

# A Phrase-Based, Joint Probability Model for Statistical Machine Translation

**Daniel Marcu**

Information Sciences Institute and  
Department of Computer Science  
University of Southern California  
4676 Admiralty Way, Suite 1001  
Marina del Rey, CA, 90292  
marcu@isi.edu

**William Wong**

Language Weaver Inc.  
1639 11th St., Suite 100A  
Santa Monica, CA 90404  
wwong@languageweaver.com

## Abstract

We present a joint probability model for statistical machine translation, which automatically learns word and phrase equivalents from bilingual corpora. Translations produced with parameters estimated using the joint model are more accurate than translations produced using IBM Model 4.

## 1 Motivation

Most of the noisy-channel-based models used in statistical machine translation (MT) (Brown et al., 1993) are conditional probability models. In the noisy-channel framework, each source sentence  $e$  in a parallel corpus is assumed to “generate” a target sentence  $f$  by means of a stochastic process, whose parameters are estimated using traditional EM techniques (Dempster et al., 1977). The generative model explains how source words are mapped into target words and how target words are re-ordered to yield well-formed target sentences. A variety of methods are used to account for the re-ordering stage: word-based (Brown et al., 1993), template-based (Och et al., 1999), and syntax-based (Yamada and Knight, 2001), to name just a few. Although these models use different generative processes to explain how translated words are re-ordered in a target language, at the lexical level they are quite similar; all these models assume that source words are *individually* translated into target words.<sup>1</sup>

<sup>1</sup>The individual words may contain a non-existent element, called NULL.

We suspect that MT researchers have so far chosen to automatically learn translation lexicons defined only over words for primarily pragmatic reasons. Large scale bilingual corpora with vocabularies in the range of hundreds of thousands yield very large translation lexicons. Tuning the probabilities associated with these large lexicons is a difficult enough task to deter one from trying to scale up to learning phrase-based lexicons. Unfortunately, trading space requirements and efficiency for explanatory power often yields non-intuitive results.

Consider, for example, the parallel corpus of three sentence pairs shown in Figure 1. Intuitively, if we allow any Source words to be aligned to any Target words, the best alignment that we can come up with is the one in Figure 1.c. Sentence pair (S2, T2) offers strong evidence that “b c” in language S means the same thing as “x” in language T. On the basis of this evidence, we expect the system to also learn from sentence pair (S1, T1) that “a” in language S means the same thing as “y” in language T. Unfortunately, if one works with translation models that do not allow Target words to be aligned to more than one Source word — as it is the case in the IBM models (Brown et al., 1993) — it is impossible to learn that the phrase “b c” in language S means the same thing as word “x” in language T. The IBM Model 4 (Brown et al., 1993), for example, converges to the word alignments shown in Figure 1.b and learns the translation probabilities shown in Figure 1.a.<sup>2</sup> Since in the IBM model one cannot link a Target word to more than a Source word, the training procedure

<sup>2</sup>To train the IBM-4 model, we used Giza (Al-Onaizan et al., 1999).

