A DP based Search Algorithm for Statistical Machine Translation

S. Nießen, S. Vogel, H. Ney, and ***C. Tillmann***
Lehrstuhl für Informatik VI
RWTH Aachen – University of Technology
D-52056 Aachen, Germany
Email: niessen@informatik.rwth-aachen.de

Abstract

We introduce a novel search algorithm for statistical machine translation based on dynamic programming (DP). During the search process two statistical knowledge sources are combined: a translation model and a bigram language model. This search algorithm expands hypotheses along the positions of the target string while guaranteeing progressive coverage of the words in the source string. We present experimental results on the Verbmobil task.

1 Introduction

In this paper, we address the problem of finding the most probable target language representation of a given source language string. In our approach, we use a DP based search algorithm which sequentially visits the target string positions while progressively considering the source string words.

The organization of the paper is as follows. After reviewing the statistical approach to machine translation, we first describe the statistical knowledge sources used during the search process. We then present our DP based search algorithm in detail. Finally, experimental results for a bilingual corpus are reported.

1.1 Statistical Machine Translation

In statistical machine translation, the goal of the search strategy can be formulated as follows: We are given a source language ('French') string $f_1^J = f_1 \ldots f_J$, which is to be translated into a target language ('English') string $e_1^I = e_1 \ldots e_I$ with the unknown length I. Every English string is considered as a possible translation for the input string. If we assign a probability $Pr(e_1^I|f_1^J)$ to each pair of strings (e_1^I, f_1^J), then we have to choose the length I_{opt} and the English string $\hat{e}_1^{I_{opt}}$ that maximize $Pr(e_1^I|f_1^J)$ for a given French string f_1^J. According to Bayes decision rule, I_{opt} and $\hat{e}_1^{I_{opt}}$ can be found by

$$\left(I_{opt}, \hat{e}_1^{I_{opt}}\right) = \underset{I,e_1^I}{\operatorname{argmax}} \left\{Pr(e_1^I|f_1^J)\right\} = \underset{I,e_1^I}{\operatorname{argmax}} \left\{Pr(e_1^I) \cdot Pr(f_1^J|e_1^I)\right\} . \quad (1)$$

$Pr(e_1^I)$ is the English language model, whereas $Pr(f_1^J|e_1^I)$ is the string translation model.

The overall architecture of the statistical translation approach is summarized in Fig. 1. In this figure, we already anticipate the fact that we will transform the source strings in a certain manner and that we will countermand these transformations on the produced output strings. This aspect is explained in more detail in Section 3.

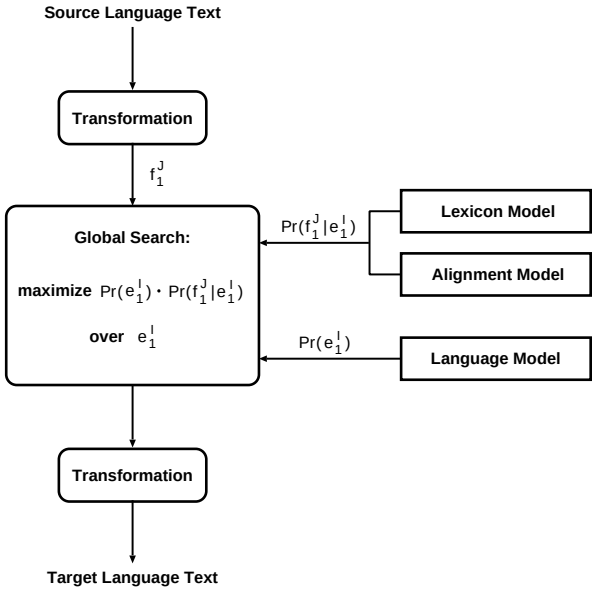


Figure 1: Architecture of the translation approach based on Bayes' decision rule.

The task of statistical machine translation can be subdivided into two fields:

1. the field of modelling, which introduces structures into the probabilistic dependencies and provides methods for estimating the parameters of the models from bilingual corpora;
2. the field of decoding, i.e. finding a search algorithm, which performs the argmax operation in Eq. (1) as efficient as possible.

1.2 Alignment with Mixture Distribution

Several papers have discussed the first issue, especially the problem of word alignments for bilingual corpora (Brown et al., 1993), (Dagan et al., 1993), (Kay and Röscheisen, 1993), (Fung and Church, 1994), (Vogel et al., 1996).

