Improving bilingual alignment models: Cognate identification, length dependence, and phrases

Andrea Burbank and Dinkar Gupta

June 2006

Part I

Cognate matching and word-length dependence

1 Introduction

Determining exactly how words in a French sentence correspond to their counterparts in an English translation is an essential component of a machine translation system. For example, given the sentences *Je suggère que tu arrives à l'heure* and *I suggest you arrive on time*, we might hope to align *je* to *I*, *suggère* to *suggest*, and so forth. The IBM Models use pure distributional statistics to determine the likelihood that any given pair of words is a translation pair, counting cumulative probabilities of cooccurrence across a set of sentences used for training. In Model 1, estimates are based only on cooccurrence in a sentence pair; Model 2 builds on this, incorporating the idea that phrases close together in English tend to be close together in French by penalizing alignments that match very distant components in the sentence. We also use an improved version of Model 1,[1] with increased null fertilities and simple smoothing, that improves performance by a few percent over Model 1. Details of its implementation will not be discussed here, as they were detailed in our previous paper.[2]

In this paper, we explore how these models can be improved by looking at the words themselves in the specific case of French-English word alignment. Although a few previous papers have explored cognate matching,[3] they have focused on sentence alignment or touched only briefly on the motivations behind their choices. Here, we explore extensively what types of cognate matching work best under what circumstances, and add a word-length dependence we haven't otherwise seen in the literature. We first explore a wide variety of cognate-matching techniques, then analyze the addition of a word-length-dependence model on model performance. Because French and English share many word roots and have similar structure, these techniques can provide a significant performance boost.

[1] cf. Moore, Robert C. 2005. Association-Based Bilingual Word Alignment. In Proceedings, Workshop on Building and Using Parallel Texts: Data-Driven Machine Translation and Beyond, Ann Arbor, Michigan , pp. 1-8. We use a null fertility of 25 and add-one smoothing.

[2] Burbank and Gupta, "Bilingual word alignment and maximum entropy classification."

[3] cf. Kondrak et al.

2 Cognate matching techniques

Because so many words in English came from French, there are many words that resemble each other in the two languages. In addition to recognizing these word pairs, a cognate recognizer can serve in effect as an aligner for proper nouns, which tend to translate from one language to another unchanged. IBM Models 1 and 2, based only on statistical distributions, tended to overalign many words, even common words, to unusual words that had been seen only once or twice in the training data. These uncommon words were often proper nouns. In addition to the obvious solution of matching related words in French and English, then, a cognate-based approach can do an elementary form of named entity recognition, correctly aligning rare proper nouns from English to French and preventing spurious alignments. Based on the preliminary success of an ad hoc cognate model based on a normalized count of shared letters, we implemented models for cognate matching based on four different criteria. Each model was restricted to identifying possible cognates within a given sentence pair, such that minimal cooccurrence restrictions were enforced. Multiple possible cognate matchings were allowed for any given French or English word, as makes sense given the morphologies associated with each (e.g *engage* could translate as *engagent*, *s'engager* or *engageons* according to context).

Naive

The first was our experimental version, dubbed "naive" cognate matching, which simply counts the number of letters in the English word that are also in the French word and vice versa and divides by word length. These percentages of shared letters are then averaged and normalized by $e^{-|l-m|}$, where l is the length of the French word and m the length of the English word. Words shorter than four letters were excluded from the model unless they contained non-alphabetic characters. For the naive model, the parameter that determines what word pairs are marked as cognates is the minimum value t_{naive} that a score must exceed to mark a pair as cognates.

Simard

We next implemented a cognate-identification model created by Michel Simard et al. that considers a pair of words to be a cognate pair if their first four letters are identical. Although this fails to recognize such cognate pairs as *erreur* and *error*, it nonetheless provides a good metric for matching words between French and English. We also implemented a modified version of Simard's metric, which ignores capitalization and, like the naive model, matches words shorter than four letters if they contain a non-alphabetic character. The Simard model, although very simple, is also of linguistic interest in that it effectuates stemming, ignoring differences in endings due to different conjugations or other derivational and inflectional morphology. For example, the Simard method considers *enchanter*, *enchantent*, *enchantons* and other conjugations all to be potential cognate matches for English *enchant*, just as would morphologically based stemming.

Dice

Our third cognate-matching model was based on the Dice coefficient. The Dice coefficient computes the number of bigrams shared between the two words and divides by the total number of bigrams in both words. For example, *erreur* and *error* share 2 bigrams (twice each) of a total of 9, for a Dice coefficient of 4/9. Like the naive model, the Dice metric computes a score for each cognate pair, and a threshold must thus be established for the minimum correspondence of a pair to be considered cognates. We experimented with two different thresholds: one, a length-dependent threshold hand-tuned to be about 93% accurate in its identification of cognates, and the other, a simple threshold t_{dice} for all words of length greater than 3 and perfect matching (a minimum score of 1) for words shorter than three letters long.

LCSR

The fourth cognate-identification model uses the Longest Common Subsequence Ratio, or LCSR, as a measure of word similarity. The LCSR is defined as the ratio of the length of the longest common subsequence of two words to the length of the longer word. For example, given *recommandent* and *recommend*, the LCSR is $8/12 = 2/3$, as the words share the subsequence "r-e-c-o-m-m-n-d." As with the Dice coefficient, a threshold t_{LCSR} determines the minimum score for a cognate pair. Again, we require all words shorter than four letters to be completely identical, and use the threshold for words of length four and longer. The LCSR is, in effect, very similar to the Levenshtein edit distance metric used in spell-checking, which uses a similar dynamic programming algorithm to determine the minimum number of insert, substitute and delete operations required to transform one string into another. We also implement an edit-distance-based cognate-matching model, but as it is very similar to the LCSR metric but performs slightly less well, we did not explore it as extensively.