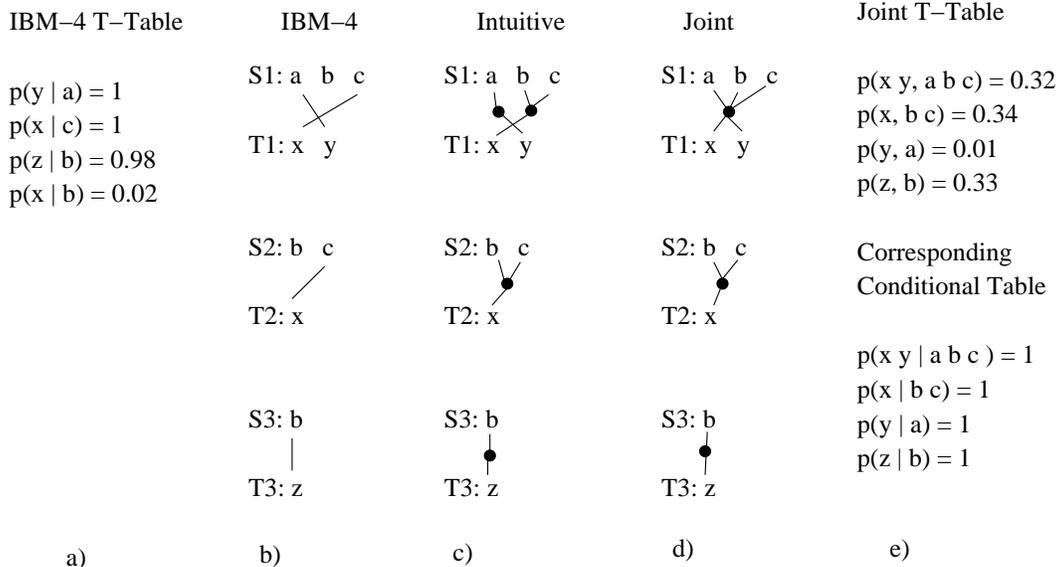


Figure 1: Alignments and probability distributions in IBM Model 4 and our joint phrase-based model.

yields unintuitive translation probabilities. (Note that another good word-for-word model is one that assigns high probability to  $p(x | b)$  and  $p(z | b)$  and low probability to  $p(x | c)$ .)

In this paper, we describe a translation model that assumes that lexical correspondences can be established not only at the word level, but at the phrase level as well. In contrast with many previous approaches (Brown et al., 1993; Och et al., 1999; Yamada and Knight, 2001), our model does not try to capture how Source sentences can be mapped into Target sentences, but rather how Source and Target sentences can be generated simultaneously. In other words, in the style of Melamed (2001), we estimate a joint probability model that can be easily marginalized in order to yield conditional probability models for both source-to-target and target-to-source machine translation applications. The main difference between our work and that of Melamed is that we learn joint probability models of translation equivalence not only between words but also between phrases and we show that these models can be used not only for the extraction of bilingual lexicons but also for the automatic translation of unseen sentences.

In the rest of the paper, we first describe our model (Section 2) and explain how it can be implemented/trained (Section 3). We briefly describe a

decoding algorithm that works in conjunction with our model (Section 4) and evaluate the performance of a translation system that uses the joint-probability model (Section 5). We end with a discussion of the strengths and weaknesses of our model as compared to other models proposed in the literature.

## 2 A Phrase-Based Joint Probability Model

### 2.1 Model 1

In developing our joint probability model, we started out with a very simple generative story. We assume that each sentence pair in our corpus is generated by the following stochastic process:

1. Generate a bag of concepts  $C$ .
2. For each concept  $c_i \in C$ , generate a pair of phrases  $(\vec{e}_i, \vec{f}_i)$ , according to the distribution  $t(\vec{e}_i, \vec{f}_i)$ , where  $\vec{e}_i$  and  $\vec{f}_i$  each contain at least one word.
3. Order the phrases generated in each language so as to create two linear sequences of phrases; these sequences correspond to the sentence pairs in a bilingual corpus.

For simplicity, we initially assume that the bag of concepts and the ordering of the generated phrases are modeled by uniform distributions. We do not assume that  $c_i$  is a hidden variable that generates

the pair  $(\vec{e}_i, \vec{f}_i)$ , but rather that  $c_i = (\vec{e}_i, \vec{f}_i)$ . Under these assumptions, it follows that the probability of generating a sentence pair (E, F) using concepts  $c_i \in C$  is given by the product of all phrase-to-phrase translation probabilities,  $\prod_{c_i \in C} t(\vec{e}_i, \vec{f}_i)$  that yield bags of phrases that can be ordered linearly so as to obtain the sentences E and F. For example, the sentence pair “a b c” — “x y” can be generated using two concepts, (“a b” : “y”) and (“c” : “x”); or one concept, (“a b c” : “x y”), because in both cases the phrases in each language can be arranged in a sequence that would yield the original sentence pair. However, the same sentence pair cannot be generated using the concepts (“a b” : “y”) and (“c” : “y”) because the sequence “x y” cannot be recreated from the two phrases “y” and “y”. Similarly, the pair cannot be generated using concepts (“a c” : “x”) and (“b” : “y”) because the sequence “a b c” cannot be created by catenating the phrases “a c” and “b”.

