Report 4: Machine Translation

Last update: June 10, 2019

1 IBM Model 1

1.1 Description of IBM Model 1

IBM model 1 (IBM-1) is one of the IBM translation models, which are a sequence of increasingly complex models used for statistical machine translation. IBM-1 is a generative model that only uses lexical translation. In this paper, the task is to translate from Spanish (the "source" or foreign language) to English (the "target" language). Given a source sentence of length m: $f = f_1, f_2, \cdots, f_m$ and the corresponding target sentence of length l: $e = e_1, e_2, \cdots, e_l$, IBM-1 learns a translation model $p(f, a|e, m)$ which can be defined as following.

$$p(f, a|e, m) = p(a|e, m) * p(f|a, e, m) \tag{1}$$

IBM-1 makes two basic assumptions. Firstly, it assumes that the alignments only depend on the length of target sentence l, i.e.

$$p(a|e, m) = \frac{1}{(l+1)^m} \tag{2}$$

Secondly, it assumes that the word translations is independent with each other. Based on the two assumptions, IBM-1 defines the conditional distribution over source sentence and alignments as

$$p\left(f, a|e, m\right) = \prod_{i=1}^{m} \frac{1}{l+1} t(f_i|e_{a_i}) \tag{3}$$

However, the two assumations makes IBM-1 weak in terms of two aspects. One problem is IBM-1 is weak of conductig reordering. Normally the word order in one language is different after translation, but IBM-1 treats all kinds of reordering as equally possible. Another problem IBM-1 does not consider aligning fertility. In most cases one input word will be translated into one single word, but some words will produce multiple words or even get dropped. IBM-1 only enables the input word to be translated into one single word.

1.2 Description of EM Algorithm

Expectation–maximization (EM) algorithm is a widely-used iterative method to find maximum likelihood estimates of parameters in statistical models. In this paper, EM algorithm is applied to estimating the parameters of IBM models.

An advantage of EM algorithm is that the likelihood is guaranteed to increase for each iteration, however convergence toward the maximum may be slow. Furthermore, EM algorithm is derivative-free, namely it does not require an optimizer, however it requires more effort in the implementation phase.

1.3 Method Overview

Our program firstly estimates the translation parameters of IBM-1 using corpus.en and corpus.es as input. After the initialization step, it runs 5 iterations of the EM algorithm and obtains the translation parameters.

Given the training corpus of size n: (f^k, e^k) for k=1,2,...,n. We firstly initialize $t(f|e)$ to be the uniform distribution over all foreign words that could be aligned to e in the corpus, i.e

$$t(f|e) = \frac{1}{n(e)} \tag{4}$$

where $n(e)$ is the number of different words that occur in any translation of a sentence containing e.

In E-steps, for each pair of source word and target word, IBM-1 updates the two count values $c(e_j^k, f_i^k)$ and $c(e_j^k)$ by adding $\delta(k, i, j)$, which is defined as

$$\delta(k, i, j) = \frac{t(f_i^k|e_j^k)}{\sum_{j=0}^{l_k} t(f_i^k|e_j^k)} \tag{5}$$

In M-steps, IBM-1 update $t(f|e)$ using $c(e, f)$ and $c(e)$, i.e

$$t(f|e) = \frac{c(e, f)}{c(e)} \tag{6}$$

Then our program reads the development sentence pairs dev.en and dev.es and index the total t parameters for the word paris from the development sentence pairs. Therefore, for each source word f_i, we can obtain a list of scores $t(f_i|e_j)$ for $j = 0, 1, ..., l$. Finally, our program aligns each source word f_i to the target word with highest $t(f|e)$ score, i.e.

$$a_i = \arg\max_{j \in 0...l} t(f_i|e_j) \tag{7}$$

1.4 Results and Discussions

Evaluating on 5920 sample, we obtain F1-score of 0.432, Precision of 0.425 and Recall of 0.439.

We runs 5 iterations of the EM algorithm. The F1-score for the 5 iterations are 0.199, 0.386, 0.417, 0.429 and 0.439 respectively. We can find that the F1-score is increasing, however the incremental change of F1-score is decreasing. This means the accuracy of translation parameters increases during the updating process. The traslation parameters will converge to certain values after a few iterations.

