CSE 256 Assignment 4

Chowdhury, Ranak Roy, A53317421

November 26, 2019

1 IBM Model 1

1.1 Description of IBM Model 1

IBM Models are used for Statistical Machine Translation (SMT) systems. Model 1 is a probabilistic generative model within a framework that assumes a source sentence f of length m translates as a target sentence e, according to the following stochastic process:

- A length l for sentence e is generated.
- For each target sentence position $j \epsilon 1, 2, ...l$:
  - A generating word f_i in f is selected and
  - The target word e_j at position j is generated depending on f_i

Model 1 is defined as a particularly simple instance of this framework, by assuming all possible lengths for l (less than some arbitrary upper bound)have a uniform probability, all possible choices of source sentence generating words are equally likely, and the translation probability $t(f|e)$ of the generated target language word depends only on the generating source language word. An alignment a identifies which target word each source word generated from. Therefore, for IBM Model 1:

$$p(f, a|e, m) = p(a|e, m)p(f|a, e, m) = \frac{1}{(l+1)^m} \prod_{j=1}^{m} t(f_j|e_{a_j}) \tag{1}$$

Limitations:

- There are $(l+1)^m$ possible alignments per foreign sentence. It grows exponentially with the length, m of the foreign sentence. This is a huge sample space to search from to translate a single sentence.
- The position of any word in the target sentence is independent of the position of the corresponding word in the source sentence, or the positions of any other source language words or their translations. The tendency for a contiguous phrase in one language to be translated as a contiguous phrase in another language is not modeled at all.
- Whether a particular source word is selected to generate the target word for a given position is independent of which or how many other target words the same source word is selected to generate.
- It aligns one word at a time and does not consider the neighboring words. Therefore, it fails to consider the context in which a word appears.
- Unlike IBM Model 2, it does not have any alignment parameters.
- It is weak in terms of conducting reordering or adding and dropping words. In most cases, words that follow each other in one language would have a different order after translation, but IBM Model 1 treats all kinds of reordering as equally possible.
- Each word in the target sentence can be generated by at most one word in the source sentence. Situations in which a phrase in the source sentence translates as a single word in the target sentence are not well-modeled.

1.2 Description of EM Algorithm

Expectation-Maximization is a general purpose learning algorithm for probabilistic models. It is a parameter estimation method that falls into the general framework of Maximum Likelihood Estimation (MLE). EM is an iterative process that works by repeating two steps:

- **E Step**: Based on the current model parameters, calculate the expectations of counts regarding latent variables.
- **M Step**: Based on these counts, update the model parameters.

The parameters of IBM model for a given pair of languages are normally estimated using EM, taking as training data a corpus of paired sentences of the two languages, such that each pair consists of sentence in one language and a possible translation in the other language.

Strengths of EM:

- Have probabilistic interpretation.
- Likelihood is guaranteed to increase for each iteration.
- The algorithm is derivative-free, i.e. it does not require an optimizer.
- Fast if analytical expressions for the M-step are available.
- Parameter constraints are often dealt with implicitly.

Weaknesses of EM:

- Initialization affects the final result.
- Overfitting issues.
- Convergence may be slow if analytical expressions for the M-step are not available since numerical optimization must be applied.
- The EM algorithm will converge to a local optimum of the log-likelihood function, not a global optimum.

1.3 Method Overview

Data Structure: Initially I stored t parameters for possible pairs of foreign and English words, i.e. words that occur together in some parallel translation, and the special English word NULL. t was implemented as a collection of dictionaries. The outer dictionary contained each english word as a key and a corresponding dictionary as a value. The dictionary corresponding to each english word contains the foreign word as the key and the corresponding $t(f|e)$ as its value.

