A Discriminative Framework for Bilingual Word Alignment

Robert C. Moore
Microsoft Research
One Microsoft Way
Redmond, WA 98052
bobmoore@microsoft.com

Abstract

Bilingual word alignment forms the foundation of most approaches to statistical machine translation. Current word alignment methods are predominantly based on generative models. In this paper, we demonstrate a discriminative approach to training simple word alignment models that are comparable in accuracy to the more complex generative models normally used. These models have the the advantages that they are easy to add features to and they allow fast optimization of model parameters using small amounts of annotated data.

1 Motivation

Bilingual word alignment is the first step of most current approaches to statistical machine translation. Although the best performing systems are "phrase-based" (e.g, Och and Ney, 2004), possible phrase translations are normally first extracted from word-aligned bilingual text segments. The standard approach to word alignment makes use of various combinations of five generative models developed at IBM by Brown et al. (1993), sometimes augmented by an HMM-based model or Och and Ney's "Model 6" (Och and Ney, 2003). The best combinations of these models can produce high accuracy alignments, at least when trained on a large corpus of fairly direct translations in related languages.

These standard models are less than ideal, however, in a number of ways, two of which we address in this paper. First, although the standard models can theoretically be trained without supervision, in practice various parameters are introduced that should be optimized using annotated data. For, example, Och and Ney (2003) suggest supervised optimization of a number of parameters, including the probablity of jumping to the empty word in the HMM model, as well as smoothing parameters for the distortion probabilities and fertility probabilities of the more complex models. Since the values of these parameters affect the values of the translation, alignment, and fertility probabilities trained by EM, there is no effective way to optimize them other than to run the training procedure with a particular combination of values and evaluate the accuracy of the resulting alignments. Since evaluating each combination of parameter values in this way can take hours to days on a large training corpus, it seems safe to say that these parameters are rarely if ever truly jointly optimized for a particular alignment task.

The second problem we address is the difficulty of adding features to the standard generative models. Generative models require a generative "story" as to how the observed data is generated by an interrelated set of stochastic processes. For example, the generative story for IBM Models 1 and 2 and the HMM alignment model is that a target language translation of a given source language sentence is generated by first choosing a length for the target language sentence, then for each target sentence position choosing a source sentence word, and then choosing the corresponding target language word. When Brown et al. (1993) wanted to add a fertility component to create Models 3, 4, and 5, however, this generative story didn't fit any longer, because it does not include how many target language words to align to each source language word as a separate decision. To model this explicitly, they had to come up with a different generative story.

In this paper, we take a different approach to word alignment, based on discriminative training of a weighted linear combination of a small number of features. For a given parallel sentence pair, for each possible word alignment considered, we simply multiply the values of each of these features by a corresponding weight to give a score for that feature, and sum the features scores to give an overall score for the alignment. The possible alignment having the best overall score is selected as the word alignment for that sentence pair. Thus, for a sentence pair (e, f) we seek the alignment $\hat{a}$ such that

$$\hat{a} = \mathrm{argmax}_a \sum_{i=1}^{n} \lambda_i f_i(a, e, f)$$

where the f_i are features and the λ_i are weights.

We optimize the model weights using a modified version of averaged perceptron learning as described by Collins (2002). This is fast to train, because selecting the feature weights is the last step in building the model and the "online" nature of perceptron learning allows the parameter optimization to converge quickly. Furthermore, no generative story has to be invented to explain how the features generate the data, so new features can be easily added without having to change the overall structure of the model.

In theory, a disadvantage of a discrimintative approach compared to a generative approach is that it requires annotated data for training. In practice, however, effective discriminative models for word alignment require only a few parameters, which can be optimized on a set of annotated sentence pairs comparable in size to what is needed to tune the free parameters used in the generative approach. As we will show, a simple sequence of two such models can achieve alignment accuracy comparable to that of a combination of more complex standard models.

2 Discriminative Alignment Models

We develop two word alignment models, incorporating different word association features intended to indicate how likely two words or groups of words are to be mutual translations, plus additional features measuring how much word reordering is required by the alignment[1], and how many words are left unlinked. One of the models also includes a feature measuring how often one word is linked to several words.

Each of our feature scores have analogs in the IBM and HMM models. The association scores corresponds to word translation probabilities; the reordering scores correspond to distortion probabilities; the scores for words left unlinked corresponds to probabilities of words being linked to the null word; and the scores for one-to-many links correspond to fertility probabilities.