3 Optimizing parameters for cognate matching

The first question, given these various cognate-matching models, was how reliable a list of cognates each produced as a function of its parameter t (where relevant). To get a sense of this, we trained fifteen different models on the test set of 447 sentences, which contained 1939 distinct French words. The resulting number of cognate pairs, precision, recall, and alignment error rate (AER) are shown in the table at right. Clearly, as the threshold for the Dice and LCSR models was increased from 0.4 to 0.95, the number of cognate pairs decreased dramatically and, in general, the model's accuracy, in terms of correctly identifying cognates (measured via the precision) increased. The lowest AER was achieved by the LCSR model with a threshold of 0.5, which was about 80% accurate in identifying cognates and achieved a 35% recall rate, remarkable for a model that matched only cognate pairs with no information on the global frequency of cooccurrence.

Model	# Cognates	Precision	Recall	AER
dice-0.95	198	0.964	0.065	0.877
lcsr-0.95	213	0.881	0.218	0.649
dice-0.9	220	0.964	0.071	0.867
lcsr-0.9	231	0.882	0.222	0.644
naive-0.86	244	0.949	0.242	0.613
dice-0.8	266	0.967	0.082	0.845
lcsr-0.8	337	0.899	0.267	0.586
dice-0.7	361	0.971	0.115	0.789
simard	415	0.964	0.109	0.792
naive-0.7	447	0.863	0.262	0.586
dice-0.6	458	0.961	0.144	0.742
dice-lendep	486	0.979	0.150	0.731
lcsr-0.7	494	0.896	0.307	0.539
modsimard	497	0.917	0.283	0.559
dice-0.5	571	0.936	0.165	0.709
lcsr-0.6	692	0.856	0.333	0.514
dice-0.4	748	0.877	0.183	0.682
lcsr-0.5	942	0.796	0.348	0.507
lcsr-0.4	2744	0.499	0.389	0.556

Given these baseline statistics for the cognate-matching models' accuracy and breadth, the next question was how to incorporate cognate matching into Models 1 and 2. After all, cognate matching alone would be unable to match even simple pairs like *la* and *the* that cooccur hundreds or thousands of times, so word-distribution statistics must be used to complement the word-specific information garnered from cognate matching. One option is to use the cognate model's output to tweak the input probabilities for Model 1. In general, Model 1 is initialized with uniform probabilities; that is $P(f|e) = 1/N$, where N is the number of distinct French words. Instead of doing this, the parameters may be initialized with a strong bias toward a high probability for the cognate pairs. However, the EM algorithm is robust to changes in initial parameters. Because the optimization function is convex, the EM algorithm will always produce the same parameters regardless of initial values, given enough iterations. We did few enough iterations that the cognates had an effect, but although these changes improved performance, we sought more reliable ways to incorporate cognate matching into Models 1 and 2.

The simplest, and most logical, way to incorporate an automatically generated (and thus not

100% accurate) set of cognates[4] is to append them as extra sentences onto the training set. Then their cooccurrence increases dramatically, prompting Model 1 to reinforce their connection, where valid, but not require it, where invalid. The question becomes how many times to append each cognate pair. Listing them once will provide some guidance to Model 1, but appending them twice or ten times would reinforce the connections that much more. We thus experimented with the optimal number of repeats for each cognate pair in the training set, from as few as one copy of each cognate pair to as many as twenty. The results, for a variety of different models, are shown in Figure 1. For most of the models, it seems that appending each cognate pair about five times

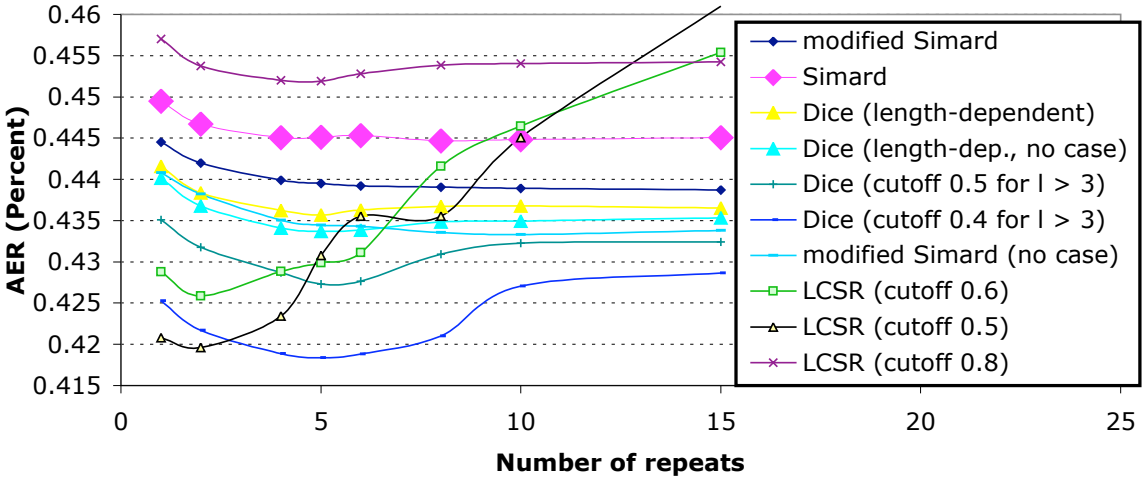


Figure 1: The AER as a function of the number of times a cognate pair is appended to the training set for various cognate-matching models.

is optimal, with the notable exception of the LCSR model with threshold $t_{LCSR} = 0.5$, which was close to being the best model and which gave the best results with fewer repeats. This makes sense: the LCSR–0.5 model performs best because, as evident in the table above, it identifies correctly many cognates for a good balance of precision and recall, but because it has lower precision than most of the other high-performing models, repeating the cognate pairs five times over-enforces the incorrect cognate alignments. We thus chose to run all the cognate models with both 5 repetitions and 1 repetition, to explore how many appendings were optimal on different quantities of training data.

4 Word-length dependence

One of the largest problems with the IBM models' performance is their tendency toward "garbage collection."[5] Many words, especially common words (*de*, *la*) align to words seen only very infrequently in the English training data. Although cognate matching is one solution that works for proper nouns and for uncommon words that happen to be cognates, this problem can also be addressed by the introduction of word-length dependence. In part because French and English are related, and in part because function words tend to be short, the likelihood of aligning

[4]If the list of cognates were known to be comprehensive and highly accurate, as from a dictionary, alignments could be made directly to the cognate pairs in the alignment step. With imperfect pairs, however, it is better to use the robustness of the training algorithm to reinforce, but not force, potential connections.