Steps of EM:

- Initialization: Set $t(f|e)$ to be the uniform distribution over all foreign words that could be aligned to e in the corpus. More specifically, $t(f|e) = \frac{1}{n(e)}$, where $n(e)$ is the number of different words that occur in any translation of a sentence containing e. Note that the special English word NULL can be aligned to any foreign word in the corpus.
- Expectation: For the k-th sentence pair and each index i in the foreign sentence f^k and j in the English sentence e^k, we define

$$\delta(k,i,j) = \frac{t(f_i^k|e_j^k)}{\sum_{j=0}^{l_k} t(f_i^k|e_j^k)} \tag{2}$$

  where l_k is the length of the English sentence.

$$c(e_j^k, f_i^k) \leftarrow c(e_j^k, f_i^k) + \delta(k,i,j) \tag{3}$$

$$c(e_j^k) \leftarrow c(e_j^K) + \delta(k,i,j) \tag{4}$$

- Maximization: After each iteration through the parallel corpus, we revise our estimate for t parameters:

$$t(f|e) = \frac{c(e,f)}{c(e)} \tag{5}$$

  for all possible foreign words f and English words e (and NULL).

- Iterate 5 times and recover the final values for the parameters $t(f|e)$ to be used for initialized in IBM Model 2 implementation.

Evaluation: The trained model was then used to find alignments for the development sentence pairs dev.en and dev.es. For each sentence, each foreign word f_i was aligned to the English word with the highest $t(f|e)$ score,

$$a_i = argmax_{j\epsilon 0...l} t(f_i|e_j) \tag{6}$$

1.4 Results

Table 1: Precision, Recall and F1-score of IBM Model 1 for 5 iterations

Total	**Precision**	**Recall**	**F1-Score**
5920	0.402	0.416	0.409

As Table 1 shows, the F1-score for my model is 0.409 but the actual answer should be 0.42. A key challenge was to figure out how to map the NULL words during evaluation. As stated in the problem, the gold alignments do not include NULL word alignments. However when I trained my model, I had to consider NULL word to be the 0-th word of every English statement. Therefore, when I computed $a_i = argmax_{j\epsilon 0...l} t(f_i|e_j)$ for every foreign word f_i, some of them aligned to NULL because j will iterate from 0 to l, and the 0-th word refers to NULL. I tried a few possibilities while computing the best alignment if a foreign word f_i maps to NULL:

- I output the corresponding gold alignment as it appears in the dev.key file.
- I output the NULL word alignment in the dev.out file, considering that NULL is the 0-th word of the english sentence.
- I avoid outputting the alignment in the dev.out file.
- I compute the alignment to the English word with the second highest alignment.

I chose to output the NULL word alignment in the dev.out file, considering that NULL is the 0-th word of the english sentence. Therefore, for some of my alignments, the spanish word gave the highest $t(f|e)$ score for the 0-th word of the English sentence, which is NULL.

1.5 Discussions

Table 2: Variation of F1 score with iteration for IBM Model 1

Iteration	1	2	3	4	5
F1-score	0.214	0.372	0.395	0.405	0.409

As Table 2 shows, the F1-score increases with increasing number of iterations. This is because the EM algorithm has the property to increase the likelihood of the data at each iteration. Moreover, the F1-score increases at a decreasing rate with the number of iterations. This implies that convergence is faster for the first few iterations but then it slows down as it approaches a local maximum.

2 IBM Model 2

2.1 Description of IBM Model 2

IBM Model 1 has a very simplified view of the world,where each word in the sentence is translated independently without any regard for word order. The IBM Model 2 has an additional model for alignment that is not present in Model 1. The IBM Model 2 addressed this issue by modeling the translation of a foreign input word in position i to a native language word in position j using an alignment probability distribution defined as: $q(j|i,l,m)$, where l is the length of the sentence in native language and m is the length of the sentence in foreign language. It is better than IBM Model 1 because it considers absolute word positions over considering uniform distribution for alignment. In this way, it allow us, for example, to capture the tendency for words close to the beginning of the French sentence to be translations of words close to the beginning of the English sentence.

Limitations of IBM Model 2:

- It is vastly overparameterized, making it prone to degenerate behavior on account of overfitting.
- EM algorithm for IBM model 2 may be sensitive to initialization: depending on the initial values, it may converge to different local optima of the log-likelihood function.