2.1 The Log-Likelihood-Based Model

In our first model, we use a log-likelihood-ratio (LLR) statistic as our measure of word association. We chose this statistic because it has previously been found to be effective for automatically constructing translation lexicons (e.g., Melamed, 2000). We compute LLR scores using the following formula presented by Moore (2004):

$$LLR(f, e) = \sum_{f? \in \{f, \neg f\}} \sum_{e? \in \{e, \neg e\}} C(f?, e?) \log \frac{p(f?|e?)}{p(f?)}$$

In this formula f and e mean that the words whose degree of association is being measured occur in the respective target and source sentences of an aligned sentence pair, $\neg f$ and $\neg e$ mean that the corresponding words do not occur in the respective sentences, $f?$ and $e?$ are variables ranging over these values, and $C(f?, e?)$ is the observed joint count for the values of $f?$ and $e?$. All the probabilities in the formula refer to maximum likelihood estimates. The LLR score for a pair of words is high if the words have either a strong positive association or a strong negative association. Since we expect translation pairs to be positively associated, we discard any negatively associated word pairs by requiring that $p(f, e) > p(f) \cdot p(e)$. To reduce the memory requirements of our algorithms we discard any word pairs whose LLR score is less than 1.0.

[1]We will use the term "alignment" to mean an overall word alignment of a sentence pair, and the term "link" to mean the alignment of a particular pair of words or small group of words.

In our first model, the value of the *word association feature* for an alignment is simply the sum of all the individual LLR scores for the word pairs linked by the alignment. The LLR-based model also includes the following features:

nonmonotonicity features It may be observed that in closely related languages, word alignments of sentences that are mutual translations tend to be approximately monotonic (i.e., corresponding words tend to be in nearly corresponding sentence positions). Even for distantly related languages, the number of crossing links is far less than chance, since phrases tend to be translated as contiguous chunks. To model these tendencies, we introduce two *nonmonotonicity features*.

To find the points of nonmonotonicity of a word alignment, we arbitrarily designate one of the languages as the source and the other as the target. We sort the word pairs in the alignment, first by source word position, and then by target word position. We then iterate through the sorted alignment, looking only at the target word positions. The points of nonmonotonicity in the alignment will be the places where there are backward jumps in this sequence of target word positions. For example, suppose we have the sorted alignment ((1,1)(2,4)(2,5)(3,2)(5,6)). The sequence of target word positions in this sorted alignment is (1,4,5,2,6); hence, there is one point of nonmonotonicity where target word position 2 follows target word position 5.

We still need to decide how to measure the degree of nonmonotonicity of an alignment. Two methods immediately suggest themselves. One is to sum the magnitudes of the backward jumps in the target word sequence; the other is to simply count the number of backward jumps. Rather than choose between them, we use both features.

the one-to-many feature It has often been observed that word alignment links tend to be one-to-one. Indeed, word alignment results can often be improved by restricting more general models to permit only one-to-one links. For example, Och and Ney (2003) found that the intersection of the alignments found training the IBM models in both directions always outperformed either direction alone in their experiments. Since the IBM models allow one-to-many links only in one direction, this intersection can contain only one-to-one links.

To model the tendency for links to be one-to-one, we define a *one-to-many feature* as the number of links connecting two words such that exactly one of them participates in at least one other link. We also define a *many-to-many feature* as the number of links that connect two words that both participate in other links. We don't use this directly in the model, but to cut down on the number of alignments we need to consider, we discard any alignments having a non-zero value of the many-to-many feature.

the unlinked word feature To control the number of words that get linked to something, we introduce an *unlinked word feature* that simply counts the total number of unlinked words in both sentences in an aligned sentence pair.

2.2 The Conditional-Link-Probability-Based Model

In this model we replace the LLR-based word association statistic with the logarithm of the estimated conditional probability of two words (or combinations of words) being linked, given that they co-occur in a pair of aligned sentences. These estimates are derived from the best alignments according to some other, simpler model. For example, if *former* occurs 1000 times in English sentences whose French translations contain *ancien*, and the simpler alignment model links them in 600 of those sentence pairs, we might estimate the conditional link probability (CLP) for this word pair as 0.6. We find it better, however, to adjust these probabilities by subtracting a small fixed discount from the link count:

$$LP_d(f,e) = \frac{links_1(f,e) - d}{cooc(f,e)}$$

$LP_d(f,e)$ represents the estimated conditional link probability for the words f and e, $links_1(f,e)$ is the number of times they are linked by the simpler alignment model, d is the discount, and $cooc(f,e)$ is the number of times they co-occur. This adjustment prevents assigning high probabilities to links between pairs of words that rarely co-occur.