[5]cf. Brown et al. "But Dictionaries are Data Too." Human Language Technology Conference, Princeton, NJ, 1993.

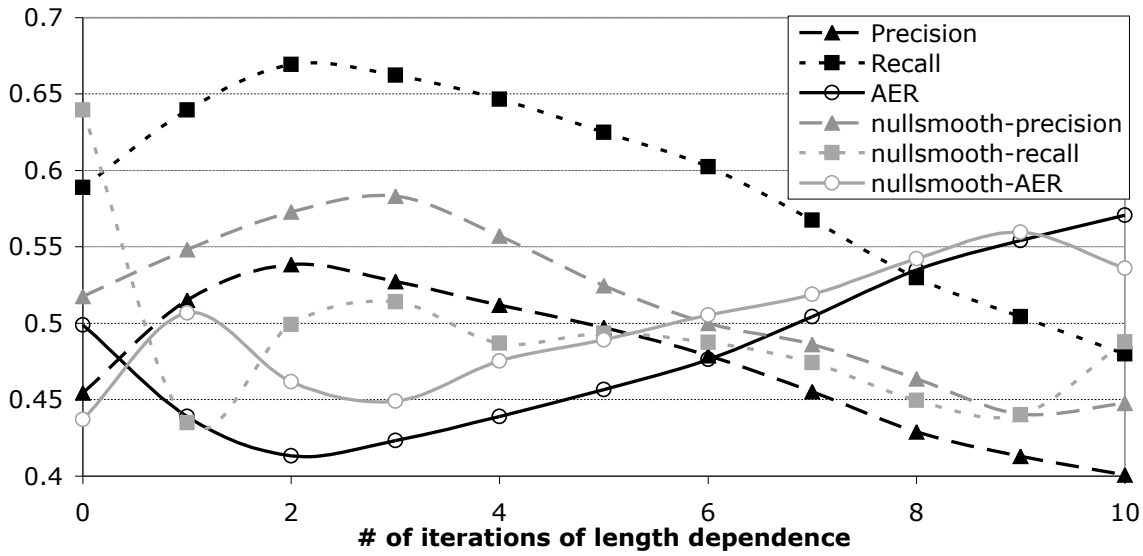


Figure 2: Precision, recall, and AER of Model 1 as a function of the number of iterations of length-dependence probability. Model 1 performance is shown in black; performance of the improved Model 1 with increased null fertility is shown in gray.

words with highly disparate lengths is significantly lower than the probability of aligning words whose lengths are similar.

To model this probability distribution, we added a word-length probability model to Model 1, with parameters iteratively tuned much as the translation parameters are. The model stores $P(l|m)$ for every English word length m and French word length l up to a certain length l_{max}. We chose $l_{max} = 20$ as the great majority of words are shorter than 20 letters, and initialized $P(l|m)$ to $1/l$ for all pairs (l, m). As alignments are cumulatively counted, they are weighted by $P(l|m)$, and the resulting cumulative counts in turn recompute $P(l|m)$ in each iteration of Model 1.

As with the distortion probabilities in Model 2, iterative readjustment of these weights tends to overfit, generating $P(l|m) \approx 0$ for $l \neq m$ if the model is allowed to proceed for too many iterations. The performance of Model 1 as a function of the number of iterations in which the length-dependence probabilities were refined, is shown in Figure 2.

Adding length dependence clearly improves performance for Model 1 both with and without added null fertilities. However, aware that the effects of this parameter might change with the size of the training data, we maintained n, the number of iterations for the length-dependency probabilities, as a tunable parameter.

5 Results

Both cognate matching and word-length dependence significantly improved word alignment performance. Although increased precision in the cognate-matching model alone generally comes at the expense of decreased recall, the better cognate-matching models simultaneously improved the precision and recall of Models 1 and 2. When little training data was available, cognate-matching models could produce a very significant improvement in alignments, decreasing the alignment error rate from 43.7% to 36.4%. Interestingly, the greatest improvement came not from a highly accurate cognate-matching model that produced almost exclusively correct cognate pairs, but rather a model with only mediocre precision($\sim 80\%$) that produced a large volume of possible cognate pairs. The

resultant increase in the size of the training set outweighed the influence of incorrect cognate pairs.

As the size of the training set increased, slightly more accurate models — but still models that identified a relatively high number of cognate pairs — produced the lowest error rates. The performance of Model 1 as a function of the number of cognate pairs identified per sentence for a variety of training sizes is shown in Figure 3. Clearly, for all training sizes, the performance of

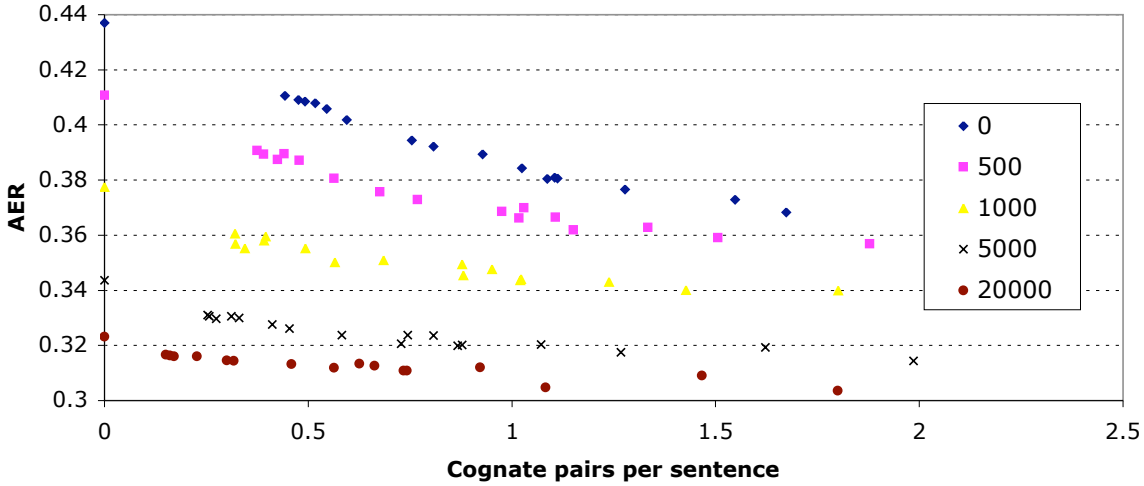


Figure 3: AER as a function of the number of cognates per sentence on training sets of various sizes

