Natural Language Processing 2, Project 1: IBM Models 1 and 2

Pepijn Kokke, Edwin Odijk and **Thijs Scheepers**

Abstract

This paper presents an implementation of IBM models 1 and 2, first described by Brown et al. (1993). These models can be used to align words in parallel corpa. We implemented various initialisations for the models as well as improvements to model 1. We ran experiments with different initialisation parameters as well as improvement parameters. Finally we discuss what effect different initialisations and improvements have on the models performance as well as give an indication which model is the best.

1 Introduction

While machine translation remains a difficult problem to this day, many steps have been take over the past decades to bring us where we are today. One of the first attempts to solve machine translation was made by Brown et al. (1993), who introduced the first five IBM models. While these models have since been superseded by newer, far better models for machine translation, the models by Brown et al. are still used to solve the problem of *alignment*—the problem of finding the words in a target language which correspond to the words in a source language.

In this paper, we implement IBM models 1 and 2, and evaluate their use in solving the problem of alignment. Whereas Brown et al. (1993) restrict themselves to a uniform initialisation of parameters, we will also investigate the use of a *random* initialisation. We also implement and evaluate several improvements to IBM model 1, due to Moore (2004).

Before we start, it should be noted that for the remainder of this paper, we will refer to the source languageas French and the target language as English. This is not only the terminology used by Brown et al. (1993), but also refers to the actual languages we used for alignment.[1]

2 Model

2.1 IBM model 1

IBM model 1 is the simplest of the two we implemented. It's performance is much worse than that of IBM model 2, but it is nevertheless useful to study. The model tries to, for each word in each French sentence, find the word in the corresponding English sentence that it was generated from. This link between two corresponding words carries a weight, and assigning these words to each other forms an alignment. To get a more in-depth definition of the model, let:

$$\begin{aligned} l &= \text{English sentence length} \\ m &= \text{French sentence length} \\ i &= \text{position of an English word in a sentence} \\ j &= \text{position of a French word in a sentence} \\ a_j &= \text{position of an English word that the } j\text{'th} \\ &\quad \text{French word is aligned to} \end{aligned}$$

Ideally, we would want a $p(f|e)$ to translate with, but as this is hard to compute, we get the alignments involved and end up with

$$p(f, a|e, m = p(a|e, m)p(f|a, e, m) \quad (1)$$

This means it assigns an alignment with probability

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

[1] A potential source of confusion lies with our description of our generative model; here we will look at how the French words in the *source* sentence were generated from the English words in the *target* sentence. Therefore, it may seem like the order is reversed.

and a french word with probability

$$p(f|a,e,m) = \prod_{j}^{m} t(f_j|e_{a_j}) \tag{3}$$

which is the translation probability of the French word f_j being translated from the English word it is aligned to.

We now wish to optimize this $t(f_j|e_{a_j})$ using the EM algorithm. To do so, we first perform the **E** step: For each sentence in the corpus, for each English and French word in that sentence we compute the counts:

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

$$c(e_j^k) \leftarrow c(e_j^k) + \delta(k,i,j) \tag{5}$$

where

$$\delta(k,i,j) = \frac{q(j|i,l_k,m_k)t(f_i^k|e_j^k)}{\sum_{j=0}^{l_k} q(j|i,l_k,m_k)t(f_i^k|e_j^k)} \tag{6}$$

We then find $t(f_j|e_{a_j})$ in the **M** step:

$$t(f|e) = \frac{c(e,f)}{c(e)} \tag{7}$$

by looping over all combinations.

We added a number of improvements as described by Moore (2004) to alleviate some of IBM model 1's shortcomings. As words have a tendency not to be aligned to NULL as often as they should under IBM model 1, one of these improvements simply involved adding additional NULL words to the English sentence, thereby increasing the weight of these words and making them more likely to be aligned to. Another problem is over-alignment to rare words, for which we smoothe the counts at the M by adjusting the $t(f|e)$ computation to:

$$t(f|e) = \frac{c(e,f) + n}{c(e) + n \cdot |V|} \tag{8}$$

where $|V|$ represents the amount of lexical items in the English language. Moore (2004) suggests using either an arbitrary estimate based on the language's entire set of lexical items or counting the number of unique words in the English corpus. We opted for the latter, as we assume the data to be an accurate representation of the English language, therefore creating a more accurate estimate. The n parameter is free for tweaking.