An important difference between the LLR-based model and CLP-based model is that the LLR-based model considers each word-to-word link separately, but allows multiple links per word, as long as they lead to an alignment consisting only of one-to-one and one-to-many links (in either direction). In the CLP-based model, however, we allow conditional probabilities for both one-to-one and one-to-many clusters, but we require all clusters to be disjoint.

For example, we estimate the conditional probability of linking *not* to *ne...pas* by considering the number of sentence pairs in which *not* occurs in the English sentence and both *ne* and *pas* occur in the French sentence, compared to the number of times *not* is linked to both *ne* and *pas* in pairs of corresponding sentences. However, when we make this estimate in the CLP-based model, we do not count a link between *not* and *ne...pas* if the same instance of *not*, *ne*, or *pas* is linked to any other words.

The CLP-based model incorporates the same addtional features as the LLR-based model, except that it omits the one-to-many feature, since we assume that the one-to-one vs. one-to-many trade-off is already modeled in the conditional link probabilities for particular one-to-one and one-to-many clusters.

We have developed two versions of the CLP-based model, using two different estimates for the conditional link probabilities. One estimate of the conditional link probabilities comes from the LLR-based model described above, optimized on an annotated development set. The other estimate comes from a heuristic alignment model that we previously developed (Moore, 2005).[2] Space does not permit a full description of this heuristic model here, but in brief, it utilizes a series of greedy searches inspired by Melamed's competitive linking algorithm (2000), in which constraints limiting alignments to being one-to-one and monotonic are applied at different thresholds of the LLR score, with a final cut-off of the LLR score below which no alignments are made.

3 Alignment Search

While the discriminative models presented above are very simple to describe, finding the optimal alignment according to these models is non-trivial. Adding a link for a new pair of words can affect the nonmonotonicity scores, the one-to-many score, and the unlinked word score differently, depending on what other links are present in the alignment. Nevertheless, we have found a beam-search procedure that seems highly effective in finding good alignments when used with these models.

For each sentence pair, we create a list of association types and their corresponding scores, consisting of the associations for which we have determined a score and for which the words involved in the association type occur in the sentence pair.[3] We sort the resulting list of association types from best to worst according to their scores.

Next, we initialize a list of possible alignments with the empty alignment, assigning it a score equal to the number of words in the sentence pair multiplied by the unlinked word weight. We then iterate through our sorted list of association types from best to worst, creating new alignments that add links for all instances of the association type currently being considered to existing alignments, potentially keeping both the old and new alignments in our set of possible alignments.

Without pruning, we would soon be overwhelmed by a combinatorial explosion of alignments. The set of alignments is therefore pruned in two ways. First, we keep track at all times of the score of the best alignment we have seen so far, and any new alignment whose overall score is worse than the best score so far by more than a fixed difference D is immediately discarded. Second, for each instance of a particular alignment type, when we have completed creating modified versions of previous alignments to include that instance, we merge the set of new alignments that we have created into the set of previous alignments. When we do this merge, the resulting set of alignments is sorted by overall score, and only the N best alignments are kept, for a fixed N.

Some details of the search differ between the LLR-based model and the CLP-based model. One difference is how we add links to existing alignments. In both cases, if there are no existing links involving any of the words involved in the new link, we simply add it (keeping a copy of the original alignment, subject to pruning).

If there are existing links involving word instances also involved in the new link, the two mod-

[2]The conditional link probabilities used in the current work are those used in Method 4 of the earlier work. Full details are provided in the reference.

[3]By *association type* we mean a possible link between a pair of words, or, in the case of the CLP-based models, a possible one-to-many or many-to-one linkage of words.

els are treated differently. For the CLP-based model, each association score is for a cluster of words that must be disjoint from any other association cluster, so when we add links for a new cluster, we must remove any other links involving the same word instances. For the LLR-based model, we can add additional links without removing old ones, but the resulting alignment may be worse due to the degradation in the one-to-many score. We therefore add both an alignment that keeps all previous links, and an additional set of alignments, each of which omits one of the previous links involving one of the word instances involved in the new link.

