Programming Assignment 5 Report
Vignesh Nanda Kumar A59010704

The report goes over the task of machine translation and finding alignments in a parallel corpus containing English and Spanish sentences.

Section 1: IBM Model 1

(a) Description of IBM Model 1

The IBM models are used for the task of machine translation and learn alignments from a parallel corpus. IBM models are an instance of a noisy-channel model. Given the source language f and destination language e, the noisy channel model is used for machine translation and has two components: the language model (p(e)) and the translation model (p(f | e)). The language model can be estimated from the monolingual corpus but the translation model requires the parallel corpus of f and e. Using the language and translation model, the conditional probability of any destination sentence given the source language sentence can be estimated using the formulation:

$$p(e \mid f) = \frac{p(e,f)}{p(f)} = \frac{p(e)p(f \mid e)}{\sum_e p(e)p(f \mid e)}$$

which can be used for finding the translation in destination language. The IBM models are used for modelling the p(f|e) part of the equation using the parallel corpus and alignments between source and destination sentence. IBM model 1 considers alignments at each source sentence position to be equally likely whereas IBM model 2 introduces alignment probabilities used for modelling the alignment part of the model.

The limitations of IBM Model are that the model does not firstly take order into consideration. The translation model considers one word at a time and it translates each word in isolation. Also, the alignments considered in model 1 are chosen uniformly randomly which is not a correct assumption because it says that a foreign word can be equally likely to be aligned to any destination sentence word. This can lead to getting noisy/erroneous alignments. Also, since words are translated in isolation, one foreign word cannot be aligned to multiple words on the destination language side.

(b) Description of EM Algorithm

The IBM models uses alignments and translation probability to estimate the probability distribution p (f | e). To estimate the probability distribution we need to first compute the alignments in the parallel corpus training data which is computed using the EM algorithm. The parameter estimation algorithm takes as input the parallel corpus and outputs the parameters: translation probability for source destination word pair in case of IBM model 1 and also the alignment probability in case of IBM model 2. EM algorithm contains two steps. The first step is the expectation step to find out the soft alignment between the words of the parallel corpus given the parameter estimates for translation and alignment. The second step is the maximization step, where given the soft alignment between words of the corpus, the parameters of the model (translation and alignment probabilities) are re-estimated and these new estimates will be used in the next expectation step. This expectation and maximization step can be run for multiple iterations until we get the desired performance or until convergence.

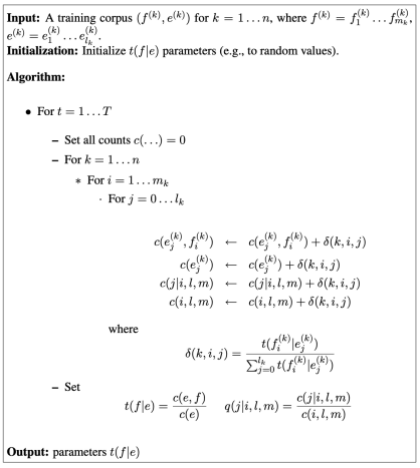
One of strengths of EM algorithm is that from an implementation perspective, the algorithm is pretty intuitive in the sense that it clearly differentiates the expectation and maximization step for estimating the parameters. Also in the case of IBM model 1 where the gold alignments are not provided the EM algorithm is able to find out a local maxima for estimating the translation parameter. Because of the iterative nature of the EM algorithm, it is able to refine the transition parameters across iterations using the maximum likelihood estimate at each step.

Coming to the weaknesses of the algorithm, the EM algorithm converges to a local maximum of the log-likelihood function. So, it is not necessary that the EM algorithm always converges to the global maxima. In terms of machine translation problem, this means that we may not necessarily get the best alignments, translations parameter estimates always using the EM algorithm. Also, the result of the EM algorithm heavily depends on how the parameters are initialized before the first iteration and based on that initialization the final parameters will be decided (after some iterations). So, considerable effort has to be put in to properly initialize the parameters and try with different parameter settings to ensure that the parameters converge to the global maxima. One more shortcoming of the algorithm is that for every iteration, the algorithm has to go over each word of all the sentences of the parallel corpus ($O(n^2)$ time complexity). This can make the algorithm slow to provide parameter estimates in cases when the corpus size is large.

