An Efficient A* Search Algorithm for Statistical Machine Translation

Franz Josef Och, Nicola Ueffing, Hermann Ney

Lehrstuhl für Informatik VI, Computer Science Department
RWTH Aachen - University of Technology
D-52056 Aachen, Germany
{och,ueffing,ney}@informatik.rwth-aachen.de

Abstract

In this paper, we describe an efficient A* search algorithm for statistical machine translation. In contrary to beam-search or greedy approaches it is possible to guarantee the avoidance of search errors with A*. We develop various sophisticated admissible and almost admissible heuristic functions. Especially our newly developped method to perform a multi-pass A* search with an iteratively improved heuristic function allows us to translate even long sentences. We compare the A* search algorithm with a beam-search approach on the Hansards task.

1 Introduction

The goal of machine translation is the translation of a text given in some source language into a target language. We are given a source string $f_1^J = f_1 ... f_j ... f_J$, which is to be translated into a target string $e_1^I = e_1 ... e_i ... e_I$. Among all possible target strings, we will choose the string with the highest probability:

$$\begin{aligned}\hat{e}_1^I &= \arg\max_{e_1^I} \left\{ Pr(e_1^I | f_1^J) \right\} \\ &= \arg\max_{e_1^I} \left\{ Pr(e_1^I) \cdot Pr(f_1^J | e_1^I) \right\}\end{aligned}$$

The argmax operation denotes the search problem, i.e. the generation of the output sentence in the target language. $Pr(e_1^I)$ is the **language model** of the target language, whereas $Pr(f_1^J | e_1^I)$ denotes the **translation model**.

Many statistical translation models (Brown et al., 1993; Vogel et al., 1996; Och and Ney, 2000b) try to model word-to-word correspondences between source and target words. These correspondences are called an **alignment**. The model is often further restricted in a way such that each source word is assigned *exactly one* target word. The alignment mapping is $j \rightarrow i = a_j$ from source position j to target position $i = a_j$. 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 to any target word. In (statistical) alignment models $Pr(f_1^J, a_1^J | e_1^I)$, the alignment a_1^J is introduced as a hidden variable.

Typically, the search is performed using the so-called maximum approximation:

$$\begin{aligned}\hat{e}_1^I &= \arg\max_{e_1^I} \left\{ Pr(e_1^I) \cdot \sum_{a_1^J} Pr(f_1^J, a_1^J | e_1^I) \right\} \\ &= \arg\max_{e_1^I} \left\{ Pr(e_1^I) \cdot \max_{a_1^J} Pr(f_1^J, a_1^J | e_1^I) \right\}\end{aligned}$$

The search space consists of the set of all possible target language strings e_1^I and all possible alignments a_1^J.

2 IBM Model 4

Various statistical alignment models of the form $Pr(f_1^J, a_1^J | e_1^I)$ have been introduced in (Brown et al., 1993; Vogel et al., 1996; Och and Ney, 2000a). In this paper we use the so-called Model 4 from (Brown et al., 1993).

In Model 4 the statistical alignment model is decomposed into five sub-models:

- the lexicon model $p(f|e)$ for the probability that the source word f is a translation of the target word e,

- the distortion model $p_{=1}(j - j' | C(f_j), E)$ for the probability that the translations of two consecutive target words have the position difference $j - j'$ where $C(f_j)$ is the word class of f_j and E is the word class of the first of the two consecutive target words,

- the distortion model $p_{>1}(j - j'|C(f_j))$ for the probability that the words aligned to one target words have the position difference $j - j'$,
- the fertility model $p(\phi|e)$ for the probability that a target language word e is aligned to ϕ source language words,
- the empty word fertility model $p(\phi_0|e_0)$ for the probability that exactly ϕ_0 words remain unaligned to.

The final probability $p(f_1^J, a_1^J|e_1^I)$ for Model 4 is obtained by multiplying the probabilities of the sub-models for all words. For a detailed description for Model 4 the reader is referred to (Brown et al., 1993).

We use Model 4 in this paper for two reasons. First, it has been shown that Model 4 produces a very good alignment quality in comparison to various other alignment models (Och and Ney, 2000b). Second, the dependences in the distortion model along the target language words make it quite easy to integrate standard n-gram language models in the search process. This would be more difficult in the HMM alignment model (Vogel et al., 1996). Yet, many of the results presented in the following are also applicable to other alignment models.

