are obtained by using the neighborhood of the Viterbi alignment without pegging.

6.5 Effect of Smoothing

Tables 9 and 10 show the effect on the alignment error rate of smoothing the alignment and fertility probabilities. We observe a significant improvement when we smooth the alignment probabilities and a minor improvement when we smooth the fertility probabilities. An analysis of the alignments shows that smoothing the fertility probabilities significantly reduces the frequently occurring problem of rare words forming "garbage collectors" in that they tend to align with too many words in the other language (Brown, Della Pietra, Della Pietra, Goldsmith, et al. 1993).

Without smoothing, we observe early overfitting: The alignment error rate increases after the second iteration of HMM, as shown in Figure 4. On the Verbmobil task, the best alignment error rate is obtained in the second iteration. On the Hansards task, the best alignment error rate is obtained in the sixth iteration. In iterations subsequent to the second on the Verbmobil task and the sixth on the Hansards task, the alignment error rate increases significantly. With smoothing of the alignment param-

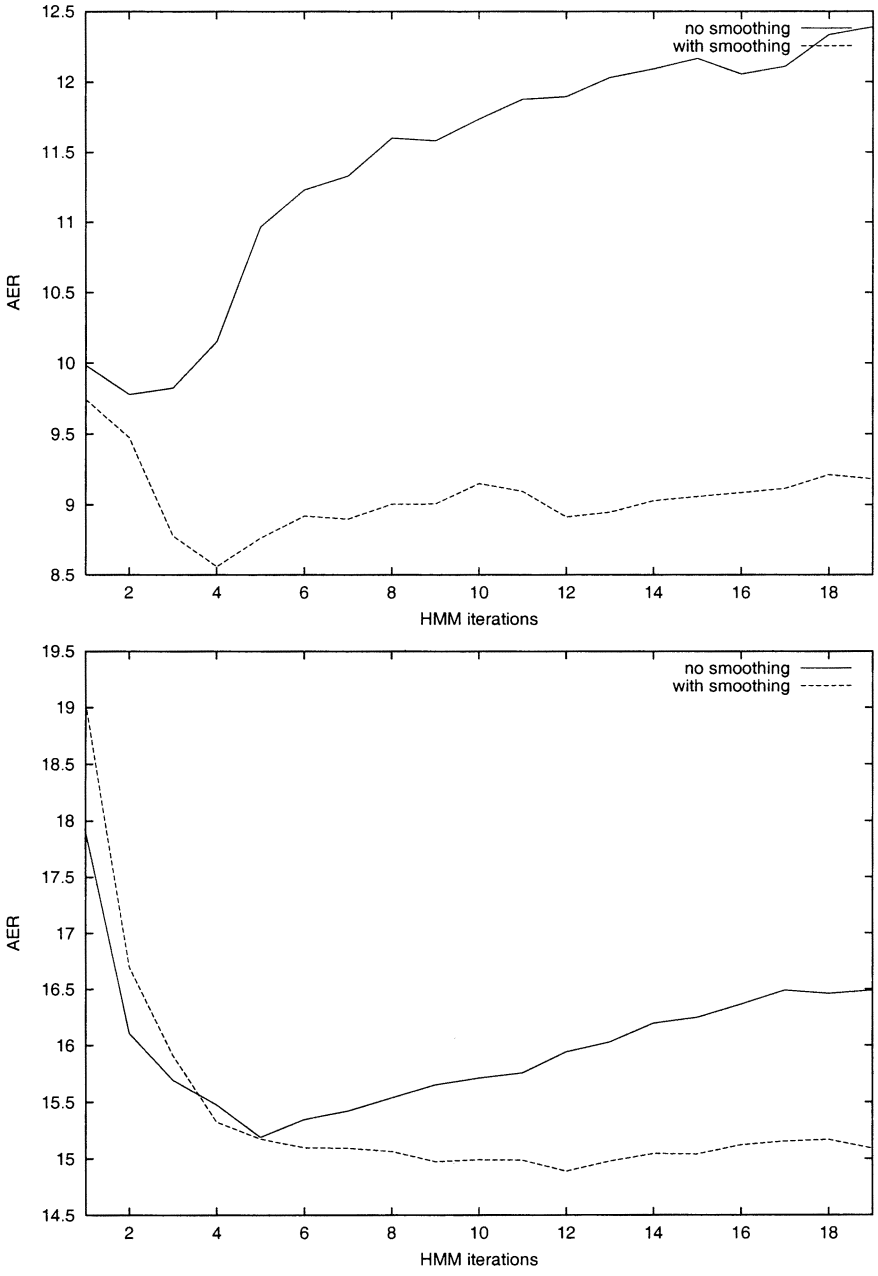

Figure 4
Overfitting on the training data with the hidden Markov alignment model using various smoothing parameters (top: 34K Verbmobil task, bottom: 128K Hansards task).