2.2 IBM model 2

The second model is similar to the first one, but with the notable addition of the q function, which returns the probability of any given alignment. In model 1, alignments were determined by a constant $\frac{1}{(l+1)^m}$, which we replace by the q function to get

$$p(a|e,m) = \prod_{j=1}^{m} q(a_j|j,l,m) \tag{9}$$

which means our $p(f,a|e,m)$ function changes to

$$p(f,a|e,m) = \prod_{i=1}^{m} q(a_j|j,l,m)t(f_j|e_{a_j}) \tag{10}$$

Introducing this new function complicates the EM algorithm. In addition to the work required to optimise $t(f|e)$, we need to compute the following counts in the **E** step:

$$c(j|i,l,m) \leftarrow c(j|i,l,m) + \delta(k,i,j) \tag{11}$$

$$c(i,l,m) \leftarrow c(i,l,m) + \delta(k,i,j) \tag{12}$$

and optimize the q function in the **M** step:

$$q(j|i,l,m) = \frac{c(j|i,l,m)}{c(i,l,m)} \tag{13}$$

3 Experiments

We ran the models on a parallel corpus taken from the Canadian Hansards, training it on 231.164 sentences of French sentences with their English translations and testing it on 447 pre-annotated sentences pairs with *sure* (unambiguous) and *possible* (ambiguous) alignments. For both IBM models 1 and 2, we retrieve the Viterbi alignments for different parameter settings on each iteration and compute the precision, recall and alignment error rate (AER) by Och and Ney (2003). All models ran 20 iterations of the EM-algorithm.

3.1 Initialisations

For IBM model 1, we initialise the parameters using a distribution.

- Uniform
- Random Dirichlet (3 times)

Additionally we change the smoothing parameter n or the number of NULL words (q_0), while keeping the other at a default. Default values are

$n = 0$ for no smoothing and $q_0 = 1$ for a single NULL word. The following parameter settings where used for training after a uniform initialisation:

- $n = 0$, $q_0 = 2$
- $n = 0$, $q_0 = 3$
- $n = 0.01$, $q_0 = 1$
- $n = 0.005$, $q_0 = 1$
- $n = 0.0005$, $q_0 = 1$

For IBM model 2, we initialise the parameters using a distribution and with the lexical types found using IBM model 1. The following parameter initialisations are tested:

- Uniform
- Random Dirichlet (3 times)
- Lexical types from 5 iterations of uniformly initialised IBM model 1, q is initialised uniformly as well

3.2 Hypothesis

Before running the algorithm we expected the following results:

1. Model 2 will outperform model 1.
2. An improved model 1, either using smoothing or extra NULL words, will outperform regular model 1.
3. Randomly initialized models will yield different results.
4. Initialising model 2 using model 1's lexical types will outperform randomly or uniformly initialised model 2.

4 Results

During the EM-iterations the likelihood grew steadily for each of the models as Figure 1 shows. The purple line in the figure shows model 2 initialised with model 1's lexical types, which was expected to have a higher likelihood after the first iteration. The yellow lines show model 1 using smoothing, which resulted in a lower likelihood. This is due to the smoothing in the denominator of Equation 8. As we will see, the model does not necessarily have a bad precision, recall or AER, on the contrary.

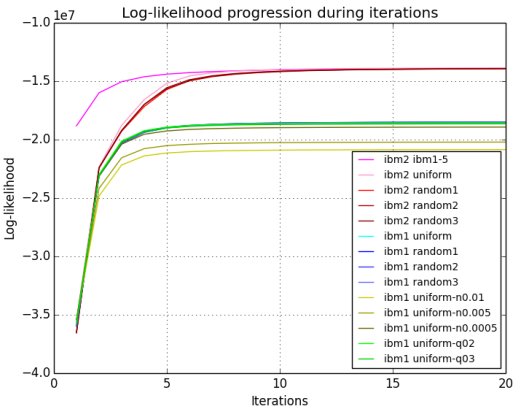

Figure 1: The Log-likelihood progression. An increase as shown in this graph is a good indication the EM-algorithm is working.