3 Search problem

The following tasks have to be performed both using A* and beam search (BS):

- The search space has to be structured into a search graph. This search graph typically includes an initial node, intermediary nodes (partial hypotheses), and goal nodes (completed hypotheses). A node contains the following information:
  - the predecessor words u, v in the target language,
  - the score of the hypothesis,
  - a backpointer to the preceding partial hypothesis,
  - the model specific information described at the end of this subsection.
- A scoring function $Q(n) + h(n)$ has to be defined which assigns a score to every node n. For beam search, this is the score $Q(n)$ of a best path to this node. In the A* algorithm, an estimation $h(n)$ of the score of a best path from node n to a goal node is added.

(Berger et al., 1996) presented a method to structure the search space. Our search algorithm for Model 4 uses a similar structuring of the search space. We will shortly review the basic concepts of this search space structure: Every partial hypothesis consists of a prefix of the target sentence and a corresponding alignment. A partial hypothesis is extended by accounting for exactly one additional word of the source sentence. Every extension yields an extension score which is computed by taking into account the lexicon, distortion, and fertility probabilities involved with this extension. A partial hypothesis is called **open** if more source words are to be aligned to the current target word in the following extensions. A hypothesis that is not open is said to be **closed**. Every extension of an open hypothesis will extend the fertility of the previously produced target word and an extension of a closed hypothesis will produce a new word. Therefore, the language model score is added as well if a closed hypothesis is extended.

It is prohibitive to consider all possible translations of all words. Instead, we restrict the search to the most promising candidates by calculating "inverse translations" (Al-Onaizan et al., 1999). The inverse translation probability $p(e \mid f)$ of a source word f is calculated as

$$p(e \mid f) = \frac{p\,(f \mid e)\; p\,(e)}{\sum\limits_{e'} p\,(f \mid e')\; p\,(e')} \quad ,$$

where we use a unigram model $p\,(e)$ to estimate the prior probability of a target word being used. Like (Al-Onaizan et al., 1999), we use only the top 12 translations of a given source language word. In addition, we remove from this list all words whose inverse translation probability is lower than 0.01 times the best inverse translation probability. This **observation pruning** is the only pruning involved in our A* search algorithm. Experiments showed this does not impair translation quality, but the search becomes much more efficient.

In order to keep the search space as small as possible it is crucial to perform a **recombination of search hypotheses**. Every two hypotheses which can be distinguished by neither the language model state nor the translation model state can be recombined, only the hypothesis with a better score of the two needs to be considered in the subsequent search process. We use a standard trigram language model, so the relevant language model state of node n consists of the current word $w(n)$ and the previous word $v(n)$ (later on we will describe an improvement to this). The translation model state depends on the specific model dependencies of Model 4:

- a coverage set $\mathcal{C}(n)$ containing the already translated source language positions,
- the position $j(n)$ of the previously translated source word,
- a flag indicating whether the hypothesis is *open* or *closed*,
- the number of source language words which are aligned to the empty word,
- a flag showing whether the hypothesis is a complete hypothesis or not.

Efficient language model recombination

The recombination procedure which is described above can be improved by taking into account the backing-off structure of the language model. The trigram language model we use has the property that if the count of the bigram $N(u, v) = 0$, then the probability $P(w|u, v)$ depends only on v. In this case the recombination can be significantly improved by recombining all nodes whose language model state has the property $N(u, v) = 0$ only with respect to v. Obviously, this could be generalized to other types of language models as well.

Experiments have shown that by using this efficient recombination, the number of needed hypotheses can be reduced by about a factor of 4.

Search algorithms

We evaluate the following two search algorithms:

- beam search algorithm (BS): (Tillmann, 2001; Tillmann and Ney, 2000)

  In this algorithm the search space is explored in a breadth-first manner. The search algorithm is based on a dynamic programming approach and applies various pruning techniques in order to restrict the number of considered hypotheses. For more details see (Tillmann, 2001).