The other difference in how the two models are treated is an extra pruning heuristic we use in the LLR-based model. In generating the list of association types to be used in aligning a given sentence pair, we use only association types which have the best association score for this sentence pair for one of the word types involved in the association. We initially explored limiting the number of associations considered for each word type simply as an efficiency heuristic, but we were surprised to discover that the most extreme form of such pruning actually reduced alignment error rate over any less restrictive form or not pruning on this basis at all.

4 Parameter Optimization

We optimize the feature weights using a modified version of averaged perceptron learning as described by Collins (2002). Starting with an initial set of feature weight values, perceptron learning iterates through the annotated training data multiple times, comparing, for each sentence pair, the best alignment a_{hyp} according to the current model with the reference alignment a_{ref}. At each sentence pair, the weight for each feature is is incremented by the difference between the value of the feature for the best alignment according to the model and the value of the feature for the reference alignment:

$$\lambda_i \leftarrow \lambda_i + (f_i(a_{ref}, e, f) - f_i(a_{hyp}, e, f))$$

The updated feature weights are used to compute a_{hyp} for the next sentence pair.

Iterating through the data continues until the weights stop changing, because $a_{ref} = a_{hyp}$ for each sentence pair, or until some other stopping condition is met. In the averaged perceptron, the feature weights for the final model are the average of the weight values over all the data rather than simply the values after the final sentence pair of the final iteration.

We make a few modifications to the procedure as described by Collins. First, we average the weight values over each pass through the data, rather than over all passes, as we found this led to faster convergence. After each pass of perceptron learning through the data, we make another pass through the data with feature weights fixed to their average values for the previous learning pass, to evaluate current performance of the model. We iterate this procedure until a local optimum is found.

Next, we used a fixed weight of 1.0 for the word-association feature, which we expect to be most important feature in the model. Allowing all weights to vary allows many equivalent sets of weights that differ only by a constant scale factor. Fixing one weight eliminates a spurious apparent degree of freedom. This necessitates, however, employing a version of perceptron learning that uses a learning rate parameter. As described by Collins, the perceptron update rule involves incrementing each weight by the difference in the feature values being compared. If the feature values are discrete, however, the minimum difference may be too large compared to the unweighted association score. We therefore multiply the feature value difference by a learning rate parameter η to allow smaller increments when needed:

$$\lambda_i \leftarrow \lambda_i + \eta(f_i(a_{ref}, e, f) - f_i(a_{hyp}, e, f))$$

For the CLP-based model, based on the typical feature values we expected to see, we guessed that 0.01 might be a good value for the learning rate parameter. That seemed to produce good results, so we did not attempt to further optimize the learning rate parameter for this model.

The situation with the LLR-based model was more complicated. Our previous experience using LLR scores in statistical NLP applications indicated that with large data sets, LLR values can get very high (upwards of 100000 for our 500000 sentence pair corpus), but small difference could be significant, which led us to believe that the same would be true of the weight values we were trying to learn. That meant that a learning rate small enough to let us converge on the desired weight values might take a very large number of iterations through the data to reach those values. We addressed this problem, by using a progression of learning rates, starting at 1000, reducing each successive weight by an order of magnitude, until we ended with a learning rate of 1.0. At each transition between learning rates, we re-initialized the weights to the optimum values found with the previous learning rate.

We experimented with one other idea for optimizing the weight values. Perceptron learning does not directly optimize error rate, but we have only a small number of parameters that we need to optimize. We therefore thought it might be helpful to apply a general optimization procedure directly to the error rate, starting from the best parameter values found by perceptron learning, using the N-best alignments found with these parameter values. We experimented with both the downhill simplex method (Press et al., 2002, Section 10.4) and Powell's method (Press et al., 2002, Section 10.5), but we obtained slightly better results with a more heuristic method designed to look past minor local minima. We found that using this approach on top of perceptron learning led to slightly lower error rates on the development set with the CLP-based model, but not with the LLR-base model, so we used it only with the former in our final evaluations.

5 Data and Methodology for Evaluation

We evaluated our models using data from the bilingual word alignment workshop held at HLT-NAACL 2003 (Mihalcea and Pedersen, 2003). We used a subset of the Canadian Hansards bilingual corpus supplied for the workshop, comprising 500,000 English-French sentences pairs, including 447 manually word-aligned sentence pairs designated as test data. The test data annotates particular pairs of words either as "sure" or "possible" links. Automatic sentence alignment of the training data was provided by Ulrich Germann, and the hand alignments of the words in the test data were created by Franz Och and Hermann Ney (Och and Ney, 2003).