In our search procedure, we use a mixture-based alignment model that slightly differs from the model introduced as *Model 2* in (Brown et al., 1993). It is based on a decomposition of the joint probability for f_1^J into a product of the probabilities for each word f_j:

$$Pr(f_1^J|e_1^I) = p(J|I) \cdot \prod_{j=1}^{J} p(f_j|e_1^I) \ , \qquad (2)$$

where the lengths of the strings are regarded as random variables and modelled by the distribution $p(J|I)$. Now we assume a sort of pairwise interaction between the French word f_j and each English word e_i in e_1^I. These dependencies are captured in the form of a mixture distribution:

$$p(f_j|e_1^I) = \sum_{i=1}^{I} p(i|j,J,I) \cdot p(f_j|e_i) \ . \qquad (3)$$

Inserting this into (2), we get

$$Pr(f_1^J|e_1^I) = p(J|I) \prod_{j=1}^{J} \sum_{i=1}^{I} p(i|j,J,I) \cdot p(f_j|e_i) \quad (4)$$

with the following components: the sentence length probability $p(J|I)$, the mixture alignment probability $p(i|j,J,I)$ and the translation probability $p(f|e)$.

So far, the model allows *all* English words in the target string to contribute to the translation of a French word. This is expressed by the sum over i in Eq. (4). It is reasonable to assume that for each source string position j one position i in the target string dominates this sum. This conforms with the experience, that in most cases a clear word-to-word correspondence between a string and its translation exists. As a consequence, we use the so-called maximum approximation: At each point, only the *best* choice of i is considered for the alignment path:

$$Pr(f_1^J|e_1^I) = p(J|I) \prod_{j=1}^{J} \max_{i \in [1,I]} p(i|j,J,I) \cdot p(f_j|e_i). \quad (5)$$

We can now formulate the criterion to be maximized by a search algorithm:

$$\max_I \left[p(J|I) \max_{e_1^I} \left\{ Pr(e_1^I) \cdot \prod_{j=1}^{J} \max_{i \in [1,I]} [p(i|j,J,I) \cdot p(f_j|e_i)] \right\} \right] . \qquad (6)$$

Because of the problem of data sparseness, we use a parametric model for the alignment probabilities. It assumes that the distance of the positions relative to the diagonal of the (j,i) plane is the dominating factor:

$$p(i|j,J,I) = \frac{r(i - j\frac{I}{J})}{\sum_{i'=1}^{I} r(i' - j\frac{I}{J})} \ . \qquad (7)$$

As described in (Brown et al., 1993), the EM algorithm can be used to estimate the parameters of the model.

1.3 Search in Statistical Machine Translation

In the last few years, there has been a number of papers considering the problem of finding an efficient search procedure (Wu, 1996), (Tillmann et al., 1997a), (Tillmann et al., 1997b), (Wang and Waibel, 1997). All of these approaches use a bigram language model, because they are quite simple and easy-to-use and they have proven their prediction power in stochastic language processing, especially speech recognition. Assuming a bigram language model, we would like to re-formulate Eq. (6) in the following way:

$$\max_I \left[p(J|I) \max_{e_1^I} \left\{ \prod_{j=1}^{J} \max_{i \in [1,I]} [p(e_i|e_{i-1}) \cdot p(i|j,J,I) \cdot p(f_j|e_i)] \right\} \right] . \qquad (8)$$

Any search algorithm tending to perform the maximum operations in Eq. (8) has to guarantee, that the predecessor word e_{i-1} can be determined at the time when a certain word e_i at position i in the target string is under consideration. Different solutions to this problem have been studied.

(Tillmann et al., 1997b) and (Tillmann et al., 1997a) propose a search procedure based on dynamic programming, that examines the source string sequentially. Although it is very efficient in terms of translation speed, it suffers from the drawback of being dependent on the so-called monotonicity constraint: The alignment paths are assumed to be monotone. Hence, the word at position $i-1$ in the target sentence can be determined when the algorithm produces e_i. This approximation corresponds to the assumption of the fundamental similarity of the sentence structures in both languages. In (Tillmann et al., 1997b) text transformations in the source language are used to adapt the word ordering in the source strings to the target language grammar.

(Wang and Waibel, 1997) describe an algorithm based on A^*-search. Here, hypotheses are extended

by adding a word to the end of the target string while considering the source string words in any order. The underlying translation model is *Model 2* from (Brown et al., 1993).

