CIS526 Homework 1: Alignment

Joe Trovato

February 4, 2015

Motivation

I choose to experiemnt with the IBM Models of Alignment. I choose this model for a variety of reasons. The simpliest of which were for the conveniece and ease of comprehension of these models. Specifically, the text book suggests this model when introducing alignment and provides detailed descriptions of how it works. The textbooks suggestion and the widespread use of the model as a standard greatly influence my decision to use this model.

The progression of the IBM models was also appealling; I could build on what I have already done to improve performance instead of implementing a completely new model. While new models may lead to better performace I choose to pursue the IBM Model progression for understandability and implementability. I also tweaked the algorithm slightly to make improvements over the baseline for models 1 and 2.

Algorithm

My algorthim uses the IBM Model 1 to train a lexical translation model. I then use this probabilistic model as a primer for IBM Model 2, which produces a probabistic model that includes an alignment model in addition to the lexical translation model. Finally, I calculated the models for translating in both directions and took the intersection of the two models to eliminate one-to-many alignments in the final alignments.

IBM Model 1

IBM Model 1 generatively models the lexical translation probability distribution between words in different languages. We want to estimate $p(a|e, f)$, and to do this we will count the probabilities of word e being aligned with word f in the same sentence. This probability must also be normalized by total probability counts in the corpus.

The alignment probability model was learned using the Expectation Maximization (EM) algorithm. This begins with the expectation step in which word alinments are assigned based on the current model, then in the maximaztion step, the model is readjsuted based on the current assignments. This algorithm is proven to minimize the log-likelihood of our cost function or in our case to alway reduce the perplexity of the alignments. My code for this model resemsble the psuedocode on page 91 of the textbook. The theory is based on the relavant equations below:

The model we want:

$$p(a|e, f) = \frac{p(e, a|f)}{p(e|f)}$$

The translation model:

$$p(e|f) = \frac{\epsilon}{(l_f + 1)^{l_e}} \prod_{j=1}^{l_e} t(e_j|f_{a(j)})$$

Probability of alignment given english and foriegn sentences:

$$p(a|e,f) = \prod_{j+1}^{l_e} \frac{t(e_j|f_{a(j)})}{\sum_{i=0}^{l_f} t(e_j|f_i)}$$

Translation probability distribution:

$$t(e|f) = \frac{\sum_{e,f} c(e|f)}{\sum_e \sum_{e,f} c(e|f)}$$

The counts:

$$c(e|f) = \sum_a p(a|e,f) \sum_{j=1}^{l_e} \delta(e,e_j)\delta(f,f_{a(j)})$$

IBM Model 2

IBM Model 2 introduces a position alignment model to the original model. Model 1 only takes into account which words are in sentence pairs and ignores there position in the actual sentence. In the new model we introduce the alignemnt probability distribution $a(i|j,l_e,l_f)$ where i and j and the word position and l_e and l_f are the length of the english and foriegn sentences. We introduce the alignment probabilities to give better estimates of $p(e|f)$, $p(e,a|f)$, and $c(e|f)$ to eventually provide a better estimate of $p(a|e,f)$. My code for this model resemsble the psuedocode on page 99 of the textbook. The theory is based on the relavant equations below:

The model we want:

$$p(a|e,f) = \frac{p(e,a|f)}{p(e|f)}$$

Probability of english sentence aligned at a given foreigne sentence:

$$p(e,a|f) = \epsilon \prod_{j=1}^{l_e} t(e_j|f_{a(j)})a(a(j)|j,l_e,l_f)$$

The translation probabilities:

$$p(e|f) = \epsilon \prod_{j=1}^{l_e} \sum_{i=0}^{l_f} t(e_j|f_{a(j)})a(a(j)|j,l_e,l_f)$$

Translation probability distribution:

$$t(e|f) = \frac{\sum_{e,f} c(e|f)}{\sum_e \sum_{e,f} c(e|f)}$$

Counts for translation:

$$c(e|f) = \sum_{j=1}^{l_e} \sum_{i=0}^{l_f} \frac{t(e|f)a(a(j)|j,l_e,l_f)\delta(e,e_j)\delta(f,f_i)}{\sum_{i'=0}^{l_f} a(i'|j,l_e,l_f)}$$

Counts for alignments:

$$c(i|j,l_e,l_f) = \frac{t(e_j|f_i)a(a(j)|j,l_e,l_f)}{\sum_{i'=0}^{l_f} a(i'|j,l_e,l_f)}$$

Merging one-to-many and many-to-one alignments

After completing the IBM Models, one of the main problems was the one-to-many alignments in the probabistic model. To solve this problem I calculated the lexical models translating from french to english and again from english to french, $p(e, a|f)$ and $p(f, a|e)$, respectively. Becuase a lexical translation model can translate one source word to many target words the model can learn many translations for the same word. Checking that both the english to frech model and the frech to english model agree on the alignment eliminates these one-to-many translations. After taking the intersection of the translation models, the model is a one-to-one model and results in more accurate translations.

Results

Table 1 shows my results as my algorithm evolved. One of the largest issues with seeing good results was the inability to run my algorithm on the entire training corpus. I found that my machine as well as the biglab servers could not run the algorithm on the full corpus due to long runtimes and the occasional MemoryError. The number of training sentences used for increasingly complex algorithm decreases to account for this difficulty. To solve this issue, I attempted to make my code as memory and time efficient as possible. This involved removing many redundant loops and variable resets from the psuedocode algorithm provide in the textbook, Statistical Machine Translation by Philipp Koehn. I believe that if I was able to run my algorithm on the entire corpus over more iterations, I would have seen darastically better results. My best results (AER = 24.5) were achieved by training on only 20% of the trainign sentence pairs.

Table 1: AER results with different models and number of training sentences

Model	Training Sentence Pairs	Approximate Alignment Error Rate
IBM Model 1	1000	45
IBM Model 1	100000	35
IBM Model 2	1000	42
IBM Model 2	50000	33
IBM Model 2 and one-to-one heuristic	1000	29
IBM Model 2 and one-to-one heuristic	20000	24.5