Model 1 improves as more cognates are added, even though many may not be correct pairs. This is, of course, true only to a point. We experimented with a threshold of 0.4 for the LCSR model and found that, while results on a small training set were about in the middle of the pack, performance lagged dramatically behind that of other models when the training size increased. The cognate-matching models that showed the best performance at all training sizes were LCSR with threshold 0.5 and 0.6. These models represented a strong balance between accuracy of cognates identified and the number of cognates generated. The highly accurate Dice model with a length-dependent cutoff designed to produce about 93% accurate cognates also fared well, especially on larger data sets. The overall improvement gained by adding cognates is shown below. Multiple repetitions of each cognate pair significantly improved performance on larger training sets, a reasonable result since the number of cognate examples is otherwise negligible in comparison to the number of sentences.

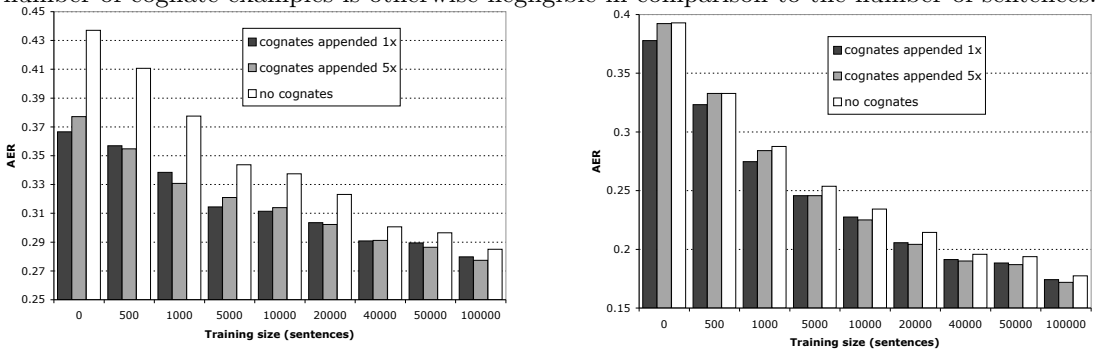
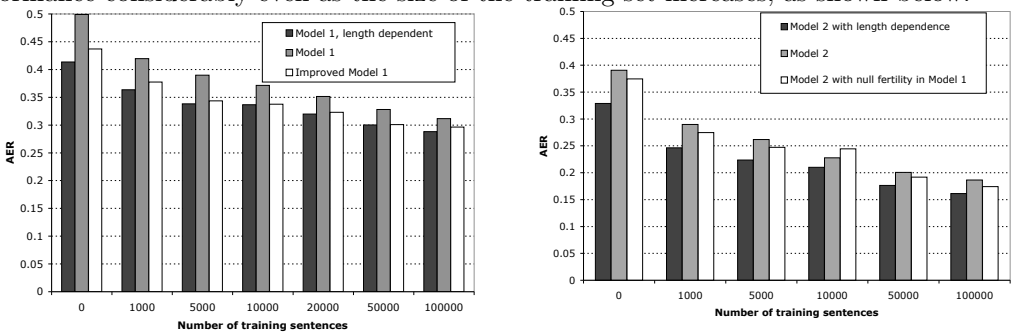


Figure 4: Performance of the models with and without cognates, with one or five repetitions of each cognate pair appended. On the left, Model 1; on the right, Model 2.

Adding length dependence to Model 1 also significantly improved performance. Figure 2 shows

a significant increase in precision and recall after 2 iterations of tuning the length-dependence probabilities. Further exploring this trend, we determined that adding length dependence improves performance considerably even as the size of the training set increases, as shown below.



The length dependency shows considerable improvement over Model 1 for all sizes of the training set, but is comparable in performance to the improved Model 1 as the training set size increases. In Model 2, the addition of word-length dependence to Model 1 significantly improves results, even in comparison to the smoothed version of Model 1 with additional NULL tokens.

The optimal number of iterations for tuning the length-dependence probabilities depends both on the model and on the number of training sentences, as shown below.

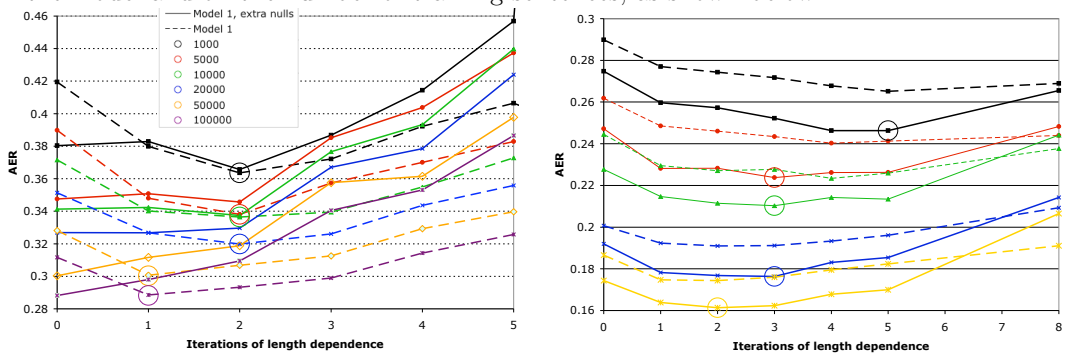


Figure 5: AER as a function of the number of iterations of length dependence for Model 1 (left) and Model 2 (right). Solid lines indicate scores obtained using the improved Model 1; dashed lines indicate scores using Model 1 without extra bells and whistles. Line color indicates the number of sentences on which each model was trained: black, 1000; red, 5000; green, 10000; blue, 20000; orange, 50000; purple, 100000.