Table 10
Effect of smoothing on alignment error rate (Hansards task, Model 6). Body of table presents error rate percentages.

	Size of training corpus			
Smoothing method	0.5K	8K	128K	1470K
None	28.6	23.3	13.3	9.5
Fertility	28.3	22.5	12.7	9.3
Alignment	26.5	21.2	13.0	8.9
Alignment and fertility	25.9	20.3	12.5	8.7

Table 11
Effect of word classes on alignment error rate (Verbmobil task). Body of table presents error rate percentages.

	Size of training corpus			
Word classes	0.5K	2K	8K	34K
No	16.5	11.7	8.0	6.3
Yes	16.4	11.7	8.0	5.7

Table 12
Effect of word classes on alignment error rate (Hansards task). Body of table presents error rate percentages.

	Size of training corpus			
Word classes	0.5K	8K	128K	1470K
No	25.5	20.7	12.8	8.9
Yes	25.9	20.3	12.5	8.7

eters, we obtain a lower alignment error rate, overfitting occurs later in the process, and its effect is smaller.

6.6 Alignment Models Depending on Word Classes

Tables 11 and 12 show the effects of including a dependence on word classes in the alignment model, as described in Section 2.3. The word classes are always trained on the same subset of the training corpus as is used for the training of the alignment models. We observe no significant improvement in performance as a result of including dependence on word classes when a small training corpus is used. A possible reason for this lack of improvement is that either the word classes themselves or the resulting large number of alignment parameters cannot be estimated reliably using a small training corpus. When a large training corpus is used, however, there is a clear improvement in performance on both the Verbmobil and the Hansards tasks.

6.7 Using a Conventional Bilingual Dictionary

Tables 13 and 14 show the effect of using a conventional bilingual dictionary in training on the Verbmobil and Hansards tasks, respectively. We compare the two methods for using the dictionary described in Section 3.4. We observe that the method with a fixed

Table 13
Effect of using a conventional dictionary on alignment error rate (Verbmobil task). Body of table presents error rate percentages.

	Size of training corpus			
Bilingual dictionary	0.5K	2K	8K	34K
No	16.4	11.7	8.0	5.7
Yes/μ var.	10.9	9.0	6.9	5.1
Yes/$\mu^+ = 8$	9.7	7.6	6.0	5.1
Yes/$\mu^+ = 16$	10.0	7.8	6.0	4.6
Yes/$\mu^+ = 32$	10.4	8.5	6.4	4.7

Table 14
Effect of using a conventional dictionary on alignment error rate (Hansards task). Body of table presents error rate percentages.

	Size of training corpus			
Bilingual dictionary	0.5K	8K	128K	1470K
No	25.9	20.3	12.5	8.7
Yes/μ var.	23.3	18.3	12.3	8.6
Yes/$\mu^+ = 8$	22.7	18.5	12.2	8.6
Yes/$\mu^+ = 16$	23.1	18.7	12.1	8.6
Yes/$\mu^+ = 32$	24.9	20.2	11.7	8.3

threshold of $\mu^+ = 16$ gives the best results. The method with a varying μ gives worse results, but this method has one fewer parameter to be optimized on held-out data.

On small corpora, there is an improvement of up to 6.7% on the Verbmobil task and 3.2% on the Hansards task, but when a larger training corpus is used, the improvements are reduced to 1.1% and 0.4%, respectively. Interestingly, the amount of the overall improvement contributed by the use of a conventional dictionary is small compared to the improvement achieved through the use of better alignment models.

6.8 Generalized Alignments

In this section, we compare the results obtained using different translation directions and using the symmetrization methods described in Section 4. Tables 15 and 16 show precision, recall, and alignment error rate for the last iteration of Model 6 for both translation directions. In this experiment, we use the conventional dictionary as well. Particularly for the Verbmobil task, with the language pair German-English, we observe that for German as the source language the alignment error rate is much higher than for English as source language. A possible reason for this difference in the alignment error rates is that the baseline alignment representation as a vector a_1^J does not allow German word compounds (which occur frequently) to be aligned with more than one English word.

The effect of merging alignments by forming the intersection, the union, or the refined combination of the Viterbi alignments in both translation directions is shown in Tables 17 and 18. Figure 5 shows the corresponding precision/recall graphs. By using the refined combination, we can increase precision and recall on the Hansards task. The lowest alignment error rate on the Hansards task is obtained by using the intersection

Table 15
Effect of training corpus size and translation direction on precision, recall, and alignment error rate (Verbmobil task + dictionary). All figures are percentages.

