Word Alignment Models in Statistical MT

CSCi-GA 3033-8 Statistical Natural Language Processing

Assignment 4

Jacqueline Gutman

Table of Contents

Baseline Model

The baseline model aligns all words in French to the correspondingly positioned word in English, and if the French sentence is longer than the English sentence, it aligns remaining words to null. This is a purely diagonal alignment, and is not trained on the data in any way.

Model results

AER: 71.2
Precision: 33.5
Recall: 19.8

Heuristic Models

I first trained two heuristic-based models that collected surface statistics on the training data and then deterministically assigned sure alignments based on the English word which attained the maximum values of those surface statistics.

Conditional probability deterministic aligner

This model was trained by traversing the training data to collect collocation counts for the number of times a given word pair (*f,e*) appeared together in a sentence, as well as counts for the number of times a given English word *e* appeared unconditionally. French words were aligned to the English word that maximized the following ratio of counts from training:

$$alignment(f_i) = \arg\max_{e_j \in E} \frac{c(f,e)}{c(e)c(f)} = \arg\max_{e \in E} \frac{c(f,e)}{c(e)}$$

Model results

1 million training sentences
AER: 39.4
Precision: 57.8
Recall: 66.6

Dice coefficient deterministic aligner

The second heuristic model was based on maximizing the Dice coefficient rather than conditional frequencies. The Dice coefficient was defined as follows:

$$alignment(f_i) = \arg\max_{e_j \in E} \frac{2|f \cap e|}{|f| + |e|}$$

where the cardinality of a word or a word pair is the number of sentences in which the word appeared (or the number of sentences in which both words of the word pair appeared together), regardless of how many times the word appeared within the given sentence. This model outperformed the conditional probability heuristic model.

Model results

1 million training sentences
AER: 36.1
Precision: 57.3
Recall: 78.1

Summary of results

AER	*Baseline model*	*Conditional probability heuristic*	*Dice coefficient heuristic*
100 thousand sentences	71.22	50.52	38.24
500 thousand sentences	71.22	41.45	36.45
1 million sentences	71.22	39.38	36.07

IBM Model 1

The implementation of IBM Model 1 had a nearly uniform distribution over $q(j \mid i,I,J)$, where *j* is the position of the English word aligned to the French word at position *i,* given an English sentence of length *J* and a French sentence of length *I*. The distribution was not completely uniform, however, because the model takes a hyperparameter p_0 such that $q(NULL \mid i,I,J) = p_0$ for all (i,I,J), and English word positions $a_i = j$ are uniformly distributed over the remaining $1 - p_0$. Ideally, we would learn the optimal value for p_0 from the training data, but here I've just treated this as a fixed hyperparameter chosen from a partial grid search on the validation data. Some numerical thresholding was also implemented here (on the suggestions of several classmates) so that values of $t(f \mid e) < 1e-8$ were rounded up to the minimum threshold value when estimating expected counts in the E-step of the EM algorithm.

Hyperparameter values

For this implementation, I considered the EM algorithm to have attained convergence after 20 iterations. Further iterations tended to reduce alignment error rate by a few tenths of a percentage point at most and often showed no performance gains at all. A convergence criterion, beyond the maximum number of iterations, was not used. A constant probability of null alignment $p_0 = .15$ was used. It should be noted that many of these hyperparameters were chosen by considering performance on training data that was orders of magnitude smaller than the full training data used to train and compare model performance, and therefore these values may not actually be optimized for the full training data. The chosen null alignment appeared to perform fairly consistently across various sizes of the training data (above a certain threshold amount of data) and across both IBM models 1 and 2. All subsequent IBM models 1 and 2 discussed were run with 20 iterations and $p_0 = .15$.

Model results

100 thousand training sentences
AER: 29.9
Precision: 68.1
Recall: 73.4

250 thousand training sentences
AER: 27.2
Precision: 72.0
Recall: 74.0

Intersection of model in 2 directions

To improve performance, I ran the same model from French to English and from English to French, and took the intersection of the alignments output by each model. If each model contained a sure alignment between a particular French and English position pair, this was marked as a sure alignment in the intersection model. (If both models contained an alignment between a particular French and English position pair, but one or both of these alignments was a possible alignment, this was added to the intersection model as a possible alignment only. However, the IBM models implemented in this assignment produce only sure alignments, so the intersection model will consist only of sure alignments). Taking the intersection of the models in both directions improved performance significantly.

Model results

100 thousand training sentences
AER: 20.3
Precision: 93.0
Recall: 66.9
500 thousand training sentences
AER: 19.1
Precision: 93.0
Recall: 68.6

Analysis

We can see that the alignment error rate has remarkably dropped by nearly 10 percentage points, before even implementing IBM model 2. Notably, the precision has increased from .681 to .930, while the recall has actually dropped slightly, from .734 to .669. This is because our errors are now more likely to be false negatives (a true alignment predicted by one but not both models, or missed by both models) than false positives (a non-alignment that is mistakenly predicted by both models), when compared to the uni-directional model.

Union of model in 2 directions

As an experiment, I took the union of the model from French to English and from English to French. If either model contained a sure alignment between a particular French and English position pair, this was marked as a sure alignment in the union model. (If either model contained a possible alignment between a particular French and English pair, this was added to the union model as a possible alignment only. Again, however, the IBM models implemented in this assignment produce only sure alignments, so the union model will likewise consist only of sure alignments.) Taking the union of the models weakened performance significantly compared to the uni-directional model.