Model 1 achieves optimal performance with one or two iterations of length-dependence parameter tuning, and the number of iterations required for a significant performance improvement decreases as more training sentences are added. Model 2 requires more iterations of length-dependence tuning to achieve its best performance, and the optimal number of iterations again decreases as more training sentences are added. This makes sense: with more training data, parameters incorporate more distinctions with each iteration, so fewer iterations are needed before overfitting begins.

The cognate-matching and length-dependence criteria provide complementary information, so running both can improve performance even further. When run on the test set, Model 1 achieves a precision of 45.4%, recall of 58.9% and AER of 49.9%. Adding cognate matching (LCSR, threshold 0.5) improves these numbers to 52.0% precision, 68.8% recall, and an AER of 42.2%, while length dependence (2 iterations) provides similar improvements: precision 53.8%, recall 66.9%, and AER 41.3%. When both cognate matching and word-length dependence are used simultaneously, Model 1 achieves 58.6% precision and 72.5% recall for an AER of 36.2%, a level of performance that Model 1 alone requires over 10,000 additional training sentences to achieve.

6 Conclusion

Both cognate matching and word-length dependence introduce significant improvements to French-English translation using word distribution models. Improvements on the order of 25% are achieved for models with limited quantities of training data, and models trained on 100,000 sentences improve by about 10% when these word-dependent metrics are added to Model 1. The changes are largely minor, but noticeable: *la* is no longer aligned to *confedération*, *commenter* is aligned to *comment*, *Orateur* is correctly aligned to *Speaker*, and so forth. Occasionally the length dependence is detrimental, as when the *un* of *quelque un* is no longer aligned to *anybody*, but this happens only rarely (and in this case, is largely a result of the tokenization that separated what is arguably a single French word into two). The garbage collection problem is significantly reduced, with proper nouns and other rare words receiving fewer spurious alignments. Because cognate identification and length-dependence occur during the training of Model 1, the benefits of diagonalization in Model 2 are realized with the added information of cognate matching and other more accurate word alignments generated by Model 1. For closely related languages like French and English, then, generating lists of cognates, even not entirely accurate ones, and conditioning probabilities on the lengths of corresponding words can significantly improve performance.

Part II
Phrase alignments

1 Introduction

Word alignment models try to determine correspondences between words in English and French, but word alignments are sometimes not possible between semantically equivalent phrases. For instance, the French phrase **f** (*ne veux jamais*) translates to the English phrase **e** (*never want*). However, all word alignments between f and e are ambiguous. This translation is best modeled as a high-level mapping of **f** to **e**.

The IBM translation models for translating French to English capture this concept partially by allowing one-to-many mappings from English words to French words. However, they do not encompass the phrase-to-phrase case described above. Various phrase translation schemes have been proposed and have demonstrably translated better than the IBM Models (cf. Koehn, Marcu). We therefore decided to investigate a phrase alignment model to see if it would improve sentence alignment quality over the purely word-alignment based IBM models.

2 Implementation

The model we implement is derived from Venugopal et al. We use our cognate-enhanced IBM Model 1 from Part 1 as the underlying lexical (word-to-word) alignment model. The model aligns phrases from a French sentence with phrases from an English sentence sequentially. Corresponding words in English and French may not be in identical positions in their respective sentences (the sentences may even be of different lengths), but we assume that the sentences' conceptual structure is the same, allowing phrases to be matched sequentially from left to right.

The original model in Venugopal et al. is extremely computationally intensive for even training sets containing around 1000 sentences (prohibitively so in our case). We thus simplify the model somewhat for computational tractability, enabling us to train on sets of up to 10,000 sentences.

2.1 Training

The training phase consists of the following specific sub-parts: phrase-mapping hypothesis generation, hypothesis scoring, hypothesis pruning, and phrase translation probability distribution generation.

2.2 Phrase-mapping Hypothesis Generation

A priori we do not know the phrase map between the languages. Hence we hypothesize that all possible combinations of English n-grams to French m-grams (where $2 \leq n \leq N$, $1 \leq m \leq M$, and M, N parameters that may be tuned for computational tractability) are feasible. We also store the distortion between a potential translation of the French phrase **f** and English phrase **e**, computed as: $dist(e, f) = position(\text{start of } \mathbf{f} \text{ in French sentence}) - position(\text{start of } \mathbf{e} \text{ in English sentence})$ The distortion is lower-bounded by a parameter D to enforce a right to left consistency between the languages.

For instance, given a French and English sentence pair: [X Y Z] → [A B C] with $N = M = 2, D = 0$, we would generate phrase pairs (and distortions) as below:

[X] → ([A], 0), ([B], -1), ([C], -2), ([A B], 0), ([B C], -1)

[X Y] → ([A], 0), ([B], -1), ([C], -2), ([A B], 0), ([B C], -1)

...

A phrase-mapping hypothesis is a unique combination of the French phrase **f**, the English phrase **e** and the distortion between the starting indices, d. From here on, we will use the notation (**e**, **f**, d) for a phrase mapping hypothesis, and will refer to it as a phrase hypothesis. For source phrases of length 1 (i.e. words) we fall back to our lexical model.

2.3 Hypotheses Scoring

We generate 4 types of scores for each hypothesis, as suggested in [VVW]: "within sentence" consistency score, cross sentence consistency score, lexical validity score and a length ratio score.

1. Within sentence consistency score

   This score measures the consistency of a phrase hypothesis with the word alignments provided by the lexical model. We train a lexical model M1 for alignments from French to English and M2 for the reverse direction. For each sentence pair {**E**, **F**}, M1 and M2 generate the Viterbi alignments V1={**F** → **E**} and V2={**E** → **F**} that are used to generate an alignment map M = V1∪V2. For a given phrase hypothesis (**e**, **f**, d), we generate a dense phrase level alignment map C that contains elements (n, m) where n ranges over the positions of the words in e_1 in the original English sentence, and m ranges over the positions of the words in f_1 in the original sentence. The "within sentence" consistency score of (**e**, **f**, d) is computed as $|M \cup C|/|M - C|$.

2. Cross Sentence consistency score

   The cross sentence consistency score measures the consistency of hypothesized English translations e_1, ..., e_N for a given French phrase f. The cross consistency score for the hypothesis (e_i, **f**, d) is given by $N/\sum_k d(e_i, e_k)$, where $d(e_i, e_k)$ computes the Levenshtein distance between e_i and e_k. Hypotheses that repeat frequently (even with minor spelling changes) will have high cross sentence consistency scores.