	English → German			German → English		
Corpus size	Precision	Recall	AER	Precision	Recall	AER
0.5K	87.6	93.1	10.0	77.9	80.3	21.1
2K	90.5	94.4	7.8	88.1	88.1	11.9
8K	92.7	95.7	6.0	90.2	89.1	10.3
34K	94.6	96.3	4.6	92.5	89.5	8.8

Table 16
Effect of training corpus size and translation direction on precision, recall, and alignment error rate (Hansards task + dictionary). All figures are percentages.

	English → French			French → English		
Corpus size	Precision	Recall	AER	Precision	Recall	AER
0.5K	73.0	83.8	23.1	68.5	79.1	27.8
8K	77.0	88.9	18.7	76.0	88.5	19.5
128K	84.5	93.5	12.1	84.6	93.3	12.2
1470K	89.4	94.7	8.6	89.1	95.2	8.6

Table 17
Effect of alignment combination on precision, recall, and alignment error rate (Verbmobil task + dictionary). All figures are percentages.

	Intersection			Union			Refined method		
Corpus size	Precision	Recall	AER	Precision	Recall	AER	Precision	Recall	AER
0.5K	97.5	76.8	13.6	74.8	96.1	16.9	87.8	92.9	9.9
2K	97.2	85.6	8.6	84.1	96.9	10.6	91.3	94.2	7.4
8K	97.5	86.6	8.0	87.0	97.7	8.5	92.8	96.0	5.8
34K	98.1	87.6	7.2	90.6	98.4	6.0	94.0	96.9	4.7

Table 18
Effect of alignment combination on precision, recall, and alignment error rate (Hansards task + dictionary). All figures are percentages.

	Intersection			Union			Refined method		
Corpus size	Precision	Recall	AER	Precision	Recall	AER	Precision	Recall	AER
0.5K	91.5	71.3	18.7	63.4	91.6	29.0	75.5	84.9	21.1
8K	95.6	82.8	10.6	68.2	94.4	24.2	83.3	90.0	14.2
128K	96.7	90.0	6.3	77.8	96.9	16.1	89.4	94.4	8.7
1470K	96.8	92.3	5.2	84.2	97.6	11.3	91.5	95.5	7.0