The annotated test set contains alignments of two different kinds: an S (sure) alignment, for alignments that are unambiguous, and a P (possible) alignment, for ambiguous alignments. The precision will be measured using both sure and possible alignments, where the recall will be measured by just the sure alignments.

Figures 2, 3 and 4 clearly show that the improvements suggested by Moore (2004) allow model 1 to converge faster. Even after only two iteration results indicated by both the green and yellow lines are promising. Where the two random initialisations of plain model 1 need 10 iterations to catch up. Furthermore, the uniformly initialised model 1 as well as the remaining randomly initialised model 1 still has not come close after 20 iterations.

If looking at the results for 20 iterations, more NULL words yield a better precision. $q_0 = 3$ is the largest amount of NULL words tested and yields the best precision result for model 1. See Table 1 for a better comparison.

When looking at recall in Figure 3 and Table 1 for model 1 smoothing has a positive effect. In fact $n = 0.01$ (which is the largest smoothing parameter tested) yielded the best recall result as well as the best AER result for model 1. Furthermore its precision was surpassed by the NULL word optimisation but only after 8 iterations and by a very small margin.

When looking at model 2 the recall of the model initialised using model 1's parameters (pur-

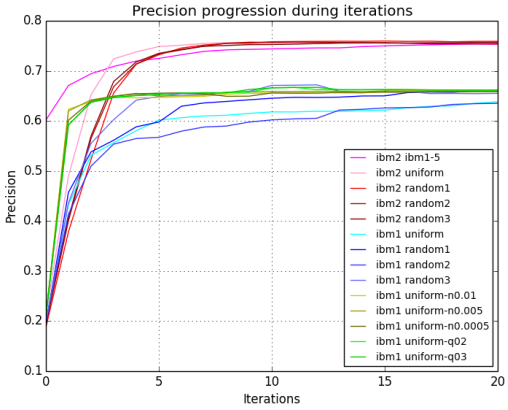

Figure 2: Precision of the *sure* and *possible* alignments.

ple line) is growing slower than the other initialisations of model 2 (pink and red lines). This is contradictory to out hypothesis but can be explained since the parameters of a uniformly initialised model 1 where used. This model (cyan line) also has a relatively low and slow growing recall. This analysis also holds true for the AER and, to a lesser degree, the precision.

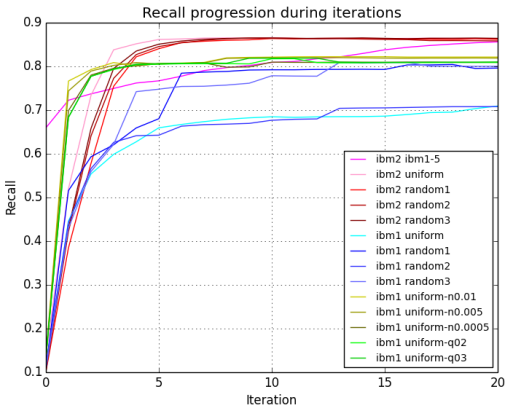

Figure 3: Recall of the *sure* alignments.

Random initialisations of both model 1 and 2 yield different progressions of their recall, precision and AER. This is especially apparent in model 1 where the type of initialisation means the difference between a converged model after 20 iterations and a model which still requires more.