(c) Method Overview

I took a class and inheritance based approach to implement the IBM models. For IBM model 1, these are the steps I give an overview of below:

1. I initialize the counts variables required in the EM algorithm as dictionaries. Then I read the corpus of English and the foreign language converted to a parallel corpus. After this, I initialize n(e) attribute which gives how many different words can a particular English word align to for all the English words in the parallel corpus. This is used in initializing the translation probabilities (t(f|e) = 1/n(e)).
2. After initializing the parameters and all the count dictionaries, I run the EM algorithm for 5 iterations to estimate the value of the translation probability which will be then used for predicting alignments in the validation dataset.



Input: A training corpus $(f^{(k)}, e^{(k)})$ for $k = 1 \ldots n$, where $f^{(k)} = f_1^{(k)} \ldots f_{m_k}^{(k)}$, $e^{(k)} = e_1^{(k)} \ldots e_{l_k}^{(k)}$.

Initialization: Initialize $t(f|e)$ parameters (e.g., to random values).

Algorithm:

- For $t = 1 \ldots T$
  - Set all counts $c(\ldots) = 0$
  - For $k = 1 \ldots n$
    - For $i = 1 \ldots m_k$
      - For $j = 0 \ldots l_k$

$$
\begin{aligned}
c(e_j^{(k)}, f_i^{(k)}) &\leftarrow c(e_j^{(k)}, f_i^{(k)}) + \delta(k,i,j) \\
c(e_j^{(k)}) &\leftarrow c(e_j^{(k)}) + \delta(k,i,j) \\
c(j|i,l,m) &\leftarrow c(j|i,l,m) + \delta(k,i,j) \\
c(i,l,m) &\leftarrow c(i,l,m) + \delta(k,i,j)
\end{aligned}
$$

where

$$\delta(k,i,j) = \frac{t(f_i^{(k)}|e_j^{(k)})}{\sum_{j=0}^{l_k} t(f_i^{(k)}|e_j^{(k)})}$$

  - Set

$$t(f|e) = \frac{c(e,f)}{c(e)} \qquad q(j|i,l,m) = \frac{c(j|i,l,m)}{c(i,l,m)}$$

Output: parameters $t(f|e)$

Figure 1: IBM Model 1 EM algorithm pseudo code

3. The pseudo code for the EM algorithm for IBM model 1 is given in the figure above. After initializing the translation probabilities, I run the EM algorithm for n iterations. In each iteration, I iterate over all the sentences of the parallel corpus. For each sentence, I go over all the words of the foreign sentence and the English sentence to estimate the counts which will be used computing the translation probabilities. The counts are updated using the delta parameter which is computed using the translation probability of a particular English, foreign word pair and is normalized such that the summation over all sentences for a particular alignment pair is 1. After finding out the counts, I use them to re-estimate the translation probabilities.
4. For finding the alignments, I load the validation data and for each sentence in the parallel corpus, I compute the alignment. For each word in the foreign sentence, I check which word in the English sentence gives the maximum translation probability for the word pair.

(d) Results

On running the model on the training corpus for 5 iterations, the following results were obtained on the dev set. The alignments for each word were compared with the gold alignment given and the precision, recall and F1-score were computed.

Precision	Recall	F1-score
0.413	0.427	0.420

Table 1: Results of IBM Model 1 on dev dataset

Some of the challenges faced during implementation were to find the alignment during validation. To find the validation, the max value of transition probability had to be chosen so instead of >, I was using >= condition due to which the F1-score was slightly off. Another challenge was to optimize the time taken to perform the EM algorithm. Some optimizations I had to do here were to pre-compute the denominator of the delta value used in computing counts. Also, one more reason why the F1-score was coming off initially was that the counts were not being reset to 0 in every iteration.

(e) Discussion

Iteration	Precision	Recall	F1-score
1	0.211	0.217	0.214
2	0.375	0.387	0.380
3	0.402	0.415	0.408
4	0.410	0.423	0.416
5	0.413	0.427	0.420

Table 2: F1 score across iterations of EM algorithm

As we can observe from the above table, we can see that the with increasing iterations, the F1-score is improving. This is expected because with increasing iterations, the translation probabilities get re-estimated and gets more closer to the local maxima. Hence, we can expect better alignments due to better translation probabilities as the number of iterations increase before we get convergence.