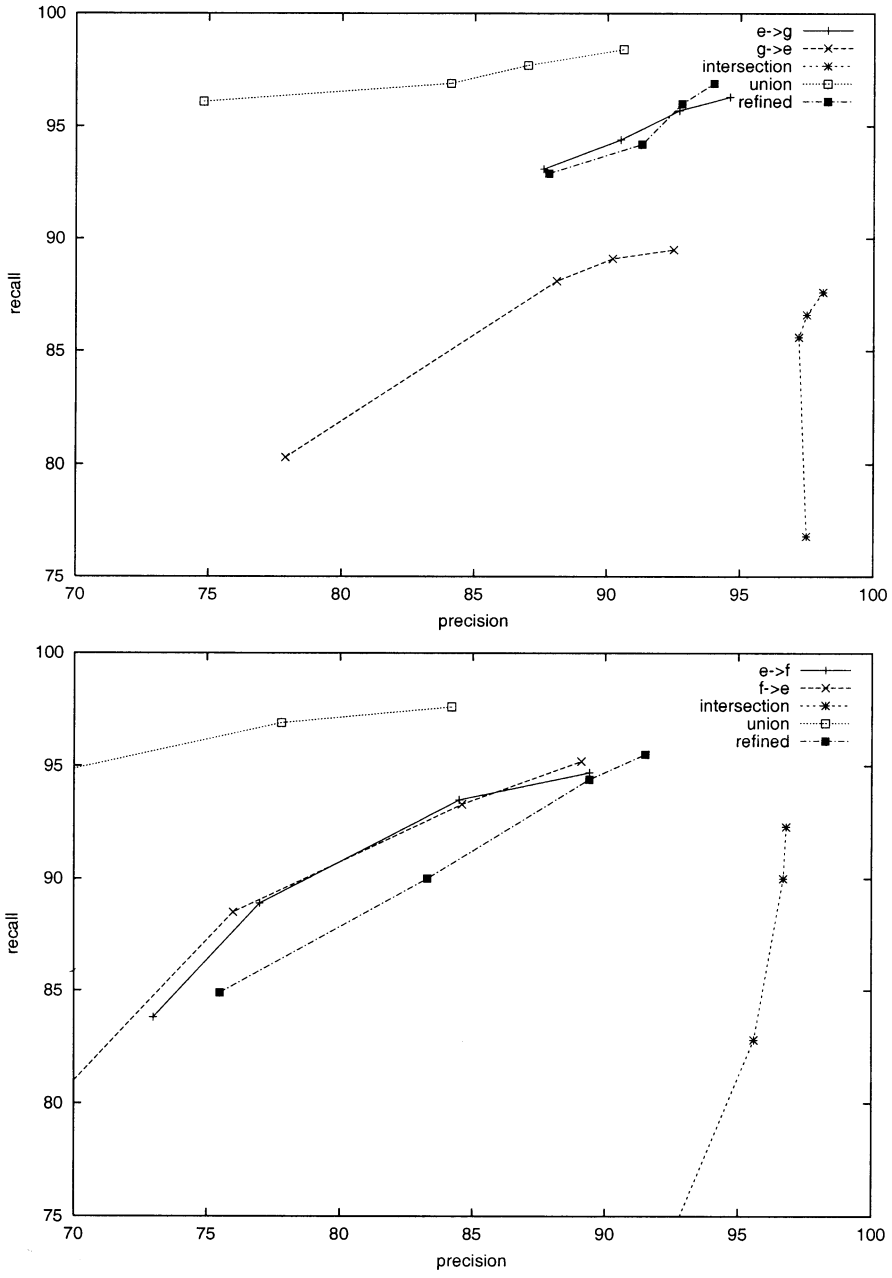

Figure 5
Effect of various symmetrization methods on precision and recall for different training corpus sizes (top: Verbmobil task, bottom: Hansards task).

method. By forming a union or intersection of the alignments, we can obtain very high recall or precision values on both the Hansards task and the Verbmobil task.

6.9 Effect of Alignment Quality on Translation Quality

Alignment models similar to those studied in this article have been used as a starting point for refined phrase-based statistical machine translation systems (Alshawi, Bangalore, and Douglas 1998; Och, Tillmann, and Ney 1999; Ney et al. 2000). In Och and Ney (2000), the overall result of the experimental evaluation has been that an improved alignment quality yields an improved subjective quality of the statistical machine translation system as well.

7. Conclusion

In this article, we have discussed in detail various statistical and heuristic word alignment models and described various modifications and extensions to models known in the literature. We have developed a new statistical alignment model (Model 6) that has yielded the best results among all the models we considered in the experiments we have conducted. We have presented two methods for including a conventional bilingual dictionary in training and described heuristic symmetrization algorithms that combine alignments in both translation directions possible between two languages, producing an alignment with a higher precision, a higher recall, or an improved alignment error rate.

We have suggested measuring the quality of an alignment model using the quality of the Viterbi alignment compared to that achieved in a manually produced reference alignment. This quality measure has the advantage of automatic evaluation. To produce the reference alignment, we have used a refined annotation scheme that reduces the problems and ambiguities associated with the manual construction of a word alignment.

We have performed various experiments to assess the effect of different alignment models, training schemes, and knowledge sources. The key results of these experiments are as follows:

- Statistical alignment models outperform the simple Dice coefficient.
- The best results are obtained with our Model 6. In general, very important ingredients of a good model seem to be a first-order dependence between word positions and a fertility model.
- Smoothing and symmetrization have a significant effect on the alignment quality achieved by a particular model.
- The following methods have only a minor effect on the quality of alignment achieved by a particular model:
  - adding entries of a conventional bilingual dictionary to the training data.
  - making the alignment models dependent on word classes (as in Models 4 and 5).
  - increasing the number of alignments used in the approximation of the EM algorithm for the fertility-based alignment models.

Further improvements in alignments are expected to be produced through the adoption of cognates (Simard, Foster, and Isabelle 1992) and from statistical alignment

models based on word groups rather than single words (Och, Tillmann, and Ney 1999). The use of models that explicitly deal with the hierarchical structures of natural language is very promising (Wu 1996; Yamada and Knight 2001).

We plan to develop structured models for the lexicon, alignment, and fertility probabilities using maximum-entropy models. This is expected to allow an easy integration of more dependencies, such as in a second-order alignment model, without running into the problem of the number of alignment parameters getting unmanageably large.

Furthermore, it will be important to verify the applicability of the statistical alignment models examined in this article to less similar language pairs such as Chinese-English and Japanese-English.

Appendix: Efficient Training of Fertility-Based Alignment Models

In this Appendix, we describe some methods for efficient training of fertility-based alignment models. The core idea is to enumerate only a small subset of good alignments in the E-step of the EM algorithm instead of enumerating all $(I+1)^J$ alignments. This small subset of alignments is the set of neighboring alignments of the best alignment that can be found by a greedy search algorithm. We use two operators to transform alignments: The move operator $m_{[i,j]}(\mathbf{a})$ changes $a_j := i$, and the swap operator $s_{[j_1,j_2]}(\mathbf{a})$ exchanges a_{j_1} and a_{j_2}. The neighborhood $\mathcal{N}(\mathbf{a})$ of an alignment $\mathbf{a}$ is then defined as the set of all alignments that differ by one move or one swap from alignment $\mathbf{a}$:

$$\mathcal{N}(\mathbf{a}) = \{\mathbf{a}' : \exists_{i,j} : \mathbf{a}' = m_{[i,j]}(\mathbf{a}) \vee \exists_{j_1,j_2} : \mathbf{a}' = s_{[j_1,j_2]}(\mathbf{a})\} \tag{43}$$