- A* search algorithm:

  In A*, all search hypotheses are managed in a priority queue. The basic A* search (Nilsson, 1971) can be described as follows:

  1. initialize priority queue with an empty hypothesis
  2. remove the hypothesis with the highest score from the priority queue
  3. if this hypothesis is a goal hypothesis: output this hypothesis and terminate
  4. produce all extensions of this hypothesis and put the extensions to the queue
  5. goto 2

  The so-called heuristic function estimates the probability of a completion of a partial hypothesis. This function is called **admissible** if it never underestimates this probability. Thus, admissible heuristic functions are always optimistic. The A* search algorithm corresponds to the Dijkstra algorithm if the heuristic function is equal to zero.

4 Admissible heuristic function

In order to perform an efficient search with the A* search algorithm it is crucial to use a good heuristic function. We only know of the work by (Wang and Waibel, 1997) dealing with heuristic functions for search in statistical machine translation. They developed a simple heuristic function for Model 2 from (Brown et al., 1993) which was non admissible. In the following we develop a guaranteed admissible heuristic function for Model 4 taking into account distortion probabilities and the coupling of lexicon, fertility, and language model probabilities.

The basic idea for developing a heuristic function for the alignment models is the fact that all source sentence positions which have not been covered so far still have to be translated in order to complete the sentence. Therefore, the value of the heuristic function $H^X(n)$ for a node n can be deduced if we have an estimation $h^X(j)$ of the optimal score of translating position j (here X denotes different possibilities to choose the heuristic function):

$$H^X(n) = \prod_{j \notin \mathcal{C}(n)} h^X(j) \ ,$$

where $\mathcal{C}(n)$ is the coverage set.

The simplest realization of a heuristic function, denoted as $h^T(j)$, takes into account only the translation probability $p(f|e)$:

$$h^T(j) = \max_e p(f_j|e)$$

This heuristic function can be refined by introducing also the fertility probabilities (symbol F) of a target word e:

$$h^{TF}(j) = \\ = \max\left\{ \max_{e \neq e_0, \phi} \ p(f_j|e) \sqrt[\phi]{p(\phi|e)}, \ p(f|e_0) \right\}$$

Thereby, a coupling between the translation and fertility probabilities is achieved. We have to take the ϕ-th root in order to avoid that the fertility probability of a target word whose fertility is higher than one is taken into account for every source word aligned to it. For words which are translated by the empty word e_0, no fertility probability is used.

The language model can be incorporated by considering that for every target word there exists an optimal language model probability:

$$p^L(e) = \max_{u,v} p(e|u,v)$$

Here, we assume a trigram language model.

Thus, a heuristic function including a coupling between translation, fertility, and language model probabilities (TFL) is given by:

$$h^{TFL}(j) = \\ = \max\left\{ \max_{e,\phi} p(f_j|e) \sqrt[\phi]{p(\phi|e) p^L(e)}, \ p(f|e_0) \right\}$$

This value can be precomputed efficiently before the search process itself starts.

The heuristic function for the distortion probabilities depends on the used model. For Model 4, we obtain:

$$h^D(j) = \max_{j',E} p(j - j'|E, C(f_j))$$

Here, E refers to the class of the previously aligned target word.

The heuristic functions $h^D(j)$ involve maximizations over the source positions j'. The domain of this variable shrinks during search as more and more words get translated. Therefore, it is possible to improve this heuristic function during search to perform a maximization only over the free source language positions j'. For Model 4 we compute the following heuristic function with two arguments:

$$h^D(j', j) = \max_E p(j - j'|E, C(f_j))$$

Thus, we obtain as an estimation of the distortion probability

$$h^D(j) \quad = \quad \max_{j' \notin \mathcal{C}(n)} h^D(j', j) \quad .$$

This yields the following heuristic functions taking into account translation, fertility, language, and distortion model probabilities:

$$H^{TFLD}(n) = \prod_{j \notin \mathcal{C}(n)} h^{TFL}(j) \cdot h^D(j) \qquad (1)$$

Using these heuristic functions we have the overhead of performing this rest cost estimation for every coverage set in search. The experiments will show that these additional costs are overcompensated by the gain in reducing the search space that has to be expanded during the A* search.

To assess the predictive power of the various components in the heuristic, we compare the value of the heuristic function of the empty hypothesis with the score of the optimal translation. A heuristic function is better if the difference between these two values is small. Table 1 contains a comparison of various heuristic functions. We compare the average costs (negative logarithm of the probabilities) of the optimal translation and the average of the estimated costs of the empty hypothesis. Typically, the estimated costs of $TFLD$ and the real costs differ by factor 3.