Section 2: IBM Model 2

(a) Description of IBM Model 2

IBM model 1 considers all alignments to be equally likely which is not a correct assumption to make. IBM model 2 improves upon this by assigning probabilities to alignments (introduces distortion parameters q). This parameter gives the probability that a particular foreign word is aligned to an English word given the sentence lengths. This tweaks the way the way p (f, a |

e,m) is defined and is shown in the below equation. The probability now has the alignment probability also included. Because of introducing this alignment factor, now non-uniform probabilities will be assigned to words in the English sentence for each word in the foreign sentence. This ensures better alignments for English and the foreign sentence.
The limitations of the IBM model 2 is that in this model too just aligning words many to one is not a valid assumption. One foreign word cannot be aligned to multiple words on the English side according to the way the model is functioning. This can lead to wrong translations and can miss a lot of translation cases. Also, since the words are looked in isolation, the translation will not take the context of the word into account. To avoid this, phrase-based translation can be used where we look at chunks of words together instead of words in isolation.

(b) Method Overview

IBM model 2 is very similar in implementation to IBM model 1 except a few changes to accommodate the alignment probability. These are the changes made:

1. IBM Model 2 inherits the IBM Model 1 class and we initialize the translation probabilities from the saved translation probabilities of IBM model 1(after 5 iterations of EM algorithm). Then the soft alignment probabilities (q) are initialized to 1 / (length of English sentence + 1).

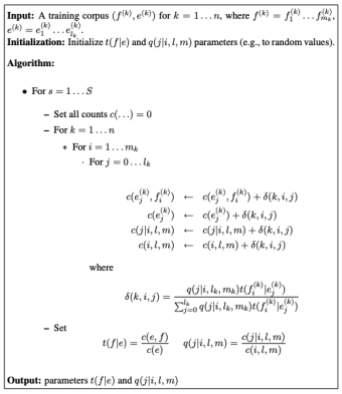


Figure 2: IBM Model 2 EM algorithm pseudo code

2. Now coming to the EM algorithm, as shown in the figure above there are 2 main differences compared to IBM Model 1. The first one is the way delta is computed, now the alignment probability is also included in the computation of delta along with the translation probability. Also, in the maximization step, after computing the counts, to compute the parameters, the soft alignment probability (q) is also re-estimated in every iteration. The other implementation details are same as that in IBM model 1.
3. For finding out the best alignments on the validation dataset, we see the max value of the product of q (alignment probability) and t (translation probability) instead of just the translation probability used in IBM model 1.

(c) Results

On running the model on the training corpus for 5 iterations, the following results were obtained on the dev set. The alignments for each word were compared with the gold alignment given and the precision, recall and F1-score were computed.

Precision	Recall	F1-score
0.442	0.456	0.449

Table 3: Results of IBM Model 2 on dev dataset

Some of the challenges faced during implementation were to find the alignment during validation. To find the validation, the max value of transition probability had to be chosen so instead of >, I was using >= condition due to which the F1-score was slightly off. Another mistake I had made was that while finding the best alignment during validation, I was just storing the max translation probability value which was giving lower F1-score but instead of that I had to store the product of alignment and translation probability.

(d) Discussion

Iteration	Precision	Recall	F1-score
1	0.432	0.446	0.439
2	0.435	0.449	0.442
3	0.439	0.453	0.446
4	0.439	0.453	0.446
5	0.442	0.456	0.449

Table 4: F1 score across iterations of EM algorithm

As we can observe from the above table, we can see that the with increasing iterations, the F1-score is improving. This is expected because with increasing iterations, the translation probabilities get re-estimated and gets more closer to the local maxima. This is similar to the behavior observed in IBM model 1. The F1 scores here are much better than those obtained in

IBM model 1. This is also expected behavior because for model 2 we are not considering all alignments to be equally likely. Here we are assigning non-uniform alignment probabilities which is being computed by the EM algorithm.