For one step of the greedy search algorithm, we define the following hill-climbing operator (for Model 3), which yields for an alignment $\mathbf{a}$ the most probable alignment $b(\mathbf{a})$ in the neighborhood $\mathcal{N}(\mathbf{a})$:

$$b(\mathbf{a}) = \operatorname*{argmax}_{\mathbf{a}' \in \mathcal{N}(\mathbf{a})} p_3(\mathbf{a}' \mid \mathbf{e}, \mathbf{f}) \tag{44}$$

Similarly, we define a hill-climbing operator for the other alignment models.

Straightforward Implementation

A straightforward count collection procedure for a sentence pair (**f**,**e**) following the description in Brown, Della Pietra, Della Pietra, and Mercer (1993) is as follows:[5]

1. Calculate the Viterbi alignment of Model 2: $\mathbf{a}_0 := \operatorname{argmax}_{\mathbf{a}} p_2(\mathbf{f}, \mathbf{a} \mid \mathbf{e})$, $n := 0$.
2. While in the neighborhood $\mathcal{N}(\mathbf{a}_n)$ an alignment $\mathbf{a}'$ exists with $p_3(\mathbf{a}' \mid \mathbf{e}, \mathbf{f}) > p_3(\mathbf{a}_n \mid \mathbf{e}, \mathbf{f})$:
   (a) Set $\mathbf{a_{n+1}}$ to the best alignment in the neighborhood.
   (b) $n := n + 1$.
3. Calculate

$$s := \sum_{\mathbf{a} \in \mathcal{N}(\mathbf{a}_n)} Pr(\mathbf{f}, \mathbf{a} \mid \mathbf{e}) \tag{45}$$

5 To simplify the description, we ignore the process known as pegging, which generates a bigger number of alignments considered in training.

4. For each alignment $\mathbf{a}$ in the neighborhood $\mathcal{N}(\mathbf{a}_n)$

(a) Calculate

$$p := Pr(\mathbf{a} \mid \mathbf{e}, \mathbf{f}) \quad (46)$$

$$= \frac{Pr(\mathbf{f}, \mathbf{a} \mid \mathbf{e})}{s} \quad (47)$$

(b) For each $j := 1$ to J: Increase alignment counts

$$c(j \mid a_j, m, l; \mathbf{e}, \mathbf{f}) := c(j \mid a_j, m, l; \mathbf{e}, \mathbf{f}) + p \quad (48)$$

(c) For each $i := 1$ to I: Increase the fertility counts with p:

$$c(\phi_i \mid e_i; \mathbf{e}, \mathbf{f}) := c(\phi_i \mid e_i; \mathbf{e}, \mathbf{f}) + p \quad (49)$$

(d) Increase the counts for p_1:

$$c(1; \mathbf{e}, \mathbf{f}) := c(1; \mathbf{e}, \mathbf{f}) + p \cdot \phi_0 \quad (50)$$

A major part of the time in this procedure is spent on calculating the probability $Pr(\mathbf{a}' \mid \mathbf{e}, \mathbf{f})$ of an alignment $\mathbf{a}'$. In general, this takes about $(I + J)$ operations. Brown, Della Pietra, Della Pietra, and Mercer (1993) describe a method for obtaining $Pr(\mathbf{a}' \mid \mathbf{e}, \mathbf{f})$ incrementally from $Pr(\mathbf{a} \mid \mathbf{e}, \mathbf{f})$ if alignment $\mathbf{a}$ differs only by moves or swaps from alignment $\mathbf{a}'$. This method results in a constant number of operations that is sufficient to calculate the score of a move or the score of a swap.

Refined Implementation: Fast Hill Climbing

Analyzing the training program reveals that most of the time is spent on the computation of the costs of moves and swaps. To reduce the number of operations required in such computation, these values are cached in two matrices. We use one matrix for the scores of a move $a_j := i$:

$$M_{ij} = \frac{Pr(m_{[i,j]}(\mathbf{a}) \mid \mathbf{e}, \mathbf{f})}{Pr(\mathbf{a} \mid \mathbf{e}, \mathbf{f})} \cdot (1 - \delta(a_j, i)) \quad (51)$$

and an additional matrix for the scores of a swap of a_j and $a_{j'}$:

$$S_{jj'} = \begin{cases} \dfrac{Pr(s_{[j,j']}(\mathbf{a}) \mid \mathbf{e}, \mathbf{f})}{Pr(\mathbf{a} \mid \mathbf{e}, \mathbf{f})} \cdot (1 - \delta(a_j, a_{j'})) & \text{if } j < j' \\ 0 & \text{otherwise} \end{cases} \quad (52)$$

During the hill climbing, it is sufficient, after making a move or a swap, to update only those rows or columns in the matrix that are affected by the move or swap. For example, when performing a move $a_j := i$, it is necessary to

- update in matrix M the columns j' with $a_{j'} = a_j$ or $a_{j'} = i$.
- update in matrix M the rows a_j and i.
- update in matrix S the rows and the columns j' with $a_{j'} = a_j$ or $a_{j'} = i$.