(Wu, 1996) formulates a DP search for stochastic bracketing transduction grammars. The bigram language model is integrated into the algorithm at the point, where two partial parse trees are combined.

2 DP Search

2.1 The Inverted Alignment Model

For our search method, we chose an algorithm which is based on dynamic programming. Compared to an A^*-based algorithm dynamic programming has the fundamental advantage, that solutions of subproblems are stored and can then be re-used in later stages of the search process. However, for the optimization criterion considered here dynamic programming is only suboptimal because the decomposition into independent subproblems is only approximately possible: to prevent the search time of a search algorithm from increasing exponentially with the string lengths and vocabulary sizes, local decisions have to be made at an earlier stage of the optimization process that might turn out to be suboptimal in a later stage but cannot be altered then. As a consequence, the global optimum might be missed in some cases.

The search algorithm we present here combines the advantages of dynamic programming with the search organization along the positions of the target string, which allows the integration of the bigram in a very natural way without restricting the alignment paths to the class of monotone alignments.

The alignment model as described above is defined as a function that assigns exactly one target word to each source word. We introduce a new interpretation of the alignment model: Each position i in e_1^I is assigned a position $b_i = j$ in f_1^J. Fig. 2 illustrates the possible transitions in this inverted model.

At each position i of e_1^I, each word of the target language vocabulary can be inserted. In addition, the *fertility* l must be chosen: A position i and the word e_i at this position are considered to correspond to a sequence of words $f_{b_i+1-l}^{b_i}$ in f_1^J. In most cases, the optimal fertility is 1. It is also possible, that a word e_i has fertility 0, which means that there is no directly corresponding word in the source string. We call this a *skip*, because the position i is skipped in the alignment path.

Using a bigram language model, Eq. (9) specifies the modified search criterion for our algorithm. Here as above, we assume the maximum approximation to be valid.

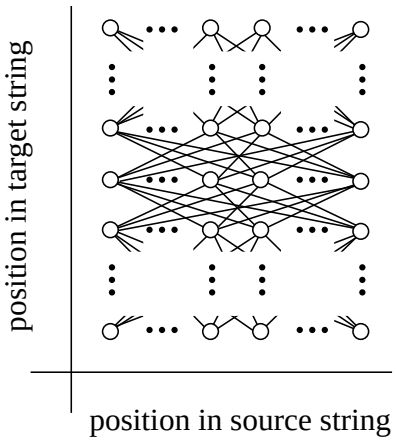


Figure 2: Transitions in the inverted model.

$$\max_I \left[p(J|I) \max_{e_1^I} \prod_{i=1}^{I} \left[p(e_i|e_{i-1}) \cdot \max_{j,l} \prod_{\tilde{\jmath}=j-l+1}^{j} \{ p(i|\tilde{\jmath}, J, I) \cdot p(f_{\tilde{\jmath}}|e_i) \} \right] \right] \quad (9)$$

For better legibility, we regard the second product in Eq. (9) to be equal to 1, if $l = 0$. It should be stressed that the pair (I, e_1^I) optimizing Eq. (9) is not guaranteed to be also optimal in terms of the original criterion (6).

2.2 Basic Problem: Position Coverage

A closer look at Eq. (9) reveals the most important problem of the search organization along the target string positions: It is not guaranteed, that all the words in the source string are considered. In other words we have to force the algorithm to cover all input string positions. Different strategies to solve this problem are possible: For example, we can introduce a reward for covering a position, which has not yet been covered. Or a penalty can be imposed for each position without correspondence in the target string.

In preliminary experiments, we found that the most promising method to satisfy the position coverage constraint is the introduction of an additional parameter into the recursion formula for DP. In the following, we will explain this method in detail.

2.3 Recursion Formula for DP

In the DP formalism, the search process is described recursively. Assuming a total length I of the target string, $Q_I(c, i, j, e)$ is the probability of the best partial path ending in the coordinates i in e_1^I and j in f_1^J, if the last word e_i is e and if c positions in the source string have been covered.

This quantity is defined recursively. Leaving a word e_i without any assignment (*skip*) is the easiest case:

$$Q_I^{\mathrm{S}}(c,i,j,e) = \max_{e'} \{p(e|e') Q_I(c,i-1,j,e')\} \ .$$

Note that it is not necessary to maximize over the predecessor positions j': This maximization is subsumed by the maximization over the positions on the next level, as can easily be proved.