We say that a set of concepts  $C$  can be linearized into a sentence pair (E, F) if E and F can be obtained by permuting the phrases  $\vec{e}_i$  and  $\vec{f}_i$  that characterize all concepts  $c_i \in C$ . We denote this property using the predicate  $L(E, F, C)$ . Under this model, the probability of a given sentence pair (E, F) can then be obtained by summing up over all possible ways of generating bags of concepts  $C \in \mathcal{C}$  that can be linearized to (E, F).

$$p(E, F) = \sum_{C \in \mathcal{C} | L(E, F, C)} \prod_{c_i \in C} t(\vec{e}_i, \vec{f}_i) \quad (1)$$

## 2.2 Model 2

Although Model 1 is fairly unsophisticated, we have found that it produces in practice fairly good alignments. However, this model is clearly unsuited for translating unseen sentences as it imposes no constraints on the ordering of the phrases associated with a given concept. In order to account for this, we modify slightly the generative process in Model 1 so as to account for distortions. The generative story of Model 2 is this:

1. Generate a bag of concepts  $C$ .
2. Initialize E and F to empty sequences  $\epsilon$ .
3. Randomly take a concept  $c_i \in C$  and generate a pair of phrases  $(\vec{e}_i, \vec{f}_i)$ , according to the dis-

tribution  $t(\vec{e}_i, \vec{f}_i)$ , where  $\vec{e}_i$  and  $\vec{f}_i$  each contain at least one word. Remove then  $c_i$  from  $C$ .

4. Append phrase  $\vec{f}_i$  at the end of F. Let  $k$  be the start position of  $\vec{f}_i$  in F.
5. Insert phrase  $\vec{e}_i$  at position  $l$  in E provided that no other phrase occupies any of the positions between  $l$  and  $l + |\vec{e}_i|$ , where  $|\vec{e}_i|$  gives the length of the phrase  $\vec{e}_i$ . We hence create the alignment between the two phrases  $\vec{f}_i$  and  $\vec{e}_i$  with probability

$$\prod_{p=k}^{k+|\vec{f}_i|} d(p, (l + |\vec{e}_i|)/2),$$

where  $d(i, j)$  is a position-based distortion distribution.

6. Repeat steps 3 to 5 until  $C$  is empty.

In Model 2, the probability to generate a sentence pair (E, F) is given by formula (2), where  $pos(\vec{f}_i^k)$  denotes the position of word  $k$  of phrase  $\vec{f}_i$  in sentence F and  $pos_{cm}(\vec{e}_i)$  denotes the position in sentence E of the center of mass of phrase  $e_i$ .

$$p(E, F) = \sum_{C \in \mathcal{C} | L(E, F, C)} \prod_{c_i \in C} [t(\vec{e}_i, \vec{f}_i) \times \prod_{k=1}^{|\vec{f}_i|} d(pos(\vec{f}_i^k), pos_{cm}(\vec{e}_i))] \quad (2)$$

Model 2 implements an absolute position-based distortion model, in the style of IBM Model 3. We have tried many types of distortion models. We eventually settled for the model discussed here because it produces better translations during decoding. Since the number of factors involved in computing the probability of an alignment does not vary with the size of the Target phrases into which Source phrases are translated, this model is not predisposed to produce translations that are shorter than the Source sentences given as input.

## 3 Training

Training the models described in Section 2 is computationally challenging. Since there is an exponential number of alignments that can generate a sentence pair (E, F), it is clear that we cannot apply the

1. Determine high-frequency n-grams in the bilingual corpus.
2. Initialize the t-distribution table.
3. Apply EM training on the Viterbi alignments, while using smoothing.
4. Generate conditional model probabilities.

Figure 2: Training algorithm for the phrase-based joint probability model.

EM training algorithm exhaustively. To estimate the parameters of our model, we apply the algorithm in Figure 2, whose steps are motivated and described below.

### 3.1 Determine high-frequency n-grams in E and F

If one assumes from the outset that any phrases  $\vec{e}_i \in \mathcal{E}^*$  and  $\vec{f}_i \in \mathcal{F}^*$  can be generated from a concept  $c_i$ , one would need a supercomputer in order to store in the memory a table that models the  $t(\vec{e}_i, \vec{f}_i)$  distribution. Since we don't have access to computers with unlimited memory, we initially learn t distribution entries only for the phrases that occur often in the corpus and for unigrams. Then, through smoothing, we learn t distribution entries for the phrases that occur rarely as well. In order to be considered in step 2 of the algorithm, a phrase has to occur at least five times in the corpus.

### 3.2 Initialize the t-distribution table

Before the EM training procedure starts, one has no idea what word/phrase pairs are likely to share the same meaning. In other words, all alignments that can generate a sentence pair (E, F) can be assumed to have the same probability. Under these conditions, the evidence that a sentence pair (E, F) contributes to the fact that  $(\vec{e}_i, \vec{f}_i)$  are generated by the same concept  $c_i$  is given by the number of alignments that can be built between (E, F) that have a concept  $c_i$  that is linked to phrase  $\vec{e}_i$  in sentence E and phrase  $\vec{f}_i$  in sentence F divided by the total number of align-

ments that can be built between the two sentences. Both these numbers can be easily approximated.