Similar updates have to be performed after a swap. In the count collection (step 3), it is possible to use the same matrices as obtained in the last hill-climbing step.

By restricting in this way the number of matrix entries that need to be updated, it is possible to reduce the number of operations in hill climbing by about one order of magnitude.

Refined Implementation: Fast Count Collection

The straightforward algorithm given for performing the count collection has the disadvantage of requiring that all alignments in the neighborhood of alignment $\mathbf{a}$ be enumerated explicitly. In addition, it is necessary to perform a loop over all targets and a loop over all source positions to update the lexicon/alignment and the fertility counts. To perform the count collection in an efficient way, we use the fact that the alignments in the neighborhood $\mathcal{N}(\mathbf{a})$ are very similar. This allows the sharing of many operations in the count collection process.

To efficiently obtain the alignment and lexicon probability counts, we introduce the following auxiliary quantities that use the move and swap matrices that are available after performing the hill climbing described above:

- probability of all alignments in the neighborhood $\mathcal{N}(\mathbf{a})$:

$$Pr(\mathcal{N}(\mathbf{a}) \mid \mathbf{e}, \mathbf{f}) = \sum_{\mathbf{a}' \in \mathcal{N}(\mathbf{a})} Pr(\mathbf{a}' \mid \mathbf{e}, \mathbf{f}) \tag{53}$$

$$= Pr(\mathbf{a} \mid \mathbf{e}, \mathbf{f}) \cdot \left(1 + \sum_{i,j} M_{ij} + \sum_{jj'} S_{jj'}\right) \tag{54}$$

- probability of all alignments in the neighborhood $\mathcal{N}(\mathbf{a})$ that differ in position j from alignment $\mathbf{a}$:

$$Pr(\mathcal{N}_j(\mathbf{a}) \mid \mathbf{e}, \mathbf{f}) = \sum_{\mathbf{a}' \in \mathcal{N}(\mathbf{a})} Pr(\mathbf{a}' \mid \mathbf{e}, \mathbf{f})(1 - \delta(a_j, a'_j)) \tag{55}$$

$$= Pr(\mathbf{a} \mid \mathbf{e}, \mathbf{f}) \left(\sum_{i} M_{ij} + \sum_{j'} (S_{jj'} + S_{j'j})\right) \tag{56}$$

For the alignment counts $c(j \mid i; \mathbf{e}, \mathbf{f})$ and the lexicon counts $c(f \mid e; \mathbf{e}, \mathbf{f})$, we have

$$c(j \mid i; \mathbf{e}, \mathbf{f}) = \begin{cases} Pr(\mathcal{N}(\mathbf{a}) \mid \mathbf{e}, \mathbf{f}) - Pr(\mathcal{N}_j(\mathbf{a}) \mid \mathbf{e}, \mathbf{f}) & \text{if } i = a_j \\ Pr(\mathbf{a} \mid \mathbf{e}, \mathbf{f}) \left(M_{ij} + \sum_{j'} \delta(a_{j'}, i) \cdot (S_{jj'} + S_{j'j})\right) & \text{if } i \neq a_j \end{cases} \tag{57}$$

$$c(f \mid e; \mathbf{e}, \mathbf{f}) = \sum_{i} \sum_{j} c(j \mid i; \mathbf{e}, \mathbf{f}) \cdot \delta(f, f_j) \cdot \delta(e, e_i) \tag{58}$$

To obtain the fertility probability counts and the count for p_1 efficiently, we introduce the following auxiliary quantities:

- probability of all alignments that have an increased fertility for position i:

$$Pr(\mathcal{N}_i^{+1}(\mathbf{a}) \mid \mathbf{e}, \mathbf{f}) = Pr(\mathbf{a} \mid \mathbf{f}, \mathbf{e}) \left(\sum_{j} (1 - \delta(a_j, i)) \cdot M_{ij}\right) \tag{59}$$

- probability of all alignments that have a decreased fertility for position i:

$$Pr(\mathcal{N}_i^{-1}(\mathbf{a}) \mid \mathbf{e}, \mathbf{f}) = Pr(\mathbf{a} \mid \mathbf{e}, \mathbf{f}) \left(\sum_{j} \delta(a_j, i) \sum_{i'} M_{i'j}\right) \tag{60}$$