Using the AER metric from Figure 4 and Table 1 we will determine which of the tested models is the best. For model 1 after 20 iterations we found the uniformly initialised model using $n = 0.01$ smoothing to be the best with an AER of 0.2832. And for model 2 after 20 iterations we found our

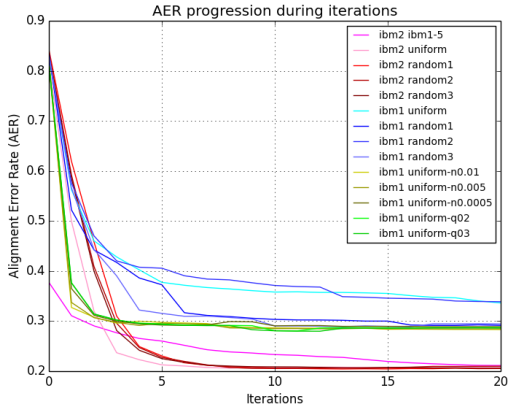

Figure 4: Alignment Error Rate (AER)

first randomly initialised model to be the best with an AER of 0.2053. This model also performed best overall.

As described by Moore (2004) model 1 has many shortcomings some of which cannot simply be remedied without significantly complicating the model. Moore (2004) describe various shortcomings, one of which is "distortion"—the effect that the position of any word in the target sentence is independent of the position of the corresponding word in the source sentence. This is an adequate explanation for model 2 significantly outperforming model 1, as model 2 incorporates word position in its alignment prediction.

Table 1: Test results after 20 EM-iterations.

Model	Recall	Precision	AER
IBM2 with IBM1	0.8561	0.7531	0.2114
IBM2 Uniform	0.8635	0.7557	0.2072
IBM2 Random	0.8621	**0.7590**	**0.2053**
IBM2 Random	0.8583	0.7540	0.2099
IBM2 Random	**0.8643**	0.7570	0.2060
IBM1 Uniform	0.7093	0.6384	0.3363
IBM1 Random	0.7962	0.6608	0.2916
IBM1 Random	0.7080	0.6350	0.3388
IBM1 Random	0.8004	0.6550	0.2944
IBM1 $n = 0.01$	**0.8214**	0.6606	**0.2832**
IBM1 $n = 0.005$	0.8190	0.6599	0.2846
IBM1 $n = 0.0005$	0.8093	0.6584	0.2890
IBM1 $q_0 = 2$	0.8091	0.6614	0.2870
IBM1 $q_0 = 3$	0.8101	**0.6618**	0.2863

5 Conclusion

We introduced our implementation of the IBM models 1 and 2 in order to test various hypothesis

about the initialisations as well as improvements on the model.

Our results show model 2 outperformed model 1, which was expected, even with model 1 benefitting from improvements such as smoothing and extra null words, which allow them to converge quickly and have good results on our test set.

As we expected, randomly initialising the model has varying results. Especially for model 1, the amount of iterations needed for a the model to convergence differs vastly. This is true even though the EM-algorithm is guaranteed to converge to a global optimum for model 1.

We expected that model 2 initialised using model 1 parameters would outperform a randomly or uniformly initialised model. To our surprise this does not seem to be the case. Our uniformly initialised model 1 converged slowly, and using this model (after five iterations) as an initialisation for model 2 made this model converge slowly as well—moreso than one would expect given the performance od the randomly or uniformly initalised models. One further, unsurprising observation about initialising model 2 with the parameters of model 1 is that it seems that the performance of such a model 2 instance depends on the performance of the instance of model 1 which was used.

Python code, that was used to run our models, accompanies this paper and is available on GitHub[2].

6 Future research

For future research one could extend our improvements from model 1 to model 2, as well as run more experiments to find optimal parameters for our improvements. One could also look into the combination of model 1 and model 2, try various types and iteration counts of model 1 to initialise model 2.

References

Brown, P. F., Pietra, V. J. D., Pietra, S. A. D., and Mercer, R. L. (1993). The mathematics of statistical machine translation: Parameter estimation. *Comput. Linguist.*, 19(2):263–311.

Moore, R. C. (2004). Improving IBM word-alignment model 1. In *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics - ACL '04*. Association for Computational Linguistics (ACL).

Och, F. J. and Ney, H. (2003). A systematic comparison of various statistical alignment models. *Computational linguistics*, 29(1):19–51.

[2] http://github.com/pepijnkokke/nlp2