Association-Based 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 current work on statistical machine translation. Standard word-alignment methods involve the use of probabilistic generative models that are complex to implement and slow to train. In this paper we show that it is possible to approach the alignment accuracy of the standard models using algorithms that are much faster, and in some ways simpler, based on basic word-association statistics.

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" (see, for instance, Och and Ney (2004) or Koehn et al. (2003)), possible phrase translations must first be extracted from word-aligned bilingual text segments. The standard approach to word alignment makes use of five translation models defined 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 of these models can produce high accuracy alignments, at least when trained on a large parallel corpus of fairly direct translations in closely related languages.

There are a number of ways in which these standard models are less than ideal, however. The higher-accuracy models are mathematically complex, and also difficult to train, as they do not factor in a way that permits a dynamic programming solution. It can thus take many hours of processing time on current standard computers to train the models and produce an alignment of a large parallel corpus.

In this paper, we take a different approach to word alignment, based on the use of bilingual word-association statistics rather than the generative probabilistic framework that the IBM and HMM models use. In the end we obtain alignment algorithms that are much faster, and in some ways simpler, whose accuracy comes surprisingly close to the established probabilistic generative approach.

2 Data and Methodology for these Experiments

The experiments reported here were carried out using data from the workshop on building and using parallel texts held at HLT-NAACL 2003 (Mihalcea and Pedersen, 2003). For the majority of our experiments, we used a subset of the Canadian Hansards bilingual corpus supplied for the workshop, comprising 500,000 English-French sentences pairs, including 37 sentence pairs designated as "trial" data, and 447 sentence pairs designated as test data. The trial and test data have been manually aligned at the word level, noting particular pairs of words either as "sure" or "possible" alignments. As an additional test, we evaluated our best alignment method using the workshop corpus of approximately 49,000 English-Romanian sentences pairs from diverse sources, including 248 manually aligned sentence pairs designated as test data.[1]

[1]For the English-French corpus, automatic sentence alignment of the training data was provided by Ulrich Germann,

We needed annotated development data to optimize certain parameters of our algorithms, and we were concerned that the small number of sentence pairs designated as trial data would not be enough for this purpose. We therefore randomly split each of the English-French and English-Romanian test data sets 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 various alignment algorithms 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.

Our initial experiments involve algorithms that do not consider the positions of words in the sentences. Thus, they are incapable of distinguishing among multiple instances of the same word type in a sentence. We will say that these methods produce word type alignments. We compare these algorithms on the basis of the best possible alignment of word tokens given an alignment of word types. We go on to consider various ways of choosing a word token alignment for a given word type alignment, and all our final evaluations are conducted on the basis of the alignment of individual word tokens.

and the hand alignments of the words in the trial and test data were created by Franz Och and Hermann Ney (Och and Ney, 2003). The manual word alignments for the English-Romanian test data were created by Rada Mihalcea and Ted Pedersen.

3 The Log-Likelihood-Ratio Association Measure

We base all our association-based word-alignment methods on the log-likelihood-ratio (LLR) statistic introduced to the NLP community by Dunning (1993). 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 $e?$. The probabilities in the formula refer to maximum likelihood estimates.

Since the LLR score for a pair of words is high if the words have either a strong positive association or a strong negative association, we discard any negatively associated word pairs by requiring that $p(f, e) > p(f) \cdot p(e)$. Initially, we computed the LLR scores for all positively associated English/French word pairs in our 500K sentence pair corpus. To reduce the memory requirements of our algorithms we discarded any word pairs whose LLR score was less than 1.0. This left us with 12,797,697 word pairs out of a total of 21,451,083 pairs that had at least one co-occurrence.

4 One-to-One, Word Type Alignment Methods

4.1 Method 1

The first set of association-based word-aligment methods we consider permit only one-to-one alignments and do not take word position into account. The simplest method we consider uses the LLR scores to link words according to Melamed's (2000) "competitive linking algorithm" for aligning words in a pair of sentences. Since competitive linking has

no way to distinguish one instance of a particular word type from another, we operate with counts of linked and unlinked instances of word types, without trying to designate the particular instances the counts refer to. This version of competitive linking can be described as follows:

- Find the pair consisting of an English word type and a French word type that have the highest association score of any pair of words types that both have remaining unlinked instances.
- Increase by 1 the count of linked occurrences of this pair of word types, and decrease by 1 the count of unlinked instances of each of these word types.
- Repeat until no more words can be linked.