3. Lexical score

   Given a phrase hypothesis (**e**, **f**, d), the lexical score computes the actual probability of the words in **e** mapping to those in **f**, based on the lexical model. More precisely, the lexical score

is given by

$$\prod_{i \le x \le l_s} \sum_{j \le y \le l_t} p(f_x|e_y) \text{ where } x \text{ and } y \text{ are indices in French and English sentences}$$

and l_s and l_t are the lengths of the source and target sentences, respectively.

4. Length Difference Ratio score

   To penalize hypotheses that align phrases of extremely disparate lengths, we try to estimate typical phrase-length ratios between English and French. Since we do not have a phrase aligned corpus, we simply learn a Gaussian distribution of the English and French sentence lengths, based on the sentence pairs in the training set. Every hypothesis (**e**, **f**, d) is then assigned a score according to the ratio of $(l_s - l_t)/l_t$.

5. **Final Score** The final score is computed according to:

$$Score(\mathbf{e}, \mathbf{f}, d) = \prod_i (Score_i(\mathbf{e}, \mathbf{f}, d))^{w_i} \text{ where } i \text{ are the four different scores and } \sum w_i = 1$$

2.4 Hypothesis Pruning

Maximal Separation pruning

Prior to generating the French to English phrase probability distribution, we use a maximal separation criteria for pruning low scoring hypotheses for any given French phrase **f**. Intuitively, if there are a lot of hypotheses for **f**, and they naturally split into one low-scoring and one high-scoring cluster, we only keep the high scoring cluster.

Aggressive Hypothesis pruning

The cross-consistency scoring step technically requires having generated all hypotheses for a French phrase **f**. This requires reading in the entire training set and generating potentially millions of hypotheses. Given our limited computational resources, cross-consistency scoring therefore is prohibitively expensive in terms of time and memory requirements. However, most of the hypotheses score badly according to the other 3 computationally inexpensive metrics. We thus generate all hypotheses iteratively. At the start of each iteration the lengths of English and French phrases of the hypotheses to be generated are fixed. Each new hypothesis is scored using only the "within sentence" consistency score, the lexical score and the length difference ratio score. Then the hypotheses for each French phrase are sorted according to their scores. All hypotheses below the 75th percentile score are discarded. Even with this level of aggressive pruning, we were forced to back up each aggressively pruned set of hypotheses on disk to avoid exceeding the memory capacity of our machines. Only after all iterations of hypothesis generation could we read in all the pruned hypothesis sets to compute cross-consistency scores.

2.5 Probability Distribution Generation

The conditional probability distribution $P(e|f)$ is given directly by normalizing the scores of the remaining (unpruned) hypotheses for the French phrase **f**.

2.6 Tuning scoring weights

We tuned parameter weights on a validation set of 5,000 sentence pairs, experimenting with different values of w_i for the four different scoring methods. The results are shown below:

Within	cross	lexical	length	Precision	Recall	AER
0.3	0.2	0.2	0.3	0.8690	**0.6612**	0.2089
0.25	0.25	0.25	0.25	0.8629	0.6454	0.2187
0.4	0.1	0.1	0.4	0.8648	0.6464	0.2171
0.4	0.1	0.4	0.1	0.8669	0.6558	0.2121
0.5	0.2	0.1	0.2	0.7946	0.6404	0.2694
0.5	0.3	0.1	0.1	**0.8701**	0.6573	**0.2093**
0.1	0.1	0.1	0.7	0.8518	0.6550	0.2226
0.1	0.1	0.2	0.6	0.8642	0.6474	0.2171
0.1	0.1	0.3	0.5	0.8632	0.6548	0.2149
0.1	0.1	0.4	0.4	0.8621	0.6523	0.2167
0.1	0.1	0.5	0.3	0.8607	0.6545	0.2166
0.1	0.1	0.6	0.2	0.8194	0.6535	0.2482
0.1	0.1	0.7	0.1	0.8601	0.6508	0.2188

The highest performance metrics are highlighted. Based on these results, we used $w_i = \{0.5, 0.3, 0.1, 0.1\}$ for performance comparison with word-based models. We were little surprised to see that the cross consistency sentence score weight was this low since it seems to be a very important metric for phrase alignments. It is possible that the hypotheses were not getting reinforced as much as we had hoped because too many of them were getting pruned in the aggressive phrase.

3 Generating Alignments

3.1 Best Alignment Probability

Phrase alignments probabilities are generated using dynamic programming. Given a French-English sentence pair {**F**, **E**}, the probability of the best match up to (and including) the words (e_i, f_j) is given by the recursive relationship:

$$\max P(e_i, f_j) = \max_{m,n} P(E(i-m), F(j-n)) \cdot \max\left(P(e)i-1, f_{j-1}\right)$$

where $E(i-m)$ is the $< i, m >$ substring of $E =< e_m, e_{m+1}, ..., e_i >$ and $F(j-n)$ is the $< j, n >$ substring of $F =< f_n, f_{n+1}, ..., f_j >$.

This can be computed in polynomial time using a dynamic programming approach by computing the best probability of matching a phrase up to words (e_m, f_n) starting at $m \in \{0, 1, ..., M-1\}$ and $n \in \{0, 1, ..., N-1\}$, where len(**E**) = M, len(**F**) = N. The best match sequence can then be extracted by backtracking along the highest probability path.

3.2 Computing the Precision, Recall, and AER

For purposes of evaluation, we need to compare proposed phrase matches (**e, f**) in a sentence pair (**E, F**) to the reference alignment for (**E, F**). If a phrase (**e, f**) is aligned correctly, then all the words inside the phrase can be assumed to align correctly—for even if they didn't line up quite right in the word-based model, a translation model using the phrases we align would produce a perfect translation. We thus mark all alignments within correctly identified phrases as identified by our model.