2 IBM Model 2

2.1 Description of IBM Model 2

IBM Model 2 (IBM-2) can be regarded as an extension of IBM-1 by adding the alignment probability distribution.

$$p(a|e,m) = \prod_{i=1}^{m} q(a_i|i,l,m) \tag{8}$$

IBM-2 translation model can be decribed as

$$p(f,a|e,m) = \prod_{i=1}^{m} q(a_i|i,l,m)*t(f_i|e_{a_i}) \tag{9}$$

Compared with IBM-1, IBM-2 address the reordering problem by using the alignment probability distribution. Therefore, IBM-2 can output more accurate translations. However, it still does not address the fertility problem. Neither it addresss the problem that the reordering of words is not independent. Moreover, in the implementation phrase, the EM algorithm for IBM-2 may be sensitive to initialization. Depending on the inital values, it may converge to different local optima of the log-likelihood function. Usually, to tackle with this issue, we usually use IBM-1 for parameter initialization.

2.2 Method Overview

The main steps of IBM-2 method is similar to IBM-1 method. Differently, we have to estimate both the translation parameters and the alignment parameters for IBM-2. The alignment parameters $q(j|i,l,m)$ is initialized with $\frac{1}{l+1}$. In the E-steps, besides updating the two count values $c(e_j^k, f_i^k)$ and $c(e_j^k)$, IBM-2 also updates $c(j|i,l,m)$ and $c(i,l,m)$ also by adding $\delta(k,i,j)$, where

$$\delta(k,i,j) = \frac{q(j|i,l_k,m_k)t(f_i^k|e_j^k)}{\sum_{j=0}^{l_k} q(j|i,l_k,m_k)t(f_i^k|e_j^k)} \tag{10}$$

In the M-steps, IBM-2 updates both $t(f|e)$ and $q(j|i,l,m)$. $q(j|i,l,m)$ is updated by using $c(j|i,l,m)$ and $c(i,l,m)$, i.e.

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

Finally, the best alignment for each source word f_i is

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

2.3 Results and Discussions

Evaluating on 5920 sample, the F1-score of IBM-2 method reaches 0.454, Precision reaches 0.447 and Recall reaches 0.461. Compared with IBM-1, F1-score is improve by 2.2%.

We runs 5 iterations of the EM algorithm. The F1-score for the 5 iterations are 0.438, 0.442, 0.449, 0.453 and 0.454. Compared with the result of IBM-1 method, the F1-score increases from 0.438. This is because we apply the trained translation parameters from IBM-1 in IBM-2 model, thus the F1-score of the first interation is higher than the fifth iteration of IBM-1. Besides, we can find than the change of F1-score is also decreasing, indicating the parameters converge.

We select the 106th sample from dev.en/dev.es to show the correctly aligned examples and misaligned examples. The result is as shown in Figure 1. The correctly aligned examples are marked with color green and misaligned examples are marked with red. From the result, we can find that all the punctuations are aligned correctly. Some word are misaligned. For example, "operation" should be aligned with "en" and "marcha". Each Spanish word in the figure are aligned with a single English word, however the gold alignments allow Spanish words to be aligned with multiple English words.

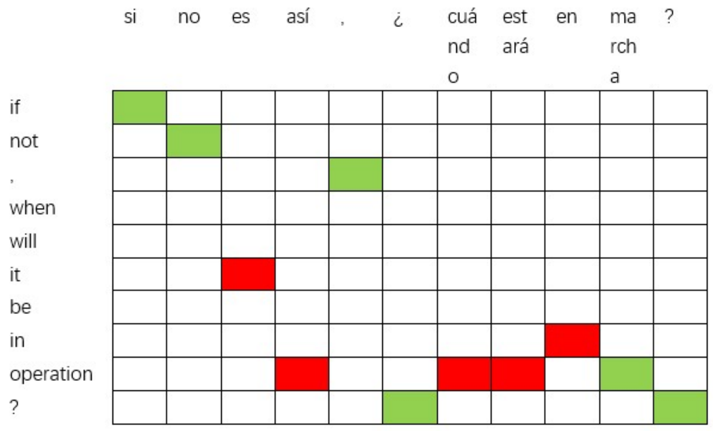

Figure 1: The alignment result

2.4 Further Improvement

As we discussed before, the limitations of IBM-2 is that it does not address the fertility problem and does not consider the alignment dependecy. Thus it can be improved in at least two ways.