Table 1: Predictive power of admissible and almost admissible heuristic functions.

sentence length	HF for initial node				empirical score	goal node score
	T	TF	TFL	TFLD		
6	5.1	7.2	12.7	13.0	25.9	35.5
8	5.7	8.2	16.0	16.3	29.8	43.7
10	8.1	11.6	19.4	19.7	36.5	55.8
12	9.5	13.7	20.7	21.1	43.9	63.4

We will see later in Section 6 that the guaranteed admissible heuristic functions described above result in dramatically more efficient search.

5 Empirical heuristic functions

In this section we describe a new method to obtain an almost admissible heuristic function by a multi pass search. This yields a significantly more efficient search than using the admissible heuristic functions. Thus, we lose the strict guarantee to avoid search errors, but obtain a significant time gain.

The idea of an **empirical heuristic function** is to perform a multi-pass search. In the first pass a good admissible heuristic function (here: H^{TFLD}) is used. If this search does not need too much memory the search process is finished. If the search failed, it is restarted using an improved heuristic function which had been obtained during the initial search process. This heuristic function is computed such that it has the property that it is admissible with respect to the explored search space. That means, the heuristic function is optimistic with respect to every node in the search space explored in the first pass.

Specifically, during the first pass, we maintain a two-dimensional matrix $h^E(j, j')$ with $(J+2) \cdot (J+2)$ entries which are all initialized with ∞. The entry $h^E(j, j')$ is the best score that was computed for translating the source language word in position j' if the previously covered source sentence position is j. The matrix entry is updated for every extension of a node $n \to n'$:

$$h^E(j(n), j(n')) := \\ = \max \left\{ h^E(j(n), j(n')),\ p(n, n') \right\}$$

Here, $p(n, n')$ is the probability of the extension $n \to n'$. $h^E(0, j)$ is the empirical score of starting a sentence by covering the j-th source sentence position first. Likewise, $h^E(j, J+1)$ is the empirical score of finishing a sentence with j as the last source sentence position that was covered. This yields

$$h^E(j) = \max_{j' \notin \mathcal{C}(n) \vee j' = J+1} h^E(j, j') \ .$$

In this calculation of $h^E(j)$, we maximize over the columns of a matrix. The translation of the source sentence can be viewed as a Traveling Salesman Problem where the source sentence positions are the cities that have to be visited. Thus, the maximization over the columns is equivalent to assuring that the position j will be left after the visit. We design an improved heuristic function using the following principle (Aigner, 1993): Each city has to be both reached and left. Therefore, in order to take an upper bound of reaching a city into account, we divide each column of the matrix by its maximum and maximize over the rows of the matrix (Aigner, 1993):

$$h^{E+}(j) = \max_{j' \notin \mathcal{C}(n) \vee j' = j(n)} h^E(j', j) / h^E(j') \ .$$

We obtain the following empirical heuristic functions:

$$H^E(n) \quad = \quad \prod_{j \notin \mathcal{C}(n) \vee j = j(n)} h^E(j)$$

$$H^{E+}(n) = \\ = \prod_{j \notin \mathcal{C}(n) \vee j = j(n)} h^E(j) \cdot \prod_{j' \notin \mathcal{C}(n) \vee j' = J+1} h^{E+}(j')$$

If the search fails in the first pass due to the restriction of the number of hypotheses – which was 1 million in all experiments – the search can be started again using $H^{E+}(n)$ as a heuristic. To avoid an overestimation of the actual costs, we multiply the empirical costs by a factor lower than

Table 3: Training corpus statistics (* without punctuation marks).

	French	English
sentences	49000	49000
words	743903	816964
words*	664058	730880
average sentence length	16.9	14.6
vocabulary size	19831	24892

Table 4: Test corpora statistics.

Corpus	# Sentences	# Words	
		F	E
T6	50	300	329
T8	50	400	403
T10	50	500	509
T12	50	600	601
T14	50	700	644

1. We found in our experiments that a factor of 0.7 is sufficient. The search was restarted up to 4 times if it failed. Using this method, it is possible to translate sentences that are longer than 10 words with a restriction to 1 million hypotheses.