2.2 Method Overview

Data Structure: Initially I stored the q parameters for pairs of target sentence lengths l and source sentence length m that occur in the corpus. q was implemented as a dictionary. The key was the string 'j i l m', where j is the set of indices from 0 to l and m is the set of indices from 1 to m. The corresponding value for each key was the parameter $q(j|i,l,m)$.

Steps of EM: The IBM model 2 extends the implementation of the EM algorithm for IBM model 1. Therefore, I have just shown the main additional step, which is adapting the delta function to include $q(j|i,l,m)$.

- Initialization: $t(f|e)$ parameters were initialized by loading their final values produced by IBM Model 1. The q parameters were initialized to the uniform distribution over all j for each i, l and m. $q(j|i,l,m) = \frac{1}{l+1}$.
- Expectation: For the k-th sentence pair and each index i in the foreign sentence f^k and j in the English sentence e^k, we define

$$\delta(k,i,j) = \frac{q(j|i,l_k,m_k)t(f_i^k|e_j^k)}{\sum_{j=0}^{l_k} q(j|i,l_k,m_k)t(f_i^k|e_j^k)} \tag{7}$$

  where l_k is the length of the English sentence.

$$c(j|i,l,m) \leftarrow c(j|i,l,m) + \delta(k,i,j) \tag{8}$$

$$c(i,l,m) \leftarrow c(i,l,m) + \delta(k,i,j) \tag{9}$$

  We also compute $c(e_j^k, f_i^k)$ and $c(e_j^k)$ like in IBM Model 1.
- Maximization: After each iteration through the parallel corpus, we revise our estimate for t and q parameters:

$$t(f|e) = \frac{c(e,f)}{c(e)} \tag{10}$$

$$q(j|i,l,m) = \frac{c(j|i,l,m)}{c(i,l,m)} \tag{11}$$

  for all possible foreign words f and English words e (and NULL).
- Iterate 5 times and recover the final values for the parameters $t(f|e)$ and $q(j|i,l,m)$.

Evaluation: The trained model was then used to find alignments for the development sentence pairs dev.en and dev.es. For each sentence, the best alignment for each foreign word f_i is

$$a_i = argmax_{j\epsilon 0\ldots l} q(j|i,l,m)t(f_i|e_j) \tag{12}$$

2.3 Results

Table 3: Precision, Recall and F1-score of IBM Model 2 for 5 iterations

Total	**Precision**	**Recall**	**F1-Score**
5920	0.435	0.449	0.442

As Table 3 shows, the F1-score for my model is 0.442 but the actual answer should be 0.45. A key challenge was to figure out how to map the NULL words during evaluation. As stated in the problem, the gold alignments do not include NULL word alignments. However when I trained my model, I had to consider NULL word to be the 0-th word of every English statement. Therefore, when I computed $a_i = argmax_{j\epsilon 0\ldots l} t(f_i|e_j)$ for every foreign word f_i, some of them aligned to NULL because j will iterate from 0 to l, and the 0-th word refers to NULL. I tried a few possibilities while computing the best alignment if a foreign word f_i maps to NULL:

- I output the corresponding gold alignment as it appears in the dev.key file.

- I output the NULL word alignment in the dev.out file, considering that NULL is the 0-th word of the english sentence.
- I avoid outputting the alignment in the dev.out file.
- I compute the alignment to the English word with the second highest alignment.

I chose to output the NULL word alignment in the dev.out file, considering that NULL is the 0-th word of the english sentence. Therefore, for some of my alignments, the spanish word gave the highest $t(f|e)$ score for the 0-th word of the English sentence, which is NULL.

2.4 Discussions

2.4.1 Discussion of the results of IBM Model 2

Table 4: Variation of F1 score with iteration for IBM Model 2

Iteration	1	2	3	4	5
F1-score	0.426	0.432	0.437	0.441	0.442

As Table 4 shows, the F1-score increases with increasing number of iterations. This is because the EM algorithm has the property to increase the likelihood of the data at each iteration. Moreover, the F1-score increases at a decreasing rate with the number of iterations. This implies that convergence is faster in the first few iterations but then it slows down as it approaches a local maximum.