Given a sentence E of  $l$  words, there are  $S(l, k)$  ways in which the  $l$  words can be partitioned into  $k$  non-empty sets/concepts, where  $S(l, k)$  is the Stirling number of second kind.

$$S(l, k) = \frac{1}{k!} \sum_{i=0}^{k-1} (-1)^i \binom{k}{i} (k-i)^l \quad (3)$$

There are also  $S(m, k)$  ways in which the  $m$  words of a sentence F can be partitioned into  $k$  non-empty sets. Given that any words in E can be mapped to any words in F, it follows that there are  $\sum_{k=1}^{\min(l, m)} k! S(l, k) S(m, k)$  alignments that can be built between two sentences (E, F) of lengths  $l$  and  $m$ , respectively. When a concept  $c_i$  generates two phrases  $(\vec{e}_i, \vec{f}_i)$  of length  $a$  and  $b$ , respectively, there are only  $l-a$  and  $m-b$  words left to link. Hence, in the absence of any other information, the probability that phrases  $\vec{e}_i$  and  $\vec{f}_i$  are generated by the same concept  $c_i$  is given by formula (4).

$$\frac{\sum_{k=1}^{\min(l-a, m-b)} k! S(l-a, k) S(m-b, k)}{\sum_{k=1}^{\min(l, m)} k! S(l, k) S(m, k)} \quad (4)$$

Note that the fractional counts returned by equation (4) are only an approximation of the t distribution that we are interested in because the Stirling numbers of the second kind do not impose any restriction on the words that are associated with a given concept be consecutive. However, since formula (4) overestimates the numerator and denominator equally, the approximation works well in practice.

In the second step of the algorithm, we apply equation (4) to collect fractional counts for all unigram and high-frequency n-gram pairs in the cartesian product defined over the phrases in each sentence pair (E, F) in a corpus. We sum over all these t-counts and we normalize to obtain an initial joint distribution  $t$ . This step amounts to running the EM algorithm for one step over all possible alignments in the corpus.

### 3.3 EM training on Viterbi alignments

Given a non-uniform t distribution, phrase-to-phrase alignments have different weights and there are no

other tricks one can apply to collect fractional counts over all possible alignments in polynomial time. Starting with step 3 of the algorithm in Figure 2, for each sentence pair in a corpus, we greedily produce an initial alignment by linking together phrases so as to create concepts that have high  $t$  probabilities. We then hillclimb towards the Viterbi alignment of highest probability by breaking and merging concepts, swapping words between concepts, and moving words across concepts. We compute the probabilities associated with all the alignments we generate during the hillclimbing process and collect  $t$  counts over all concepts in these alignments.

We apply this Viterbi-based EM training procedure for a few iterations. The first iterations estimate the alignment probabilities using Model 1. The rest of the iterations estimate the alignment probabilities using Model 2.

During training, we apply smoothing so we can associate non-zero values to phrase-pairs that do not occur often in the corpus.

### 3.4 Derivation of conditional probability model

At the end of the training procedure, we take marginals on the joint probability distributions  $t$  and  $d$ . This yields conditional probability distributions  $t(\vec{f}_i | \vec{e}_i)$  and  $d(pos^F | pos^E)$ , which we use for decoding.

### 3.5 Discussion

When we run the training procedure in Figure 2 on the corpus in Figure 1, after four Model 1 iterations we obtain the alignments in Figure 1.d and the joint and conditional probability distributions shown in Figure 1.e. At prima facie, the Viterbi alignment for the first sentence pair appears incorrect because we, as humans, have a natural tendency to build alignments between the smallest phrases possible. However, note that the choice made by our model is quite reasonable. After all, in the absence of additional information, the model can either assume that “a” and “y” mean the same thing or that phrases “a b c” and “x y” mean the same thing. The model chose to give more weight to the second hypothesis, while preserving some probability mass for the first one.

Also note that although the joint distribution puts the second hypothesis at an advantage, the conditional distribution does not. The conditional distri-

bution in Figure 1.e is consistent with our intuitions that tell us that it is reasonable both to translate “a b c” into “x y”, as well as “a” into “y”. The conditional distribution mirrors perfectly our intuitions.