Table 1 shows the value of the empirical heuristic function of the empty node compared to the score of the optimal goal node. The estimated costs and the real costs now differ only by a factor of 1.5 instead of a factor of 3 for the TFLD heuristic function before.

6 Results

We present results on the HANSARDS task which consists of proceedings of the Canadian parliament that are kept both in French and in English. Table 3 shows the details of our training corpus. We used different the test corpora with sentences of length 6-14 words (Table 4).

In all experiments, we use the following two error criteria:

- WER (word error rate):
  The WER is computed as the minimum number of substitution, insertion and deletion operations that have to be performed to convert the generated string into the target string.
- PER (position independent word error rate):
  The word order of a French/English sentence pair can be quite different. As a result, the word order of the automatically generated target sentence can be different from that of the given target sentence, but nevertheless acceptable so that the WER measure alone could be misleading. In order to overcome this problem, we introduce the position independent word error rate (PER) as additional measure. This measure compares the words in the two sentences *without* taking the word order into account.

In the following experiments we restricted the maximum number of active search hypotheses in A* search to 1 million. Every hypothesis has an effective memory requirement of about 100 Byte. Therefore, we obtain a dynamic memory requirement of about 100 MByte.

In order to speed up the search, we restricted the reordering of words in IBM-style (Berger et al., 1996; Tillmann, 2001). According to this restriction, up to 3 source sentence positions may be skipped and translated later, i. e. during the search process there may be up to 3 uncovered positions left of the rightmost covered position in the source sentence. The word error rate does not increase compared to a non-restricted reordering, but the search becomes much more efficient.

Table 5 shows how many sentences with different sentence lengths can be translated using beam search and A* with various heuristic functions. Obviously, the BS approach is able to translate any sentence length, therefore the search success rate is 100%. Without any heuristic function A* is only able to translate all 8-word sentences (with the restriction of a maximum number of 1 million hypotheses). Using more sophisticated heuristic functions we are also able to translate all 10-word sentences with A*.

Table 6 compares the search errors of A* and BS. During the BS search, **translation pruning** is carried out. The different hypotheses are distinguished according to the set of covered positions of the source sentence. For every set, the best score of all hypotheses is computed. Only those hypotheses are kept whose score is greater than this best score multiplied with a threshold. We chose the threshold to be 2.5, 5.0, 7.5 and 10.0 (see Table 6).

Table 2: Effect of observation pruning on the translation quality (average over all test sets).

# inverse translations	10	12	14	16	18	20
WER	73.81	73.33	75.50	76.23	76.19	76.59
PER	68.02	66.93	70.07	71.16	71.24	71.16

Table 5: Search Success Rate (1 million hypotheses) [%].

sentence length		6	8	10	12
BS		100	100	100	100
A*:	no	100	100	86	12
	T	100	100	88	20
	TF	100	100	88	22
	TFL	100	100	92	36
	TFLD	100	100	92	36
	E	100	100	100	74
	E+	100	100	100	84

Table 6: Search errors [%].

sentence length		6	8	10	12	14
BS	2.5	26	28	38	50	38
	5.0	2	0	2	6	4
	7.5	0	0	0	4	2
	10.0	0	0	0	4	2
A*		0	0	0	0	0

For A* we never observe any search errors. In the case of the admissible heuristic functions, this is guaranteed by the approach. As can be seen from Table 6, the BS algorithm with a large beam rarely produces search errors.

Table 7 compares the translation efficiency of the various search algorithms. We see that beam search even with a very large beam producing only very few search errors is much more efficient than the used A* search algorithm.

Table 8 contains an assessment of translation quality comparison of A* and BS using the T6, T8, T10, T12-test corpus. For A*, we use the E+ rest cost estimation as this gives optimal results. From the 200 sentences of these test corpora we can translate 192 sentences using the 1 million hypotheses constraint. For the remaining sentences we performed a search with 4 million hypotheses

Table 7: Average search time [s] per sentence.

sentence length		6	8	10	12
BS:	2.5	0.06	0.18	0.60	1.16
	5.0	0.24	0.84	2.90	6.48
	7.5	0.50	2.14	7.06	16.26
	10.0	0.78	3.30	11.86	26.42
A*:	E+	1.58	13.04	100	394

(cf. below) which lead to a success for all the 12-word sentences.