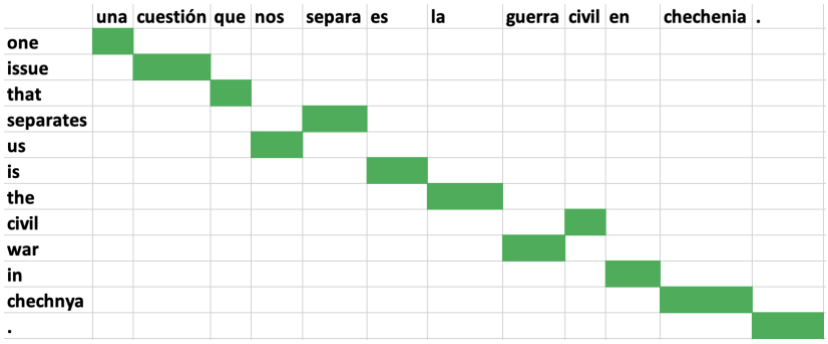
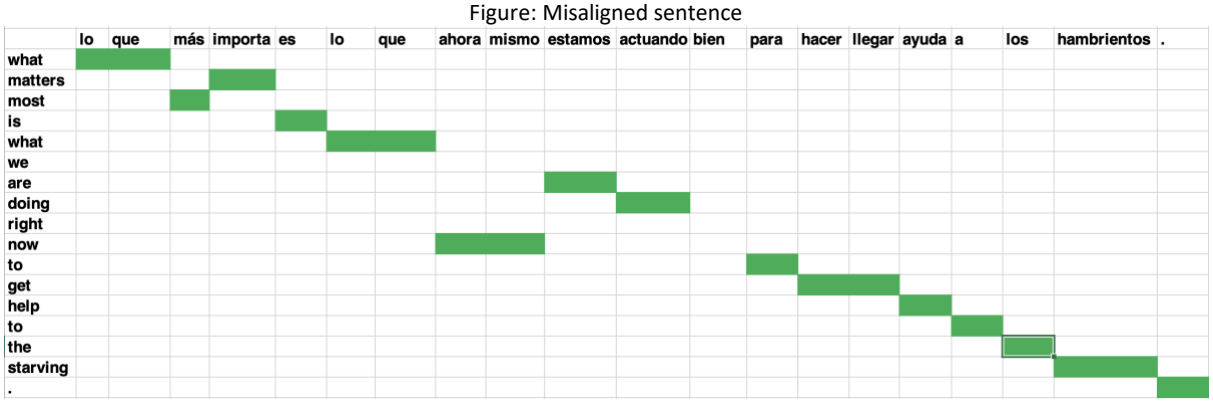


Figure: Correctly aligned sentence

This is an example of a correctly aligned sentence where the given gold alignment matches with the prediction. We can see here that the English and Spanish words are aligning one to one and the IBM model 2 is able to capture this type alignment. IBM model 2 is also able to capture the alignments which are off by one position (for example nos separa for separates us and Guerra civil for civil war). This sentence is able to work well because there are no such phrases in the Spanish sentence which have to map to a English word and also each Spanish word is mapping just to a single English word and not multiple English words so the assumption of the IBM model 2 is also valid for this sentence.

Figure: Misaligned sentence



The above figure gives an example of a sentence where there are a lot of misalignments in the predicted alignments compared to the gold alignments given. We can see that in the gold alignment there are a lot of phrases in the foreign sentence that are getting mapped to a single English word (lo que mapped to what, ahora mismo mapped to now, hacer llegar mapped to get). This is not being done in the prediction. In the prediction we see that a lot of Spanish words are getting mapped to starving. This is a shortcoming of the IBM model where it looks only at words in its isolation and not at phrases, it is not able to correctly capture these models. Also, the sentence being a long sentence, there are some long range dependencies present in the sentence (like ahora and mismo being mapped to now) which is not being done correctly by the predicted alignment. To overcome these shortcomings phrase based models or neural seq2seq models can be used which will be able to capture these dependencies better and also able ot do phrase based translation.

References:

1. http://www.cs.columbia.edu/~mcollins/courses/nlp2011/notes/ibm12.pdf
2. Lecture slides
3. https://www.youtube.com/watch?v=XvM0KP3ZgAE&list=PLlQBy7xY8mbIxoh44jbC22Kc8zBj0bGbQ