2.4.2 Comparison of IBM Model 1 and 2

Table 5: Comparison of F1 score for IBM Model 1 and 2 with varying iteration

Iteration	1	2	3	4	5
IBM Model 1 F1-score	0.214	0.372	0.395	0.405	0.409
IBM Model 2 F1-score	0.426	0.432	0.437	0.441	0.442

As Table 5 shows, the F1-score for IBM Model 2 is always greater than that of IBM Model 1 at each iteration because IBM Model 2 considers the alignment parameter. Moreover, since I am using the t parameters learnt from IBM Model 1 to initialize the t parameters for IBM Model 2, the IBM Model 2 starts with a very high F1 score of 0.426 compared to 0.214 of IBM Model 1. Both the models converge after the first few iterations. Since IBM Model 2 starts from a relatively high score of 0.426, the F1 score does not increase much with iteration. But since IBM Model 1 starts from a very poor value of 0.214, the F1 score increases rapidly with iteration.

2.4.3 Examples of correctly and incorrectly aligned examples

I plotted the alignment for sentence 197 of the validation set. As seen from Figure 1, the spanish words 'deseo' and 'exitos' have been correctly aligned to the english words 'wish' and 'success', respectively. However, the two misaligned examples, namely spanish words 'le' and 'mayor' have also been aligned to the english word success, which is incorrect. The gold alignments for the spanish words 'le' and 'mayor' are 'you' and 'every', respectively. However, the spanish translation of 'every' can be 'cada' and the english translation of 'mayor' can be 'higher'. Similarly, the spanish translation of 'you' can be 'tu'. So we can observe that there are ambiguities in the two spanish words 'le' and 'mayor' because they can have different meanings based on the context of the surrounding words. Since, we do not look at surrounding words, while translating a particular word, there is no way to capture the meaning of a word specific to that context. Therefore, some of the spanish words may be misaligned.

3 Growing Alignments

3.1 Method Overview

Initially, I ran my IBM Model 1 for 5 iterations to save the t parameters when English is the source and Spanish is the target language, $t(e|f)$. Then I used those t parameters to initialize my IBM Model 2. Then I trained my IBM Model

Figure 1: Figure showing two correctly aligned examples (colored green) and two misaligned examples (colored black) for sentence number 197 of the validation set under IBM Model 2

2 for 5 iterations to generate alignments when English is the source and Spanish is the target language. Now, I have two sets of alignments i.e. Spanish to English, $t(f|e)$ and English to Spanish $t(e|f)$. Then I found the intersection and union between the two sets of alignments. Then, in pursuit to generate the best possible alignments, I initially inserted all the intersection alignments into the best alignment set. Then starting from the intersection, I applied a heuristic to grow the alignment.

Heuristic: I only explored alignments in the union of $t(e|f)$ and $t(f|e)$. I added alignment points which align a word that currently has no alignment. I did it for both Spanish and English words. If I found multiple alignments for a single word in the union, I looked for alignment points that are 'neighbors' (adjacent or diagonal) of current alignment points. If I did not find any such alignment, then I randomly chose an alignment point.

3.2 Results and Discussions

Table 6: Precision, Recall and F1-score of growing alignment

Total	**Precision**	**Recall**	**F1-Score**
5920	0.389	0.534	0.450

As Table 6 shows the growing alignment gives a better F1 score of 0.45 than any of the IBM models did after 5 iterations. I also ran this algorithm without considering 'neighboring' (adjacent or diagonal) alignment points for a given word. In this case, if a word had multiple alignments in the union set, I randomly chose one of the alignments. This gave a poorer F1 score of 0.446. This is because, words in close proximity in one language also generally appears in close proximity of another language. Therefore, if we randomly chose alignments instead of considering the neighboring alignment pairs first, we generally get poorer translation.