In the original criterion (6), each position j in the source string is aligned to exactly one target string position i. Hence, if i is assigned to l subsequent positions in f_1^J, we want to verify that none of these positions has already been covered: We define a control function v which returns 1 if the above constraint is satisfied and 0 otherwise. Then we can write:

$$\begin{aligned} Q_I^{\mathrm{n}}(c,i,j,e) = \max_{l>0} \Bigg[& \prod_{\tilde{\jmath}=j-l+1}^{j} \{p(i|\tilde{\jmath},J,I) \cdot p(f_{\tilde{\jmath}}|e_i)\} \cdot \\ & \max_{e'} \{p(e|e') \cdot \\ & \max_{j'} \left[Q_I(c-l,i-1,j',e') \cdot v(c,l,j',j,e')\right]\} \Bigg] \ , \end{aligned}$$

We now have to find the maximum:

$$Q_I(c,i,j,e) = \max \left\{Q_I^{\mathrm{S}}(c,i,j,e), Q_I^{\mathrm{n}}(c,i,j,e)\right\} \ .$$

The decisions made during the dynamic programming process (choices of l, j' and e') are stored for recovering the whole translation hypothesis.

The best translation hypothesis can be found by optimizing the target string length I and requiring the number of covered positions to be equal to the source string length J:

$$\max_I \left\{ p(J|I) \cdot \max_{j,e} Q_I(J,I,j,e) \right\} \ . \qquad (10)$$

2.4 Acceleration Techniques

The time complexity of the translation method as described above is

$$O(I_{\max}^2 \cdot J^3 \cdot |\mathcal{E}|^2) \ ,$$

where $|\mathcal{E}|$ is the size of the target language vocabulary $\mathcal{E}$. Some refinements of this algorithm have been implemented to increase the translation speed.

1. We can expect the progression of the source string coverage to be roughly proportional to the progression of the translation procedure along the target string. So it is legitimate to define a minimal and maximal coverage for each level i:

$$c_{\min}(i) = \left\lfloor i\frac{J}{I} \right\rfloor - r, \ c_{\max}(i) = \left\lceil i\frac{J}{I} \right\rceil + r \ ,$$

where r is a constant integer number. In preliminary experiments we found that we could set r to 3 without any loss in translation accuracy. This reduces the time complexity by a factor J.

2. Optimizing the target string length as formulated in Eq. (10) requires the dynamic programming procedure to start all over again for each I. If we assume the dependence of the alignment probabilities $p(i|j,J,I)$ on I to be negligible, we can renormalize them by using an estimated target string length $\hat{I}$ and use $p(i|j,J,\hat{I})$. Now we can produce one translation e_1^I at each level $i = I$ without restarting the whole process:

$$\max_{j,e} Q_{\hat{I}}(J,I,j,e) \ . \qquad (11)$$

For $\hat{I}$ we choose: $\hat{I} = \hat{I}(J) = J \cdot \frac{\mu_I}{\mu_J}$, where μ_I and μ_J denote the average lengths of the target and source strings, respectively.
This approximation is partly undone by what we call *rescoring*: For each translation hypothesis e_1^I with length I, we compute the "true" score $\tilde{Q}(I)$ by searching the best inverted alignment given e_1^I and f_1^J and evaluating the probabilities along this alignment. Hence, we finally find the best translation via Eq. (12):

$$\max_I \left\{ p(J|I) \cdot \tilde{Q}(I) \right\} \ . \qquad (12)$$

The time complexity for this additional step is negligible, since there is no optimization over the English words, which is the dominant factor in the overall time complexity

$$O(I_{\max} \cdot J^2 \cdot |\mathcal{E}|^2) \ .$$

3. We introduced two thresholds:

   θ_L: If e' is the predecessor word of e and e is not aligned to the source string ("skip"), then $p(e|e')$ must be higher than θ_L.

   θ_T: A word e can only be associated with a source language word f, if $p(f|e)$ is higher than θ_T.

   This restricts the optimization over the target language vocabulary to a relatively small set of candidate words. The resulting time complexity is

$$O(I_{\max} \cdot J^2 \cdot |\mathcal{E}|) \ .$$

4. When searching for the best partial path to a gridpoint $G = (c,i,j,e)$, we can sort the arcs leading to G in a specific manner that allows us to stop the computation whenever it becomes clear that no better partial path to G exists. The effect of this measure depends on the quality of the used models; in preliminary experiments we observed a speed-up factor of about 3.5.