We will refer to this version of the competitive linking algorithm using LLR scores as Method 1. This is the method that Melamed uses to generate an initial alignment that he refines by re-estimation in his "Method A" (Melamed, 2000).

Method 1 can terminate either because one or both sentences of the pair have no more unlinked words, or because no association scores exist for the remaining unlinked words. We can use this fact to trade off recall for precision by discarding association scores below a given threshold. Table 1 shows the precision/recall trade-off for Method 1 on our development set. Since Method 1 produces only word type alignments, these recall and precision scores are computed with respect to an oracle that makes the best possible choice among multiple occurrences of the same word type.[2] The best (oracular) AER is 0.216, with recall of 0.840 and precision of 0.747, occurring at an LLR threshold of 11.7.

4.2 Method 2

A disadvantage of Method 1 is that it makes alignment decisions for each sentence pair independently of the decisions for the same words in other sentence pairs. It turns out that we can improve alignment accuracy by biasing the alignment method towards linking words in a given sentence that are also linked in many other sentences. A simple way to do this is to perform a second alignment based on the conditional probability of a pair of words being linked according to Method 1, given that they both occur in a given sentence pair. We estimate this link probability LP as

$$LP(f,e) = \frac{links_1(f,e)}{cooc(f,e)}$$

where $links_1(f,e)$ is the number of times f and e are linked according to Method 1, and $cooc(f,e)$ is the number of times f and e co-occur in aligned sentences.[3]

We now define alignment Method 2 as follows:

- Count the number of links in the training corpus for each pair of words linked in any sentence pair by Method 1.
- Count the number of co-occurrences in the training corpus for each pair of words linked in any sentence pair by Method 1.
- Compute LP scores for each pair of words linked in any sentence pair by Method 1.
- Align sentence pairs by competitive linking using LP scores.

Recall	Precision	Threshold
0.111	0.991	168368
0.239	0.923	71074
0.304	0.902	53286
0.400	0.838	26001
0.501	0.822	11306
0.600	0.788	4224
0.700	0.778	1141
0.800	0.765	124
0.848	0.732	1

Table 1: Recall/Precision Trade-Off for Method 1.

[2] The oracle goes through the word type pairs in the same order as the competitive linking algorithm, linking particular instances of the word types. It prefers a pair that has a sure alignment in the annotated test data to a pair that has a possible alignment; and prefers a pair with a possible alignment to one with no alignment.

[3] Melamed (1998) points out there are at least three ways to count the number of co-ccurrences of f and e in a given sentence pair if one or both of f and e have more than one occurrence. Based on preliminary explorations, we chose to count the co-occurrences of f and e as the maximum of the number of occurrences of f and the number of occurrences of e, if both f and e occur; otherwise $cooc(f,e) = 0$.

Recall	Precision	Threshold
0.100	0.887	0.989
0.230	0.941	0.982
0.301	0.952	0.967
0.400	0.964	0.938
0.501	0.967	0.875
0.600	0.967	0.811
0.705	0.948	0.649
0.816	0.921	0.441
0.880	0.775	0.000

Table 2: Recall/Precision Trade-Off for Method 2.

Recall	Precision	Threshold
0.178	1.000	0.982
0.200	0.998	0.977
0.300	0.999	0.958
0.405	0.998	0.923
0.502	0.994	0.871
0.602	0.987	0.758
0.737	0.947	0.647
0.804	0.938	0.441
0.883	0.776	0.000

Table 3: Recall/Precision Trade-Off for Method 3.

Table 2 shows the precision/recall trade-off for Method 2 on our development set. Again, an oracle is used to choose among multiple occurrences of the same word type. The best (oracular) AER is 0.126, with recall of 0.830 and precision of 0.913, occurring at an LP threshold of 0.215.

4.3 Method 3

It is apparent that Method 2 performs much better than Method 1 at any but the lowest recall levels. However, it fails to display a monotonic relationship between recall and precision as the score cut-off threshold is tightened or loosened. This seems to be due to the fact that the LP measure, unlike LLR, does not discount estimates made on the basis of little data. Thus a pair of words that has one co-occurrence in the corpus, which is linked by Method 1, gets the same LP score of 1.0 as a pair of words that have 100 co-occurrences in the corpus and are linked by Method 1 every time they co-occur.

A simple method of compensating for this over-confidence in rare events is to apply absolute discounting. We will define the discounted link probability LP_d similarly to LP, except that a fixed discount d is subtracted from each link count:

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

Method 3 is then identical to Method 2, except that LP_d is used in place of LP. We determined the optimal value of d for our development set to be approximately 0.9, using the optimal, oracular AER as our objective function.