## 4 Decoding

For decoding, we have implemented a greedy procedure similar to that proposed by Germann et al. (2001). Given a Foreign sentence  $F$ , we first produce a gloss of it by selecting phrases in  $\mathcal{E}^*$  that maximize the probability  $p(E, F)$ . We then iteratively hillclimb by modifying  $E$  and the alignment between  $E$  and  $F$  so as to maximize the formula  $p(E)p(F | E)$ . We hillclimb by modifying an existing alignment/translation through a set of operations that modify locally the alignment/translation built until a given time. These operations replace the English side of an alignment with phrases of different probabilities, merge and break existing concepts, and swap words across concepts. The probability  $p(E)$  is computed using a simple trigram language model that was trained using the CMU Language Modeling Toolkit (Clarkson and Rosenfeld, 1997). The language model is estimated at the word (not phrase) level. Figure 3 shows the steps taken by our decoder in order to find the translation of sentence “je vais me arrêter là .” Each intermediate translation in Figure 3 is preceded by its probability and succeeded by the operation that changes it to yield a translation of higher probability.

## 5 Evaluation

To evaluate our system, we trained both Giza (IBM Model 4) (Al-Onaizan et al., 1999) and our joint probability model on a French-English parallel corpus of 100,000 sentence pairs from the Hansard corpus. The sentences in the corpus were at most 20 words long. The English side had a total of 1,073,480 words (21,484 unique tokens). The French side had a total of 1,177,143 words (28,132 unique tokens).

We translated 500 unseen sentences, which were uniformly distributed across lengths 6, 8, 10, 15, and 20. For each group of 100 sentences, we manually determined the number of sentences translated perfectly by the IBM model decoder of Germann et al. (2001) and the decoder that uses the joint prob-

Model	Percent perfect translations						IBM Bleu score					
	Sentence length						Sentence length					
	6	8	10	15	20	Avg.	6	8	10	15	20	Avg.
IBM	36	26	35	11	2	22	0.2076	0.2040	0.2414	0.2248	0.2011	0.2158
Phrase-based	43	37	33	19	6	28	0.2574	0.2181	0.2435	0.2407	0.2028	0.2325

Table 1: Comparison of IBM and Phrase-Based, Joint Probability Models on a translation task.

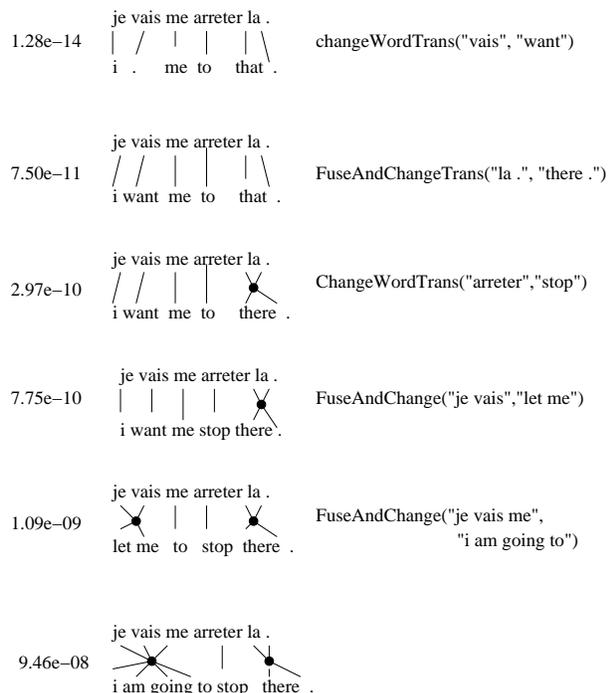


Figure 3: Example of phrase-based greedy decoding.

ability model. We also evaluated the translations automatically, using the IBM-Bleu metric (Papineni et al., 2002). The results in Table 1 show that the phrased-based translation model proposed in this paper significantly outperforms IBM Model 4 on both the subjective and objective metrics.

## 6 Discussion

### 6.1 Limitations

The main shortcoming of the phrase-based model in this paper concerns the size of the t-table and the cost of the training procedure we currently apply. To keep the memory requirements manageable, we arbitrarily restricted the system to learning phrase translations of at most six words on each side. Also,

the swap, break, and merge operations used during the Viterbi training are computationally expensive. We are currently investigating the applicability of dynamic programming techniques to increase the speed of the training procedure.