3 Experiments

The search algorithm suggested in this paper was tested on the Verbmobil Corpus. The results of preliminary tests on a small automatically generated Corpus (Amengual et al., 1996) were quite promising and encouraged us to apply our search algorithm to a more realistic task.

The Verbmobil Corpus consists of spontaneously spoken dialogs in the domain of appointment scheduling (Wahlster, 1993). German source sentences are translated into English. In Table 1 the characteristics of the training and test sets are summarized. The vocabularies include category labels for dates, proper names, numbers, times, names of places and spellings. The model parameters were trained on 16 296 sentence pairs, where names etc. had been replaced by the appropriate labels.

Table 1: Training and test conditions of the Verbmobil task.

Words in Vocabulary	
German	4 498
English	2 958
Number of Sentences	
in Training Corpus	16 296
in Test Corpus	150

Given the vocabulary sizes, it becomes quite obvious that the lexicon probabilities $p(f|e)$ can not be trained sufficiently on only 16 296 sentence pairs. The fact that about 40% of the words in the lexicon are seen only once in training illustrates this. To improve the lexicon probabilities, we interpolated them with lexicon probabilities $p_M(f|e)$ manually created from a German-English dictionary:

$$p_M(f|e) = \begin{cases} \frac{1}{N_e} & \text{if } (e,f) \text{ is in the dictionary} \\ 0 & \text{otherwise} \end{cases} ,$$

where N_e is the number of German words listed as translations of the English word e. The two lexica were combined by linear interpolation with the interpolation parameter λ. For our first experiments, we set λ to 0.5.

The test corpus consisted of 150 sentences, for which sample translations exist. The labels were translated separately: First, the test sentences were preprocessed in order to replace words or groups of words by the correct category label. Then, our search algorithm translated the transformed sentences. In the last step, a simple rule-based algorithm replaced the category labels by the translations of the original words.

We used a bigram language model for the English language. Its perplexity on the corpus of transformed sample translations (i.e. after labelling) was 13.8.

In preliminary evaluations, optimal values for the thresholds θ_L and θ_T had been determined and kept fixed during the experiments.

As an automatic and easy-to-use measure of the translation performance, the Levenshtein distance between the produced translations and the sample translations was calculated. The translation results are summarized in Table 2.

Table 2: Word error rates on the Verbmobil Corpus: insertions (INS), deletions (DEL) and total rate of word errors (WER) before (BL) and after (AL) rule-based translation of the labels.

before / after	Error Rates (%)		
	INS	DEL	WER
BL	7.3	18.4	45.0
AL	7.6	17.3	39.6

(Tillmann et al., 1997a) report a word error rate of 51.8% on similar data.

Although the Levenshtein distance has the great advantage to be automatically computable, we have to keep in mind, that it depends fundamentally on the choice of the sample translation. For example, each of the expressions "thanks", "thank you" and "thank you very much" is a legitimate translation of the German "danke schön", but when calculating the Levenshtein distance to a sample translation, at least two of them will produce word errors. The more words the vocabulary contains, the more important will be the problem of synonyms.

This is why we also asked five experts to classify independently the produced translations into three categories, being the same as in (Wang and Waibel, 1997):

Correct translations are grammatical and convey the same meaning as the input.

Acceptable translations convey the same meaning but with small grammatical mistakes or they convey most but not the entire meaning of the input.

Incorrect translations are ungrammatical or convey little meaningful information or the information is different from the input.

Examples for each category are given in Table 3. Table 4 shows the statistics of the translation performance. When different judgements existed for one sentence, the majority vote was accepted. For the calculation of the subjective sentence error rate (SSER), translations from the second category counted as "half-correct".

When evaluating the performance of a statistical machine translator, we would like to distinguish errors due to the weakness of the underlying models

Table 3: Examples of Correct (C), Acceptable (A), and Incorrect (I) translations on Verbmobil. The source language is German and the target language is English.

C	Input:	Äh neunter März bin ich in Köln.
	Output:	I am in Cologne on the ninth of March.
	Input:	Habe ich mir notiert.
	Output:	I have noted that.
A	Input:	Samstag und Februar sind gut , aber der siebzehnte wäre besser.
	Output:	Saturday and February are quite but better the seventeenth.
	Input:	Ich könnte erst eigentlich jetzt wieder dann November vorschlagen. Ab zweiten November.
	Output:	I could actually coming back November then. Suggest beginning the second of November.
I	Input:	Ja, also mit Dienstag und mittwochs und so hätte ich Zeit, aber Montag kommen wir hier nicht weg aus Kiel.
	Output:	Yes, and including on Tuesday and Wednesday as well, I have time on Monday but we will come to be away from Kiel.
	Input:	Dann fahren wir da los.
	Output:	We go out.