Table 3 shows the precision/recall trade-off for Method 3 on our development set, with $d = 0.9$ and use of an oracle to choose among multiple occurrences of the same word type. The best (oracular) AER is 0.119, with recall of 0.827 and precision of 0.929, occurring at an LP_d threshold of 0.184. This is an improvement of 0.7% absolute in AER, but perhaps as importantly, the monotonic trade-off between precision and recall is essentially restored. We can see in Table 3 that we can achieve recall of 60% on this development set with precision of 98.7%, and we can obtain even higher precision by sacrificing recall slightly more. With Method 2, 96.7% was the highest precision that could be obtained at any recall level measured.

5 Allowing Many-to-One Alignments

It appears from the results for Methods 2 and 3 on the development set that reasonable alignment accuracy may be achievable using association-based techniques (pending a way of selecting the best word token alignments for a given word type alignment). However, we can never learn any many-to-one alignments with methods based on competitive linking, as either we or Melamed have used it so far.

To address this issue, we introduce the notion of bilingual word clusters and show how iterated applications of variations of Method 3 can learn many-to-one mappings by building up clusters incrementally. Consider the abstract data structure to which competitive linking is applied as a tuple of bags (multisets). In Methods 1–3, for each sentence pair, competitive linking is applied to a tuple of a bag of French words and a bag of English words. Suppose we apply Method 3 with a high LP_d cut-off threshold so that we can be confident that almost all

the links we produce are correct, but many French and English words remain unlinked. We can regard this as producing for each sentence pair a tuple of three bags: bags of the remaining unlinked English and French words, plus a third bag of word clusters consisting of the linked English and French words. To produce more complex alignments, we can then carry out an iteration of a generalized version of Method 3, in which competitive linking connects remaining unlinked English and French words to each other or to previously derived bilingual clusters.[4]

As just described, the approach does not work very well, because it tends to build clusters too often when it should produce one-to-one alignments. The problem seems to be that translation tends to be nearly one-to-one, especially with closely related languages, and this bias is not reflected in the method so far. To remedy this, we introduce two biases in favor of one-to-one alignments. First, we discount the LLR scores between words and clusters, so the competitive linking pass using these scores must find a substantially stronger association for a given word to a cluster than to any other unlinked word before it will link the word to the cluster. Second, we apply the same high LP_d cut-off on word-to-cluster links that we used in the first iteration of Method 3 to generate word-to-word links. This leaves many unlinked words, so we apply one more iteration of yet another modified version of Method 3 in which competitive linking is allowed to link the remaining unlinked words to other unlinked words, but not to clusters. We refer to this sequence of three iterations of variations of Method 3 as Method 4.

To evaluate alignments involving clusters according Och and Ney's method, we translate clusters back into all possible word-to-word alignments consistent with the cluster. We found the optimal value on the development set for the LLR discount for clusters to be about 2000, and the optimal value for the LP_d cut-off for the first two iterations of Method 3 to be about 0.7. With these parameter values, the best (oracular) AER for Method 4 is 0.110, with recall of 0.845 and precision of 0.929, occurring at a final LP_d threshold of 0.188. This is an improvement of 0.9% absolute in AER over Method 3, resulting from an improvement of 1.7% absolute in recall, with virtually no change in precision.

[4] In principle, the process can be further iterated to build up clusters of arbitrary size, but at this stage we have not yet found an effective way of deciding when a cluster should be expanded beyond two-to-one or one-to-two.

6 Token Alignment Selection Methods

Finally, we turn to the problem of selecting the best word token alignment for a given word type alignment, and more generally to the incorporation of positional information into association-based word-alignment. We consider three token alignment selection methods, each of which can be combined with any of the word type alignment methods we have previously described. We will therefore refer to these methods by letter rather than number, with a complete word token alignment method being designated by a number/letter combination.

6.1 Method A

The simplest method for choosing a word token alignment for a given word type alignment is to make a random choice (without replacement) for each word type in the alignment from among the tokens of that type. We refer to this as Method A.

6.2 Method B

In Method B, we find the word token alignment consistent with a given word type alignment that is the most nearly mononotonic. We decide this by defining the degree of nonmonotonicity of an alignment, and minimizing that. If more than one word token alignment has the lowest degree of nonmonotonicity, we pick one of them arbitrarily.