- probability of all alignments that have an unchanged fertility for position i:

$$Pr(\mathcal{N}_i^{+0}(\mathbf{a}) \mid \mathbf{e}, \mathbf{f}) = Pr(\mathcal{N}(\mathbf{a}) \mid \mathbf{e}, \mathbf{f}) - Pr(\mathcal{N}_i^{+1}(\mathbf{a}) \mid \mathbf{e}, \mathbf{f}) - Pr(\mathcal{N}_i^{-1}(\mathbf{a}) \mid \mathbf{e}, \mathbf{f}) \quad (61)$$

These quantities do not depend on swaps, since a swap does not change the fertilities of an alignment. For the fertility counts, we have:

$$c(\phi \mid e; \mathbf{e}, \mathbf{f}) = \sum_i \delta(e, e_i) \sum_k Pr(\mathcal{N}_i^{+k}(\mathbf{a}) \mid \mathbf{e}, \mathbf{f}) \delta(\phi_i + k, \phi) \quad (62)$$

For p_1, we have:

$$c(1; \mathbf{e}, \mathbf{f}) = \sum_k Pr(\mathcal{N}_0^{+k}(\mathbf{a}) \mid \mathbf{e}, \mathbf{f}) (\phi_0 + k) \quad (63)$$

Using the auxiliary quantities, a count collection algorithm can be formulated that requires about $O(\max(I, J)^2)$ operations. This is one order of magnitude faster than the straightforward algorithm described above. In practice, we observe that the resulting training is 10–20 times faster.

Acknowledgments

This work has been partially supported as part of the Verbmobil project (contract number 01 IV 701 T4) by the German Federal Ministry of Education, Science, Research and Technology and as part of the EuTrans project (project number 30268) by the European Union. In addition, this work has been partially supported by the National Science Foundation under grant no. IIS-9820687 through the 1999 Workshop on Language Engineering, Center for Language and Speech Processing, Johns Hopkins University. All work for this paper was done at RWTH Aachen.

References

Al-Onaizan, Yaser, Jan Curin, Michael Jahr, Kevin Knight, John D. Lafferty, I. Dan Melamed, David Purdy, Franz J. Och, Noah A. Smith, and David Yarowsky. 1999. Statistical machine translation. Final Report, JHU Workshop. Available at http://www.clsp.jhu.edu/ws99/projects/mt/final_report/mt-final-report.ps.

Alshawi, Hiyan, Srinivas Bangalore, and Shona Douglas. 1998. Automatic acquisition of hierarchical transduction models for machine translation. In *COLING–ACL '98: 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics*, volume 1, pages 41–47, Montreal, Canada, August.

Baum, L. E. 1972. An inequality and associated maximization technique in statistical estimation for probabilistic functions of Markov processes. *Inequalities*, 3:1–8.

Berger, Adam L., Peter F. Brown, Stephen A. Della Pietra, Vincent J. Della Pietra, John R. Gillett, John D. Lafferty, Harry Printz, and Lubos Ures. 1994. The Candide system for machine translation. In *Proceedings of the ARPA Workshop on Human Language Technology*, pages 157–162, Plainsboro, New Jersey, March.

Brown, Peter F., Stephen A. Della Pietra, Vincent J. Della Pietra, M. J. Goldsmith, J. Hajic, R. L. Mercer, and S. Mohanty. 1993. But dictionaries are data too. In *Proceedings of the ARPA Workshop on Human Language Technology*, pages 202–205, Plainsboro, New Jersey, March.

Brown, Peter F., Stephen A. Della Pietra, Vincent J. Della Pietra, and R. L. Mercer. 1993. The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics*, 19(2):263–311.

Brown, Ralf D. 1997. Automated dictionary extraction for "knowledge-free" example-based translation. In *Seventh International Conference on Theoretical and Methodological Issues in Machine Translation (TMI-97)*, pages 111–118, Santa Fe, New Mexico, July.

Dagan, Ido, Kenneth W. Church, and

William A. Gale. 1993. Robust bilingual word alignment for machine aided translation. In *Proceedings of the Workshop on Very Large Corpora*, pages 1–8, Columbus, Ohio, June.

Dempster, A. P., N. M. Laird, and D. B. Rubin. 1977. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society, Series B*, 39(1):1–22.

Diab, Mona. 2000. An unsupervised method for multilingual word sense tagging using parallel corpora: A preliminary investigation. In *ACL-2000 Workshop on Word Senses and Multilinguality*, pages 1–9, Hong Kong, October.

Dice, Lee R. 1945. Measures of the amount of ecologic association between species. *Journal of Ecology*, 26:297–302.