Table 4: Subjective evaluation of the translation performance on Verbmobil: number of sentences evaluated as Correct (C), Acceptable (A) or Incorrect (I). For the total percentage of non-correct translations (SSER), the "acceptable" translations are counted as half-errors.

Total	Correct	Acceptable	Incorrect	SSER
150	61	45	44	44.3%

from *search errors*, occuring whenever the search algorithm misses a translation hypothesis with a higher score. Unfortunately, we can never be sure that a search error *does not occur*, because we do not know whether or not there is another string with an even higher score than the produced output.

Nevertheless, it is quite interesting to compare the score of the algorithm's output and the score of the sample translation in such cases in which the output is not correct (it is classified as "acceptable" or "incorrect").

The original value to be maximized by the search algorithm (see Eq. (6)) is the score as defined by the underlying models and described by Eq. (13).

$$Pr(e_1^I) \cdot p(J|I) \prod_{j=1}^{J} \max_{i \in [1,I]} [p(i|j,J,I) \cdot p(f_j|e_i)] \ . \quad (13)$$

We calculated this score for the sample translations as well as for the automatically generated translations. Table 5 shows the result of the comparison. In most cases, the incorrect outputs have higher scores than the sample translations, which leads to the conclusion that the improvement of the models (stronger language model for the target language, better translation model and especially more training data) will have a strong impact on the quality of the produced translations. The other cases, i. e. those in which the models prefer the sample translations to the produced output, might be due to the difference of the original search criterion (6) and the criterion (9), which is the basis of our search algorithm. The approximation made by the introduction of the parameters θ_T and θ_L is an additional reason for search errors.

Table 5: Comparison: Score of Reference Translation $\mathbf{e}$ and Translator Output $\mathbf{e}'$ for "acceptable" translations (A) and "incorrect" translations (I). For the total number of non-correct translations (T), the "acceptable" translations are counted as half-errors.

	A	I	T	%
Total number	45	44	66.5	100.0
$Score(\mathbf{e}) \geq Score(\mathbf{e}')$	11	13	18.5	27.8
$Score(\mathbf{e}) < Score(\mathbf{e}')$	34	31	48.0	72.2

As far as we know, only two recent papers have dealt with decoding problem for machine translation systems that use translation models based on hidden alignments without a monotonicity constraint: (Berger et al., 1994) and (Wang and Waibel, 1997). The former uses data sets that differ significantly from the Verbmobil task and hence, the reported results cannot be compared to ours. The latter presents experiments carried out on a corpus com-

parable to our test data in terms of vocabulary sizes, domain and number of test sentences. The authors report a subjective sentence error rate which is in the same range as ours. An exact comparison is only possible if exactly the same training and testing data are used and if all the details of the search algorithms are considered.

4 Conclusion and Future Work

In this paper, we have presented a new search algorithm for statistical machine translation. First experiments prove its applicability to realistic and complex tasks such as spontaneously spoken dialogs.

Several improvements to our algorithm are planned, the most important one being the implementation of pruning methods (Ney et al., 1992). Pruning methods have already been used successfully in machine translation (Tillmann et al., 1997a). The first question to be answered in this context is how to make two different hypotheses H_1 and H_2 comparable: Even if they cover the same *number* of source string words, they might cover *different* words, especially words that are not equally difficult to translate, which corresponds to higher or lower translation probability estimates. To cope with this problem, we will introduce a heuristic for the estimation of the cost of translating the remaining source words. This is similar to the heuristics in A^*-search.

(Vogel et al., 1996) report better perplexity results on the Verbmobil Corpus with their HMM-based alignment model in comparison to *Model 2* of (Brown et al., 1993). For such a model, however, the new interpretation of the alignments becomes essential: We cannot adopt the estimates for the alignment probabilities $p(i|i', I)$. Instead, we have to re-calculate them as inverted alignments. This will provide estimates for the probabilities $p(j|j', J)$. The most important advantage of the HMM-based alignment models for our approach is the fact, that they do not depend on the unknown target string length I.