To compute the nonmonotonicity of a word token 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, primarily by source word position, and secondarily by target word position. We then iterate through the sorted alignment, looking only at the target word positions. The nonmonotonicity of the alignment is defined as the sum of the absolute values of the 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)). The sequence of target word positions in this sorted alignment is (1,4,5,2). This has only one backwards jump, which is of size 3, so that is the nonmonotonicity value for this alignment. For a complete or partial alignment, the

nonmonotonicity is clearly easy to compute, and nonmonotonicity can never be decreased by adding links to a partial alignment. The least nonmonotonic alignment is found by an incremental best-first search over partial alignments kept in a priority queue sorted by nonmonotonicity.

6.3 Method C

Method C is similiar to Method B, but it also uses nonmonotonicity in deciding which word types to align. In Method C, we modify the last pass of competitive linking of the word type alignment method to stop at a relatively high score threshold, and we compute all minimally nonmonotonic word token alignments for the resulting word type alignment.

We then continue the final competitive linking pass applied to word tokens rather than types, but we select only word token links that can be added to one of the remaining word token alignments without increasing its nonmonotonicity. Specifically, for each remaining word type pair (in order of decreasing score) we make repeated passes through all of the word token alignments under consideration, adding one link between previously unlinked instances of the two word types to each alignment where it is possible to do so without increasing nonmonotonicity, until there are no longer unlinked instances of both word types or no more links between the two word types can be added to any alignment without increasing its nonmonotonicity. At the end of each pass, if some, but not all of the alignments have had a link added, we discard the alignments that have not had a link added; if no alignments have had a link added, we go on to the next word type pair. This final competitive linking pass continues until another, lower score threshold is reached.

6.4 Comparison of Token Alignment Selection Methods

Of these three methods, only C has additional free parameters, which we jointly optimized on the development set for each of the word type alignment methods. All other parameters were left at their optimal values for the oracular choice of word token alignment.

Table 4 shows the optimal AER on the development set, for each combination of word type alignment method and token alignment selection method that we have described. For comparison, the oracle for each of the pure word type alignment methods is added to the table as a token alignment selection method. As we see from the table, Method 4 is the best word type alignment method for every token alignment selection method, and Method C is the best actual token alignment selection method for every word type alignment method. Method C even beats the token alignment selection oracle for every word alignment type method except Method 1. This is possible because Method C incorporates nonmonotonicity information into the selection of linked word types, whereas the oracle is applied after all word type alignments have been chosen.

	Oracle	A	B	C
1	0.216	0.307	0.255	0.243
2	0.126	0.210	0.147	0.109
3	0.119	0.208	0.138	0.103
4	0.110	0.196	0.130	0.098

Table 4: Development Set AER for all Methods.

The best combined overall method is 4C. For this combination, the optimal value on the development set for the first score threshold of Method C was about 0.65 and the optimal value of the second score threshold of Method C was about 0.075.

7 Evaluation

We computed the recall, precision, and AER on the held-out subset of the English-French data both for our Method 4C (using parameter values optimized on the development subset) and for IBM Model 4, computed using Och's Giza++ software package (Och and Ney, 2003) trained on the same data as Method 4C. 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 and evaluated the models in both directions, English-to-French and French-to-English, as well as the union, intersection, and what Och and Ney (2003) call the "refined" combination of the two alignments. The results are shown in Table 5. We applied the same evaluation methodology to the English-Romanian data, with the results shown in Table 6.

Alignment	Recall	Precision	AER
Method 4C	0.879	0.929	0.094
E → F	0.870	0.890	0.118
F → 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 5: English-French Results.

Alignment	Recall	Precision	AER
Method 4C	0.580	0.881	0.301
E → R	0.545	0.759	0.365
R → E	0.549	0.741	0.370
Union	0.570	0.423	0.515
Intersection	0.180	0.901	0.820
Refined	0.584	0.759	0.328

Table 6: English-Romanian Results.

Comparison of the AER for Method 4C and IBM Model 4 shows that, in these experiments, only the refined combination of both directions of the Model 4 alignments outperforms our method, and only on the English-French data (and by a relatively small amount: 16% relative reduction in error rate). Our existing Perl implementation of Method 4C takes about 3.5 hours for the 500K sentence pair data set on a standard desk top computer. It took over 8 hours to train each direction of Model 4 using Giza++ (which is written in C++). We believe that if our method was ported to C++, our speed advantage over Giza++ would be substantially greater. Previous experience porting algorithms of the same general type as Method 4C from Perl to C++ has given us speed ups of a factor of 10 or more.