To address the fertility problem, we can use the growing alignment method which will be introduced below. Besides, we can also adopt the method of IBM model 3, which adds a model of fertility before the the insertion and lexical translation steps. The fertility is modeled by a display distribution function. In this way, IBM model 3 enables input word to be translated into one multiple words.

To address the problem of alignment dependecy, i.e. to consider that words usually move in groups, we can adopt the method of IBM model 4, which regards that each word is dependent on the previously aligned word and on the word classes of the surrounding words.

3 Growing Alignments

3.1 Method Overview

The growing alignment method consists of the following steps.

1. Estimate IBM-2 for $p(f|e)$.
2. Estimate IBM-2 for $p(e|f)$.
3. Calculate the best alignments with $p(f|e)$ and $p(e|f)$.
4. Calculate the intersection and union of these alignments.
5. Starting from the intersection, apply a heuristic to grow the alignment.

The adopted heuristic is as below. In the implementation process, we find it is important to set a suitable threshold. We can easily find a suitable threshold with several experiments.

1. Only explore alignment in the union of the $p(f|e)$ and $p(e|f)$ alignments.
2. Add one alignment point at a time.
3. Only add alignment points which align a word that currently has no alignment.
4. Calculate the intersection and union of these alignments.
5. alignment points that are "neighbors" (the distance is smaller than a threshold).

3.2 Results and Discussions

Evaluating on 5920 sample, applying the growing alignment method, we obtain F1-score of 0.545, Precision of 0.639 and Recall of 0.476. Compared with IBM-2, F1-score is improve by 9.1%.

We still select the 106th sample from dev.en/dev.es to show the correctly aligned examples and misaligned examples. The result is as shown in Figure 2. From the result, we can find it allows Spanish words to be aligned with multiple English words. The word "estará" is correctly aligned with "be" but wrongly aligned with "will" and "operation". Compared with the result in Figure 1, we can find that the growing alignments bring the word "estará" a correct alignment, but still have the wrong alignment "operation". The wrong alignments can be reduced by minimize the threshold. But in some cases, the smaller threshold will lead to less correct alignment.

3.3 Further Improvement

The grow alignment method can be improved by the following three ways, which are all considering to add additional alignment points.

- grow-diag: besides the neighboring points in the grow method, add the diagonally neighboring alignment points. The diagonally neighboring points are the points that are neighboring the current point on the four diagonal directions, i.e. left, right, up and down.
- grow-diag-final: in addition to the alignment points in grow-diag, add the non-neighboring alignment points between words, of which at least one is currently unaligned, are added in a final step.

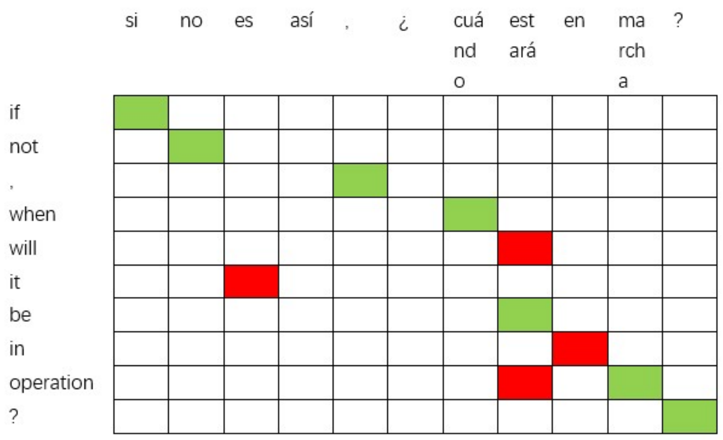

Figure 2: The growing alignment result

- grow-final: In addition to the alignment points in grow, add the non-neighboring alignment points between words in a final step. Among the non-neighboring alignment points, at least one of them should be currently unaligned.