García-Varea, Ismael, Francisco Casacuberta, and Hermann Ney. 1998. An iterative, DP-based search algorithm for statistical machine translation. In *Proceedings of the International Conference on Spoken Language Processing (ICSLP'98)*, pages 1235–1238, Sydney, Australia, November.

Germann, Ulrich, Michael Jahr, Kevin Knight, Daniel Marcu, and Kenji Yamada. 2001. Fast decoding and optimal decoding for machine translation. In *Proceedings of the 39th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 228–235, Toulouse, France, July.

Huang, Jin-Xia and Key-Sun Choi. 2000. Chinese-Korean word alignment based on linguistic comparison. In *Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 392–399, Hong Kong, October.

Ker, Sue J. and Jason S. Chang. 1997. A class-based approach to word alignment. *Computational Linguistics*, 23(2):313–343.

Kneser, Reinhard and Hermann Ney. 1993. Improved clustering techniques for class-based statistical language modelling. In *European Conference on Speech Communication and Technology*, pages 973–976, Berlin, Germany, September.

Knight, Kevin. 1999a. Decoding complexity in word-replacement translation models. *Computational Linguistics*, 25(4):607–615.

Knight, Kevin. 1999b. *A Statistical MT Tutorial Workbook*. Available at http://www.isi.edu/natural-language/mt/wkbk.rtf.

Melamed, I. Dan. 1998. Manual annotation of translational equivalence: The blinker project. Technical Report 98-07, Institute for Research in Cognitive Science, Philadelphia.

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

Ney, Hermann, Sonja Nießen, Franz J. Och, Hassan Sawaf, Christoph Tillmann, and Stephan Vogel. 2000. Algorithms for statistical translation of spoken language. *IEEE Transactions on Speech and Audio Processing*, 8(1):24–36.

Nießen, Sonja, Stephan Vogel, Hermann Ney, and Christoph Tillmann. 1998. A DP-based search algorithm for statistical machine translation. In *COLING-ACL '98: 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics*, pages 960–967, Montreal, Canada, August.

Och, Franz J. 2000. Giza++: Training of statistical translation models. Available at http://www-i6.informatik.rwth-aachen.de/~och/software/GIZA++.html.

Och, Franz J. and Hermann Ney. 2000. A comparison of alignment models for statistical machine translation. In *COLING '00: The 18th International Conference on Computational Linguistics*, pages 1086–1090, Saarbrücken, Germany, August.

Och, Franz J., Christoph Tillmann, and Hermann Ney. 1999. Improved alignment models for statistical machine translation. In *Proceedings of the Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora*, pages 20–28, University of Maryland, College Park, June.

Och, Franz J., Nicola Ueffing, and Hermann Ney. 2001. An efficient A* search algorithm for statistical machine translation. In *Data-Driven Machine Translation Workshop*, pages 55–62, Toulouse, France, July.

Och, Franz J. and Hans Weber. 1998. Improving statistical natural language translation with categories and rules. In *COLING-ACL '98: 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics*, pages 985–989, Montreal, Canada, August.

Simard, M., G. Foster, and P. Isabelle. 1992. Using cognates to align sentences in bilingual corpora. In *Fourth International Conference on Theoretical and Methodological Issues in Machine Translation (TMI-92)*, pages 67–81, Montreal, Canada.

Smadja, Frank, Kathleen R. McKeown, and Vasileios Hatzivassiloglou. 1996. Translating collocations for bilingual lexicons: A statistical approach. *Computational Linguistics*, 22(1):1–38.

Vogel, Stephan, Hermann Ney, and Christoph Tillmann. 1996. HMM-based word alignment in statistical translation. In *COLING '96: The 16th International Conference on Computational Linguistics*, pages 836–841, Copenhagen, Denmark, August.

Wahlster, Wolfgang, editor. 2000. *Verbmobil: Foundations of speech-to-speech translations*. Springer Verlag, Berlin.

Wang, Ye-Yi and Alex Waibel. 1998. Fast decoding for statistical machine translation. In *Proceedings of the International Conference on Speech and Language Processing*, pages 1357–1363, Sydney, Australia, November.

Wu, Dekai. 1996. A polynomial-time algorithm for statistical machine translation. In *Proceedings of the 34th Annual Conference of the Association for Computational Linguistics (ACL '96)*, pages 152–158, Santa Cruz, California, June.

Yamada, Kenji and Kevin Knight. 2001. A syntax-based statistical translation model. In *Proceedings of the 39th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 523–530, Toulouse, France, July.

Yarowsky, David, Grace Ngai, and Richard Wicentowski. 2001. Inducing multilingual text analysis tools via robust projection across aligned corpora. In *Human Language Technology Conference*, pages 109–116, San Diego, California, March.

Yarowsky, David and Richard Wicentowski. 2000. Minimally supervised morphological analysis by multimodal alignment. In *Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 207–216, Hong Kong, October.