If however, the reference alignment contains no sure or possible alignment between a proposed phrase match (**e, f**), we count $\min(l_e, l_f)$ as the number of bad proposed aligments. The choice of using the minimum length of the phrase pair is optimistic, but we hope that by consistently doing this we would still be able to get comparable performance numbers when the phrase alignment

model is trained on different datasets and run with different parameter values. For comparison, we also compute accuracy values using $\max(l_e, l_f)$, as the number of bad proposed alignments.

4 Results

For the purposes of computational tractability we limited French phrase lengths from 1 to 4 words and English phrase lengths from 2 to 4 words. Also, the distortion parameter for relative negative displacement between hypotheses was set to 0. As we discuss below, the entire hypothesis set was still enormously large, and we were unable to train and run the model with 20,000 and 50,000 sentences.

This phrase model turned out to be computationally intensive. Even with our disk-backed, IdentityHashMap backed, multi-phased approach, we were not able to run on more than 10,000 additional training sentences. With just 5,000 training sentences, the process would consistently take up close to 1GB of memory and hence run very slowly. Our aggressive pruning was necessary to limit the huge number of hypotheses generated, as shown below. The number of hypotheses

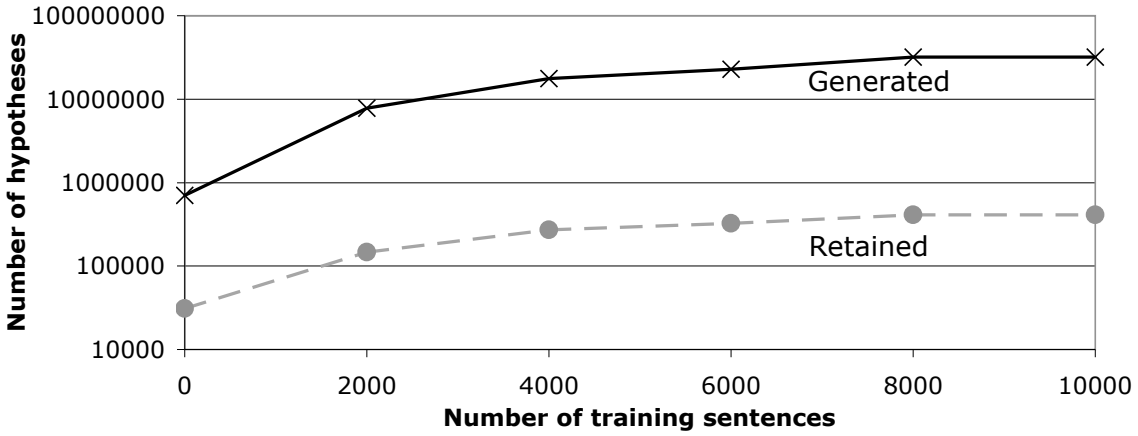


Figure 4: The number of hypotheses generated and retained as a function of the training set size. Note that the y-axis is on a logarithmic scale.

grow linearly as the training data set increases, but is already on the order half a million for 10,000 training sentences. Notably, for all training set sizes we need to prune hypotheses by two orders of magnitude for computational reasons.

As mentioned above, in case of a proposed phrase alignment ($\mathbf{e}$, $\mathbf{f}$) that does not have any match in the reference alignment for the sentence pair (E, F) we compute the number of mismatches as either:

1. $\min(l_e, l_f)$ — optimistic scoring method
2. $\max(l_e, l_f)$ — pessimistic scoring method

To determine the effect of the choice between these methods, we compare the AER values generated for the same training data sizes when using the optimistic and pessimistic scoring methods.

The AER increases significantly when using the pessimistic scoring method. This is to be expected, since we overly penalize the bad alignments between phrases of differing lengths. Arguably the optimistic (min()) scoring method under-penalizes bad matches. Both models significantly outperform the word-based Model 1, however, and we continue to use the optimistic scoring method as a metric of alignment quality.

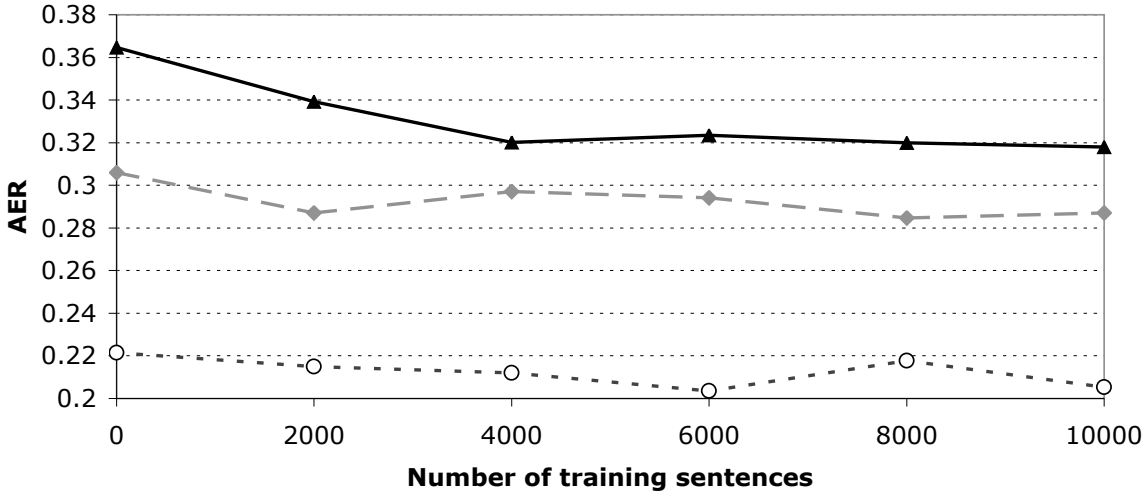


Figure 5: AER for Model 1 (solid black line), phrase alignments with pessimistic scoring (gray dashed line), and phrase alignments with optimistic scoring (dark gray circles with dotted line) as a function of the size of the training set.

Adding phrase alignments nonetheless significantly improves the alignment error rate. Although the algorithm does not identify many of the phrases that could be identified, the phrases it does identify are highly accurate. The precision, recall and AER of the phrase-based model are shown in comparison with the cognate-enhanced Model 1 results in Figure 6 for a variety of training sizes.