Model results

100 thousand training sentences
AER: 34.7
Precision: 58.4
Recall: 81.4

Analysis

It is worth noting that while AER rose significantly in the union model, the union model had higher recall than both the original model and the intersection model. This is worth considering if applications of the alignment model have use for a model that optimizes recall rather than precision or AER. It also suggests that we might consider a hybrid alignment model: add the intersection of sure alignments as sure alignments, and add the union of non-intersected alignments as possible alignments. This could potentially be a helpful approach in creating many-to-many alignments. Running this model with the current implementations of the IBM models used here resulted in no difference in performance compared to the intersection model.

IBM Model 2

It should be noted that the original formulation of IBM model 2 attempts to learn a distribute over all possible values of (i,I,J), and as a result of this overparameterization, the data tends to be quite sparse and provides much less efficient estimates. A better approach might be to constrain the probability distributions to fit a particular function or known probability distribution, and then estimate the parameters of that distribution. The downside to this approach is that the MLEs for these new probability distributions often do not have closed, analytic forms, and working around this requires either some clever thinking (cf. Dyer, Chahuneau, & Smith, 2013) or a gradient descent algorithm. To simplify the problem, I have avoided estimating these parameters directly from the data and simply chosen a constant function of a fixed hyperparameter chosen through partial grid search over the validation data. (I began to implement more general distributions that could be learned from the data, by estimating counts within discretized intervals of distortion distance, but did not finish testing these.)

Hyperparameter values

In IBM model 2, we use the same values for $p_0 = .15$ and iterations = 20 used in IBM model 1. As mentioned above, to simplify the problem of estimating a non-uniform probability distribution over the remaining possible non-null alignments, I used a constant function of a fixed hyperparameter α. We define a distortion function and a normalization constant:

$$h(j \mid i,I,J) = \exp\left(-\alpha\left|\frac{j}{J} - \frac{i}{I}\right|\right) \qquad Z(i,I,J) = \sum_{j=1}^{j=J} h(j \mid i,I,J)$$

The following distortion function performed similarly, but somewhat worse, compared to the preferred distortion function described above:

$$h(j \mid i,I,J) = \exp\left(-\alpha\left(j - i\frac{J}{I}\right)\right)$$

We can then define the alignment probabilities as:

$$q(j \mid i,I,J) = \begin{cases} p_0 & j = NULL \\ \dfrac{1-p_0}{Z(i,I,J)} h(j \mid i,I,J) & 1 \le j \le J \end{cases}$$

The translation $t(f|e)$ probability distributions are estimated in the same manner as IBM model, and with the same numerical thresholding. For all subsequent models, $\alpha = 1.4$, as this hyperparameter value appeared to perform quite well across a range of experiments.

Model results

100 thousand training sentences
AER: 23.4
Precision: 75.1
Recall: 79.0

Initializing with IBM Model 1 estimates

IBM model 1 is guaranteed to converge (eventually) on a global optimum, but IBM model 2, in its original formulation, can get caught on local optima, and in this implementation, the model does not necessarily iterate to convergence, but rather, it terminates after a fixed number of iterations. To improve performance, I experimented with initializing IBM model 2 with the IBM model 1 estimates for $t(f|e)$ rather than initializing these translation probabilities as uniform. IBM model 1 was run for only 10 iterations instead of the full 20. This typically improved performance only slightly (and sometimes appeared to hurt performance slightly) compared to the uniform initialization, and would probably be more critical if we were truly estimating $q(j|i,I,J)$ from the data rather than as a function of constant $\alpha = 1.4$.

Model results

250 thousand training sentences
AER: 21.8
Precision: 76.9
Recall: 80.5

Intersection of model in 2 directions

Using the intersection of the alignment model from French to English and English to French, with both models initialized from IBM model 1, and set with $\alpha = 1.4, p_0 = .15$ for 20 iterations, I obtained the following results on the validation data.

Model results

100 thousand training sentences
AER: 16.6
Precision: 94.1
Recall: 72.2
This model was also tested on the test data, and yielded a 15.85% AER on the leaderboard (trained on 100 thousand sentences).

Union of model in 2 directions

Similar to the results for IBM model 1, using the union of the bi-directional models significantly weakened AER and hurt precision tremendously, but it did yield the highest recall of any of the candidate models.

Model results

100 thousand training sentences
AER: 28.5
Precision: 64.3
Recall: 88.8

Summary of IBM Models 1 and 2

AER (100,000 sentences)	*IBM Model 1*	*IBM Model 2*
Uni-directional	29.9	23.4
Intersection	20.3	16.6
Union	34.7	28.5

Source code files

My code for this assignment is available on my private Github repository, which has been shared with the instructor.

https://github.com/jgutman/stat-nlp-fall15-private.git

All code can be found in the file:
`hw4/src/WordAlignmentTester.java`

References

- Collins, Michael. (2011). *Statistical Machine Translation: IBM Models 1 and 2.* Lecture notes from Columbia University. http://www.cs.columbia.edu/~mcollins/courses/nlp2011/notes/ibm12.pdf
- Dyer, C., Chahuneau, V., Smith, N.A. (2013). *A Simple, Fast, and Effective Reparameterization of IBM Model 2*. Article in *Proceedings of NAACL-HLT 2013*, pp. 644-648. http://aclweb.org/anthology/N/N13/N13-1073.pdf
- Koehn, Philipp. (2010). *Word-Based Models*. Chapter 4 in *Statistical Machine Translation*, pp. 81-125. lp20.org/clir/materials/koehn-04.pdf