Acknowledgement. This work was partly supported by the German Federal Ministry of Education, Science, Research and Technology under the Contract Number 01 IV 601 A (Verbmobil).

References

J. C. Amengual, J. M. Benedí, A. Castaño, A. Marzal, F. Prat, E. Vidal, J. M. Vilar, C. Delogu, A. di Carlo, H. Ney, and S. Vogel. 1996. Example-Based Understanding and Translation Systems (EuTrans): Final Report, Part I. Deliverable of ESPRIT project No. 20268, October.

A.L. Berger, P.F. Brown, J. Cocke, S.A. Della Pietra, V.J. Della Pietra, J.R. Gillett, J.D. Lafferty, R.L. Mercer, H. Printz, and L. Ures. 1994. The Candide System for Machine Translation. In *Proc. ARPA Human Language Technology Workshop, Plainsboro, NJ*, pages 152–157. Morgan Kaufmann Publ., March.

P.F. Brown, S.A. Della Pietra, V.J. Della Pietra, and R.L. Mercer. 1993. Mathematics of Statistical Machine Translation: Parameter Estimation. *Computational Linguistics*, 19(2):263–311.

I. Dagan, K. W. Church, and W. A. Gale. 1993. Robust Bilingual Word Alignment for Machine Aided Translation. In *Proceedings of the Workshop on Very Large Corpora, Columbus, Ohio*, pages 1–8.

P. Fung and K.W. Church. 1994. K-vec: A new Approach for Aligning Parallel Texts. In *Proceedings of the 15th International Conference on Computational Linguistics, Kyoto, Japan*, pages 1096–1102.

M. Kay and M. Röscheisen. 1993. Text-Translation Alignment. *Computational Linguistics*, 19(1):121–142.

H. Ney, D. Mergel, A. Noll, and A. Paeseler. 1992. Data Driven Search Organization for Continuous Speech Recognition. *IEEE Transactions on Signal Processing*, 40(2):272–281, February.

C. Tillmann, S. Vogel, H. Ney, H. Sawaf, and A. Zubiaga. 1997a. Accelerated DP based Search for Statistical Translation. In *Proceedings of the 5th European Conference on Speech Communication and Technology, Rhodes, Greece*, pages 2667–2670, September.

C. Tillmann, S. Vogel, H. Ney, and A. Zubiaga. 1997b. A DP-Based Search using Monotone Alignments in Statistical Translation. In *Proceedings of the ACL/EACL '97, Madrid, Spain*, pages 289–296, July.

S. Vogel, H. Ney, and C. Tillmann. 1996. HMM-Based Word Alignment in Statistical Translation. In *Proceedings of the 16th International Conference on Computational Linguistics, Copenhagen, Denmark*, pages 836–841, August.

W. Wahlster. 1993. Verbmobil: Translation of Face-to-Face Dialogs. In *Proceedings of the MT Summit IV*, pages 127–135, Kobe, Japan.

Ye-Yi Wang and A. Waibel. 1997. Decoding Algorithm in Statistical Translation. In *Proceedings of the ACL/EACL '97, Madrid, Spain*, pages 366–372, July.

D. Wu. 1996. A Polynomial-Time Algorithm for Statistical Machine Translation. In *Proceedings of the 34th Annual Conference of the Association for Computational Linguistics, Santa Cruz, CA*, pages 152 – 158, June.

Zusammenfassung

Wir stellen einen neuartigen Suchalgorithmus für die statistische maschinelle Übersetzung vor, der auf der dynamischen Programmierung (DP) beruht. Während des Suchprozesses werden zwei statistische Wissensquellen kombiniert: Ein Übersetzungsmodell und ein Bigramm-Sprachmodell. Dieser Suchalgorithmus erweitert Hypothesen entlang den Positionen des Zielsatzes, wobei garantiert wird, daß alle Wörter im Quellsatz berücksichtigt werden. Es werden experimentelle Ergebnisse auf der Verbmobil-Aufgabe angegeben.

Résumé

Nous présentons un nouveau algorithme de recherche pour la traduction automatique statistique qui est basée sur la programmation dynamique (DP). Pendant la recherche deux sources d'information statistiques sont combinées: Un modèle de traduction et un bigram language model. Cet algorithme de recherche construit des hypothèses le long des positions de la phrase en langue de cible tout en garantissant la considération progressive des mots dans la phrase en langue de source. Des résultats expérimentaux sur la tâche Verbmobil sont présentés.