The number of hypotheses in A* search

We restricted the maximal number of hypotheses to 1 million. This was sufficient for translating 10-word sentences, as the search algorithm success rate in Table 5 shows. For longer sentences it is necessary to allow for a larger number of hypotheses. For the sentences of lengths 12 and 14, we performed an A* search (E+) with 2, 4 and 8 million possible hypotheses. The search algorithm success rate for those searches is contained in Table 9. We see a significant effect on the number of successful searches.

7 Conclusion

We have developed sophisticated admissible and almost admissible heuristic functions for statistical machine translation. We have focussed on Model 4, but most of the computations could be easily extended to other statistical alignment models (like HMM or Model 5). We especially have observed the following effects:

- The heuristic function has a strong effect on the efficiency of the A* search. Without any heuristic function only 75 % of the test corpus sentences can be translated (using the 1 million hypotheses constraint). Using the

Table 8: Translation quality.

	BS (2.5)	BS (5.0)	BS (7.5)	BS (10.0)	A* (E+)
WER	69.65	68.78	68.68	68.68	68.68
PER	62.65	61.62	61.51	61.51	61.45

Table 9: A* (E+) Success Rate for 12- and 14-word sentences [%].

# hypotheses	1 million	2 million	4 million	8 million
12	42	80	100	100
14	2	20	70	100

best admissible heuristic function $TFLD$ we can translate 82 %.

- Using the empirical heuristic function we can translate 96 % of the sentences with A* search. This heuristic function does not guarantee to avoid search errors, but this case never occurred in our experiments.

From these results we conclude that it is often possible to faster compute acceptable results using a beam search approach. Therefore, this is the method of choice in practice. From a theoretical viewpoint it is interesting that using A* it is possible to translate guaranteed without search errors. In addition, without having a chance to perform search without search errors it is almost impossible to assess if errors in translation should be assigned to the model/training or to the search heuristics. Therefore, the A* algorithm is especially useful during the development of a statistical machine translation system.

Acknowledgment

This paper is based on work supported partly by the VERBMOBIL project (contract number 01 IV 701 T4) by the German Federal Ministry of Education, Science, Research and Technology. In addition, this work was 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.

References

M. Aigner. 1993. *Diskrete Mathematik*. Verlag Vieweg, Braunschweig/Wiesbaden, Germany.

Y. Al-Onaizan, J. Curin, M. Jahr, K. Knight, J. Lafferty, I. D. Melamed, F. J. Och, D. Purdy, N. A. Smith, and D. Yarowsky. 1999. Statistical machine translation, final report, JHU workshop. `http://www.clsp.jhu.edu/ws99/projects/mt/final_report/mt-final-report.ps`.

A. L. Berger, S. A. Della Pietra P. F. Brown, V. J. Della Pietra, J. R. Gillett, A. S. Kehler, and R. L. Mercer. 1996. Language translation apparatus and method of using context-based translation models. In *United States Patent*, number 5510981. April.

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

N. Nilsson. 1971. *Problem-Solving Methods in Artificial Intelligence*. McGraw-Hill, McGraw-Hill, New York.

F. J. Och and H. Ney. 2000a. A comparison of alignment models for statistical machine translation. In *COLING '00: The 18th Int. Conf. on Computational Linguistics*, pages 1086–1090, Saarbrücken, Germany, August.

F. J. Och and H. Ney. 2000b. Improved statistical alignment models. In *Proc. of the 38th Annual Meeting of the Association for Computational Linguistics*, pages 440–447, Hongkong, China, October.

C. Tillmann and H. Ney. 2000. Word re-ordering and DP-based search in statistical machine translation. In *COLING '00: The 18th Int. Conf. on Computational Linguistics*, pages 850–856, Saarbrücken, Germany, August.

C. Tillmann. 2001. *Word Re-Ordering and Dynamic Programming based Search Algorithms for Statistical Machine Translation*. Ph.D. thesis, RWTH Aachen, Germany, May.

S. Vogel, H. Ney, and C. Tillmann. 1996. HMM-based word alignment in statistical translation. In *COLING '96: The 16th Int. Conf. on Computational Linguistics*, pages 836–841, Copenhagen, August.

Ye-Yi Wang and Alex Waibel. 1997. Decoding algorithm in statistical translation. In *Proc. 35th Annual Conf. of the Association for Computational Linguistics*, pages 366–372, Madrid, Spain, July.