Note that we were unable to optimize the many options and free parameters of Giza++ on the development data, as we did with the parameters of Method 4C, which perhaps inhibits us from drawing stronger conclusions from these experiments. However, it was simply impractical to do so, due the time required to re-train the Giza++ models with new settings. With Method 4C, on the other hand, most of the time is spent either in computing initial corpus statistics that are independent of the parameters settings, or in performing the final corpus alignment once the parameters settings have been optimized. Of the five parameters Method 4C requires, changes to three of them took less than one hour of retraining (on the English-French data – much less on the English-Romanian data), and settings of the last two need to be tested only on the small amount of annotated development data, which took only a few seconds. This made it possible to optimize the parameters of Method 4C in a small fraction of the time that would have been required for Giza++.

8 Related Work

The literature on measures of bilingual word association is too large to review thoroughly, but mostly it concerns extracting bilingual lexicons rather than word alignment. We discuss three previous research efforts that seem particularly relevant here.

Gale and Church (1991) made what may be the first application of word association to word alignment. Their method seems somewhat like our Method 1B. They use a word association score directly, although they use the ϕ^2 statistic instead of LLR, and they consider forward jumps as well as backward jumps in a probability model in place of our nonmonotonicity measure. They report 61% recall at 95% precision on Canadian Hansards data.

Obviously, we are building directly on the work of Melamed (2000), sharing his use of the LLR statistic and adopting his competitive linking algorithm. We diverge in other details, however. Moreover, Melamed makes no provision for other than one-to-one alignments, and he does not deal with the problem of turning a word type alignment into a word token alignment. As Table 4 shows, this is crucial to obtaining high accuracy alignments.

Finally, our work is similar to that of Cherry and Lin (2003) in our use of the conditional probability of a link given the co-occurrence of the linked words. Cherry and Lin generalize this idea to incorporate additional features of the aligned sentence pair into the conditioning information. The chief difference between their work and ours, however, is their dependence on having parses for the sentences in one of the languages being aligned. They use this to enforce a phrasal coherence constraint, which basically says that word alignments cannot cross constituent boundaries. They report excellent alignment

accuracy using this approach, and one way of comparing our results to theirs is to say that we show it is also possible to get good results (at least for English and French) by using nonmonotonicity information in place of constituency information.

9 Conclusions

The conventional wisdom in the statistical MT community has been that "heuristic" alignment methods based on word association statistics could not be competitive with methods that have a "well-founded mathematical theory that underlies their parameter estimation" (Och and Ney, 2003, p. 37). Our results seem to suggest that this is not the case. While we would not claim to have demonstated that association-based methods are superior to the established approach, they certainly now appear to be worth investigating further.

Moreover, our alignment method is faster than standard models to train; potentially much faster if it were re-implemented in a language like C++. Efficiency issues, especially in training, are often dismissed as unimportant, but one should consider simply the number of experiments that it is possible to do in the course of system development. In our case, for example, it was impractical to try to try to optimize all the options and parameters of the Giza++ models in a reasonable amount of time, given the computational resources at our disposal.

While the wealth of details regarding various passes through the data in our best methods might seem to undercut our claim of simplicity, it is important to realize that each of our methods makes a fixed number of passes, and each of those passes involves a simple procedure of computing LLR scores, collecting co-occurrence counts to estimate link probabilities, or performing competitive linking; plus one best first search for minimally nonmonotonic alignments. All these procedures are simple to understand and straightforward to implement, in contrast to some of the difficult mathematical and computational issues with the standard models.

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.

Colin Cherry and Dekang Lin. 2003. A Probability Model to Improve Word Alignment. In *Proceedings of the 41st Meeting of the Association for Computational Linguistics*, pp. 88–95, Sapporo, Japan.

Ted Dunning. 1993. Accurate Methods for the Statistics of Surprise and Coincidence. *Computational Linguistics*, 19(1):61–74.

William A. Gale and Kenneth W. Church. 1991. Identifying Word Correspondences in Parallel Texts. In *Proceedings of the Speech and Natural Language Workshop*, pp. 152–157, Pacific Grove, California.

Philipp Koehn, Franz Joseph Och, and Daniel Marcu. 2003. Statistical Phrase-Based Translation. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics (HLT-NAACL 2003)*, pp. 127–133, Edmonton, Alberta, Canada.

I. Dan Melamed. 1998. Models of Co-occurrence. University of Pennsylvania, IRCS Technical Report #98-05.

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.

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