Since our discriminative training approach requires a small amount of annotated data for parameter optimization, we split the test data set into two virtually equal subsets, by randomly ordering the test data pairs, and assigning alternate pairs from the random order to the two subsets. We used one of these subsets as a development set for parameter optimization, and held out the other for a final test set.

We report the performance of our alignment models in terms of precision, recall, and alignment error rate (AER) as defined by Och and Ney (2003):

$$\text{recall} = \frac{|A \cap S|}{|S|}$$

$$\text{precision} = \frac{|A \cap P|}{|A|}$$

$$\text{AER} = 1 - \frac{|A \cap P| + |A \cap S|}{|A| + |S|}$$

In these definitions, S denotes the set of alignments annotated as sure, P denotes the set of alignments annotated possible or sure, and A denotes the set of alignments produced by the method under test. Following standard practice in the field, we take AER, which is derived from F-measure, as the primary evaluation metric that we are attempting to optimize.

6 Experimental Results

We first trained the LLR-based model by perceptron learning, using an N-best value of 20 and an unbounded allowable score difference in the alignment search, using the development set as annotated training data. We then aligned all the sentences of length 100 or less in our 500,000 sentence pair corpus, using an N-best value of 20 and a maximum allowable score difference of 125000. We collected link counts and co-occurrence counts from these alignments for estimating conditional link probabilities. We trained CLP-based models from these counts for a range of values for the discount used in the conditional link probability estimation, finding a value of 0.4 to be a roughly optimal value of the discount parameter for the development set. We also trained a CLP-based model using the conditional link probabilities from the heuristic alignment model mentioned previously. In training both CLP-based models, we also used an N-best value of 20 and an unbounded allowable score difference in the alignment search.

We evaluated three models on the final test data: the LLR-based model (LLR) and the two CLP-based models, one with conditional link probabilities from

Alignment	Recall	Precision	AER
LLR	0.829	0.848	0.160
CLP_1	0.889	0.934	0.086
CLP_2	0.898	0.947	0.075

Table 1: Discriminative Model Results.

Alignment	Recall	Precision	AER
$E \rightarrow F$	0.870	0.890	0.118
$F \rightarrow E$	0.876	0.907	0.106
Union	0.929	0.845	0.124
Intersection	0.817	0.981	0.097
Refined	0.908	0.929	0.079

Table 2: IBM Model 4 Results.

the LLR-based model (CLP_1), and one with conditional link probabilities from the heuristic alignment model (CLP_2). All parameters were optimized on the development set. Recall, precision, and alignment error rates on the test set are shown in Table 1.

For comparison, we aligned our parallel corpus with IBM Model 4 using Och's Giza++ software package (Och and Ney, 2003).[4] We used the default configuration file included with the version of Giza++ that we used, which resulted in five iterations of Model 1, followed by five iterations of the HMM model, followed by five iterations of Model 4. We trained the models in both directions, English-to-French and French-to-English, and computed the union, intersection, and what Och and Ney (2003) call the "refined" combination of the two alignments. We evaluated the resulting alignments of the final test set, with the results shown in Table 2.

As these tables show, our discriminatively trained CLP-based models compare favorably to IBM Model 4 on this data set. The one with conditional link probabilities from the heuristic alignment model, CLP_2, performs slightly better than the best of the Model 4 combinations, and the one with conditional link probabilities from the LLR-based model, CLP_1, performs only slightly worse.

An interesting question is why CLP_2 outperformed CLP_1. CLP_1 is the more "principled" model, so one might have expected it to perform better. We believe the most likely explanation is the fact that CLP_2 received 403,195 link probabilities from the heuristic model, while CLP_1 received only 144,051 link probabilities from the LLR-based model. Hence CLP_2 was able to consider more possible links.

In light of our claims about the ease of optimizing the models, we should make some comments on the time need to train the parameters. Our current implementation of the alignment search is written in Perl, and is therefore quite slow. Alignment of our 500,000 sentence pair corpus with the LLR-based mode took over a day on a 2.8 GHz Pentium IV workstation. Nevertheless, the parameter optimization was still quite fast, since it took only a few iterations over our 224 sentence pair development set. With either the LLR-based or CLP-based models, one combined learning/evaluation pass of perceptron training always took less than two minutes, and it never took more that six passes to reach the local optimum we took to indicate convergence. Total training time was greater since we used multiple runs of perceptron learning with different learning rates for the LLR-based model and different conditional link probability discounts for CLP_1, but total training time for each model was around an hour.