Clearly, there are language pairs for which it would be helpful to allow concepts to be realized as non-contiguous phrases. The English word “not”, for example, is often translated into two French words, “ne” and “pas”. But “ne” and “pas” almost never occur in adjacent positions in French texts. At the outset of this work, we attempted to develop a translation model that enables concepts to be mapped into non-contiguous phrases. But we were not able to scale and train it on large amounts of data. The model described in this paper cannot learn that the English word “not” corresponds to the French words “ne” and “pas”. However, our model learns to deal with negation by memorizing longer phrase translation equivalents, such as (“ne est pas”, “is not”); (“est inadmissible”, “is not good enough”); and (“ne est pas ici”, “is not here”).

### 6.2 Comparison with other work

A number of researchers have already gone beyond word-level translations in various MT settings. For example, Melamed (2001) uses word-level alignments in order to learn translations of non-compositional compounds. Och and Ney (1999) learn phrase-to-phrase mappings involving word classes, which they call “templates”, and exploit them in a statistical machine translation system. And Marcu (2001) extracts phrase translations from automatically aligned corpora and uses them in conjunction with a word-for-word statistical translation system. However, none of these approaches learn simultaneously the translation of phrases/templates and the translation of words. As a consequence, there is a chance that the learning procedure will not discover phrase-level patterns that occur often in the

data. In our approach, phrases are not treated differently from individual words, and as a consequence the likelihood of the EM algorithm converging to a better local maximum is increased.

Working with phrase translations that are learned independent of a translation model can also affect the decoder performance. For example, in our previous work (Marcu, 2001), we have used a statistical translation memory of phrases in conjunction with a statistical translation model (Brown et al., 1993). The phrases in the translation memory were automatically extracted from the Viterbi alignments produced by Giza (Al-Onaizan et al., 1999) and re-used in decoding. The decoder described in (Marcu, 2001) starts from a gloss that uses the translations in the translation memory and then tries to improve on the gloss translation by modifying it incrementally, in the style described in Section 4. However, because the decoder hill-climbs on a word-for-word translation model probability, it often discards good phrasal translations in favour of word-for-word translations of higher probability. The decoder in Section 4 does not have this problem because it hill-climbs on translation model probabilities in which phrases play a crucial role.

**Acknowledgments.** This work was supported by DARPA-ITO grant N66001-00-1-9814 and by NSF-STTR grant 0128379.

## References

- Yaser Al-Onaizan, Jan Curin, Michael Jahr, Kevin Knight, John Lafferty, Dan Melamed, Franz-Josef Och, David Purdy, Noah A. Smith, and David Yarowsky. 1999. Statistical machine translation. Final Report, JHU Summer Workshop.
- Peter F. Brown, Stephen A. Della Pietra, Vincent J. Della Pietra, and Robert L. Mercer. 1993. The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics*, 19(2):263–311.
- Philip Clarkson and Ronald Rosenfeld. 1997. Statistical language modeling using the CMU-Cambridge toolkit. In *Proceedings of Eurospeech*, September.
- A. P. Dempster, N. M. Laird, and D. B. Rubin. 1977. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society*, 39(Ser B):1–38.
- Ulrich Germann, Mike 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'01)*, pages 228–235, Toulouse, France, July 6–11. Decoder available at <http://www.isi.edu/natural-language/projects/rewrite/>.
- Daniel Marcu. 2001. Towards a unified approach to memory- and statistical-based machine translation. In *Proceedings of the 39th Annual Meeting of the Association for Computational Linguistics (ACL'01)*, pages 378–385, Toulouse, France, July 6–11.
- Dan Melamed. 2001. *Empirical Methods for Exploiting Parallel Texts*. The MIT Press.
- Franz Josef Och, Christoph Tillmann, and Herman Ney. 1999. Improved alignment models for statistical machine translation. In *Proceedings of the Joint Workshop on Empirical Methods in NLP and Very Large Corpora*, pages 20–28, University of Maryland, Maryland.
- Kishore Papineni, Salim Roukos, Todd Ward, John Henderson, and Florence Reeder. 2002. Corpus-based comprehensive and diagnostic MT evaluation: Initial Arabic, Chinese, French, and Spanish results. In *Proceedings of the Human Language Technology Conference*, pages 124–127, San Diego, CA, March 24–27.
- Kenji Yamada and Kevin Knight. 2001. A syntax-based statistical translation model. In *Proceedings of the 39th Annual Meeting of the Association for Computational Linguistics (ACL'01)*, Toulouse, France, July 6–11.