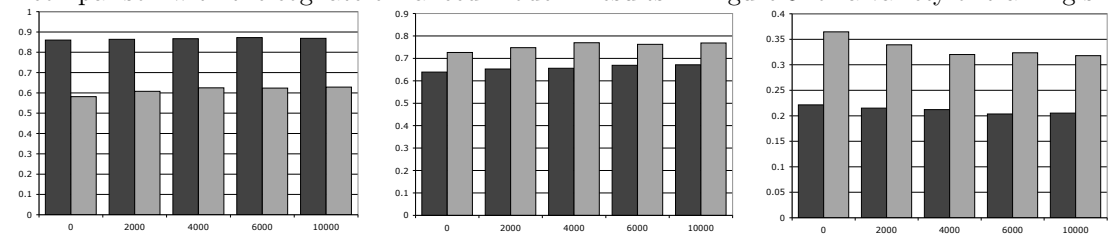


Figure 6. Precision, Recall and AER as a function of training size for Model 1 with cognate matching (light gray) and phrase-based alignment model (dark gray).

The precision of the phrase-based alignments clearly accounts for their lower error rate, as recall is somewhat lower for the phrase-based model than for the word-based model. The phrase-based alignments also seem to improve more slowly with added training data than does the pure word-based model. This makes sense, since the only phrase alignments relevant to performance metrics are phrases that occur in the test set, so added phrase alignments improve the scores only incrementally. In addition, we are discarding potentially good hypotheses extremely aggressively (upto 75%) even before applying the cross consistency metric. As a result, some phrases hypotheses that may have been frequent, but mapped phrases that seemed lexically improbable would have been discarded very early in the aggressive pruning phase. Hence they may not have had a chance to reinforce themselves. As a result, these less probable hypotheses are not in our final set and thus the AER does not change significantly with training data size.

Nonetheless, the phrase-based model significantly improves performance over the word-based model for all quantities of training data.[6] In the sentence pair *elle pourrait constituer une sérieuse*

[6]Unfortunately, due to limited computational resources, we were unable to explore the phrase-based model for

ménace pour la confédération... and *it could become a serious threat to Confederation...* , for example, the phrase-based model correctly identifies subphrases *elle pourrait constituer une / it could become a* and *ménace pour la / serious threat to*, allowing it to correctly align multiple-word sequences of a sentence in which *constituer* had previously been misaligned to the relatively rare word "addition." Phrase alignment, then, helps reduce "garbage collection" as well as improving overall alignment scores and providing a useful translation dictionary.

5 Conclusion

Phrase-based modeling thus significantly improves alignment error rate over a purely word-based model, correctly identifying many phrases that align in the source and target languages. Moreover, accurate phrase identification can dramatically improve translation quality as well as word alignments (cf. Koehn). Identifying phrases can also help address one serious drawback of Model 2: its tendency to over-diagonalize, failing to correctly align sentences in which sentence-initial prepositional phrases or modifying clauses are moved to the end of the sentence. A good phrase-alignment algorithm could identify these chunks and allow Model 2 to align within chunks rather than within the sentence as a whole, significantly improving Model 2's performance. In the meantime, however, phrase alignment allows us to identify multiple-word sequences and French and English that correspond to one another definitively, even when their component words may not align directly. This significantly improves performance in word alignments, and provides a good basis upon which to build an improved machine translation system.

We thus find that adding phrase alignments and introducing dependence on the characteristics of words themselves—rather than merely their distributions—can significantly improve word alignments from French to English, especially when little training data is available. Cognate-matching techniques are most beneficial not when they provide short lists of 100% accurate cognates, but rather when they generate reasonably extensive lists of cognates that are about 80% reliable. Appending each cognate pair to the training set approximately five times generates optimal improvement, as it allows the iterative word-alignment models to reinforce valid connections without over-enforcing invalid cognate pairs. Similarly, word-length dependence can significantly improve alignment performance, with fewer words aligning to the rare words that sometimes serve as "garbage collectors" and other nearby words. Phrase alignments, known to dramatically improve translation quality, also improve the quality of word alignments, and the coordination of the three produces improvement of up to 40% improvement in alignment error rate.[7] We thus find this to be a worthwhile field of investigation, with significant improvements based on only a few weeks' work.

Works Cited

Brown, P., Della Pietra,V., Della Pietra, S., and Mercer, R. "The mathematics of statistical machine translation: parameter estimation." Computational Linguistics, v.19 n.2, June 1993

Koehn, P., Och,F.J., and Marcu, D. "Statistical Phrase-Based Translation." In NAACL/HLT 2003, Proceedings of the Human Language Technology and North American Association for Computational Linguistics Conference, Edmonton, Canada, May 27–June 1 2003.

Kondrak, G., Marcu, D. and Knight, K. 2003. "Cognates Can Improve Statistical Translation Models." HLT-NAACL 2003, companion volume, pp. 46-48. Edmonton, AB, Canada.

training sizes greater than 10,000 sentences.

[7]For Model 1 trained on 10,000 sentences, AER = 0.337 without phrase alignments and 0.209 with phrase alignments.

Marcu, Daniel and William Wong. "A phrase-based, joint probability model for statistical machine translation." Proceedings of the ACL-02 conference on Empirical methods in natural language processing, p.133-139, July 06, 2002.

Simard, M., Foster, G., and Isabelle, P. 1992. "Using Cognates to Align Sentences in Bilingual Corpora." In Proceedings of the 4th Conference on Theoretical and Methodological Issues in Machine Translation (TMI), pages 67–82, Montréal, Canada.

Tiedemann, Jörg. "Automatic Construction of Weighted String Similarity Measures." In Proceedings of the Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora, University of Maryland, College Park.

Venugopal, A., Vogel, S., and Waibel, A. "Effective Phrase Translation Extraction from Alignment Models." Proceedings of 41st Annual Meeting of Association of Computational Linguistics, July 2003, pp. 319–326.

Contributions of each member

Andrea wrote the cognate-matching models and length-dependence and performed well over a thousand tests for various parameters. Dinkar implemented the phrase-matching model, and we worked together to tune its parameters and integrate the cognate matching with phrase alignments to obtain optimal results. We apologize for the length of the report, but we had so many parameters to optimize and explored so many different models that we had a bit of explaining to do.