7 Related Work

When the first version of this paper was submitted for review, we could honestly state, "We are not aware of any previous work on discriminative word alignment models." Callison-Burch et al. (2004) had investigated the use of small amounts of annotated data to help train the IBM and HMM models, but the models were still generative and were trained using maximum-likelihood methods.

Recently, however, three efforts nearly simultaneous with ours have made use of discriminative methods to train alignment models. Fraser and Marcu (2005) modify Model 4 to be a log-linear combination of 11 submodels (5 based on standard Model 4 parameters, and 6 based on additional features) and discriminatively optimize the submodel weights on each iteration of a Viterbi approximation to EM.

Liu et al. (2005) also develop a log-linear model, based on IBM Model 3. They train Model 3 using Giza++, and then use the Model 3 score of a possible alignment as a feature value in a discriminatively trained log-linear model, along with fea-

[4]Thanks to Chris Quirk for carrying out this alignment.

tures incorporating part-of-speech information, and whether the aligned words are given as translations in a bilingual dictionary. The log-linear model is trained by standard maximum-entropy methods.

Klein and Taskar (2005), in a tutorial on maximum margin methods for natural-language processing, described a weighted linear model incorporating association, position, and orthography features, with its parameters trained by a structured-support-vector-machine method. This model is in some respects very similar to our LLR-based model, using Dice coefficient association scores where we use LLR scores, and absolute position differences where we use nonmonotonicity measures.

8 Conclusions

The results of our work and other recent efforts on discriminatively trained alignment models show that results comparable to or better than those obtained with the IBM models are possible within a framework that makes it easy to add arbitrary additional features. After many years using the same small set of alignment models, we now have an easy way to experiment with a wide variety of knowledge sources to improve word-alignment accuracy.

References

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.

Chris Callison-Burch, David Talbot, and Miles Osborne. 2005. Statistical Marchine Translation with Word- and Sentences-Aligned Parallel Corpora. In *Proceedings of the 42nd Annual Meeting of the ACL*, pp. 176–183, Barcelona, Spain.

Michael Collins. 2002. Discriminative Training Methods for Hidden Markov Models: Theory and Experiments with Perceptron Algorithms. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pp. 1–8, Philadelphia, Pennsylvania.

Alexander Fraser and Daniel Marcu. 2005. ISI's Participation in the Romanian-English Alignment Task. In *Proceedings of the ACL Workshop on Building and Using Parallel Texts*, pp. 91–94, Ann Arbor, Michigan.

Yang Liu, Qun Liu, and Shouxun Lin. 2005. Log-linear Models for Word Alignment. In *Proceedings of the 43rd Annual Meeting of the ACL*, pp. 459–466, Ann Arbor, Michigan.

Dan Klein and Ben Taskar. 2005. Max-Margin Methods for NLP: Estimation, Structure, and Applications. Tutorial presented at ACL 2005, Ann Arbor, Michigan.

I. Dan Melamed. 2000. Models of Translational Equivalence. *Computational Linguistics*, 26(2):221–249.

Rada Mihalcea and Ted Pedersen. 2003. An Evaluation Exercise for Word Alignment. In *Proceedings of the HLT-NAACL 2003 Workshop, Building and Using Parallel Texts: Data Driven Machine Translation and Beyond*, pp. 1–6, Edmonton, Alberta, Canada.

Robert C. Moore. 2004. On Log-Likelihood-Ratios and the Significance of Rare Events. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, pp. 333–340, Barcelona, Spain.

Robert C. Moore. 2005. Association-Based Bilingual Word Alignment. In *Proceedings of the ACL Workshop on Building and Using Parallel Texts*, pp. 1–8, Ann Arbor, Michigan.

Franz Joseph Och and Hermann Ney. 2003. A Systematic Comparison of Various Statistical Alignment Models. *Computational Linguistics*, 29(1):19–51.

Franz Joseph Och and Hermann Ney. 2004. The Alignment Template Approach to Statistical Machine Translation. *Computational Linguistics*, 30(4):417–449.

William H. Press, Saul A. Teukolsky, William T. Vetterling, and Brian P. Flannery. 1992. *Numerical Recipies in C: The Art of Scientific Computing, Second Edition*. Cambridge University Press, Cambridge, England.