Word alignment models for machine translation

Harsh Jhamtani
Language Technology Institue
Carnegie Mellon University
jharsh@cs.cmu.edu

Abstract

This report discusses various methods for generating word alignment from pairs of sentences. Various **heursitics** alongwith **IBM model 1** and **HMM** based models are tried. A **supervised classification** model is also explored by framing a binry classification problem i.e. predicting whether an alignment should be added or not based on features like cooccurrence statistics, difference between positions, model 1 translation probabilities, and so on. Heursitic methods use statistical similarity measures like Jaccard, Dice coefficient and Ochichai. Variants of HMM models like intersection HMM models are also tried. Best model achieves AER of $\mathbf{13.91}\%$ on test data set.

1 Introduction

Alignments are used to generate phrase tables, which are used in downstream tasks like Machine Translation.

2 Algorithms

Following algorithms are considered:

2.1 Heuristic based approaches

Let x be a foreign language word and e be a target language word. Let $c(x)$ denote the number of sentences in training data in which x occurs. Similarly, $c(y)$ denotes number of sentences in training data where y occurs. And let $c(x, y)$ denote the number of foreign-target language pairs of sentences such that x occurs in foreign language sentence, and y occurs in corresponding target language sentence. Following heuristics are considered:

- Dice coefficient (Dice, 1945)
  $2 * \frac{c(x,y)}{c(x)+c(y)}$
- Pointwise mutual infirmation (Church and Hanks, 1990) :
  $\frac{c(x,y)}{c(x)\,c(y)}$
- Ochichai (Janson and Vegelius, 1981)
  $\frac{c(x,y)}{\sqrt{c(x)\,c(y)}}$

Given a sentence pair, for every foreign word x, a y in native langauge sentence is selected which has the maximum heuristic score with x among all words in the sentence.

2.2 IBM model 1

IBM Model 1 is a word alignment model, which uses an Expectation Maximization Algorithm to compute the lexical translation probabilities in parallel texts.
$E->F$:
$P(f, a|e) = \frac{\epsilon}{(J+1)^I} \prod_{j=1}^{J} tr(f_j|e_{a_i})$
Here f is the foreign / source sentence, a represents alignment, and e represents target language sentence. ϵ is normalization constant, and J represents number of words in f. $tr(f_j|e_{a_i})$ represents lexical translation probabilities. Target words that are not aligned with any source word are aligned with the null token, the probability being given by $tr(f_j|NULL)$

2.3 HMM

HMM based models have been used for word alignment (Vogel et al., 1996). HMM models can capture the fact that neighboring words in source language often align with closeby words in target language. $p(f|a, e) = \prod_j p\left(f_j|e_{a_j}\right) \prod_j p(a_j, a_{j-1})$

The transition probabilities are given as follows: $p(a_j|a_{j-1}) = s(a_j, a_{j-1})$ Consider NS to represent null state in HMM, then $s(a_j, a_{j-1})$ is given

as follows:

$$\mathrm{s}(f,t) = \begin{cases} FTnull\,, & \text{if } f = \text{NS and } t = \text{NS} \\ Tnull\,, & \text{if } f \neq \text{NS and } t = \text{NS} \\ Fnull\,, & \text{if } f = \text{NS and } t \neq \text{NS} \\ t - f\,, & \text{if } f \neq \text{NS and } t \neq \text{NS} \end{cases}$$

2.4 Supervised classification

Whether a pair of e, f should be added as an alignment or not can be framed as a binary classification problem. Algorithms like perceptron and SVM can be used to leverage a bunch of features like distance in positions, model 1 translation probabilities, etc.

3 Experiments and Results

3.1 Heursitics

Traning time, AER score and BLEU score are reported for each of the three heuristics in Table 1. As can be seen , Dice coefficient outperforms other heuristics. Moreover, heuristics scale up to large data easily. time for training increases almost linearly, but is still small (It took less than 10 mins to train for Dice using 100K pairs of sentences). In general AER scores and BLEU scores are poor using heuristic methods.

Method	Training data size	AER	Time	BLEU
Dice	1000	0.48	44s	15.42
PMI	1000	0.50	1m 17s	12.89
Ochichai	1000	0.53	1m 8s	13.85
Dice	10000	0.43	8m 24s	19.39
PMI	10000	0.47	10m 17s	15.65
Ochichai	10000	0.49	8m 11s	18.66

Table 1: Results for various heuristic measures

3.2 IBM Model 1

To impose 0.2 probability of aligning to null, I set $p(a_i) = 0.8/N$ if a_i is non-null and 0.2 otherwise.

3.2.1 Intersection of models

Table 2 shows the intersection from results of E->F and F->E models improves AER significantly, though it decreases BLEU sscores as well.

3.2.2 Tuning of mass allotted to *NULL*

As mentioned earlier, $p(a_i) = 0.8/N$ if a_i is non-null and 0.2 otherwise. Instead of 0.2, I experimented with other values (0.1,0.3) - however this does not change AER at all.

Method	AER	Time	BLEU
Model1 $E->F$	0.40	35s	16.01
Model1 $F->E$	0.37	45s	14.96
Model1 INT	0.30	1m 11s	13.65

Table 2: Results for various variants of IBM model1 1. Both E->F and F->E have similar performance. Model1 INT, which refers to alignment obtained from intersection of alignments of E->F and F->E, achieves significant improvment in AER

3.2.3 Impact of number of iterations in EM procedure

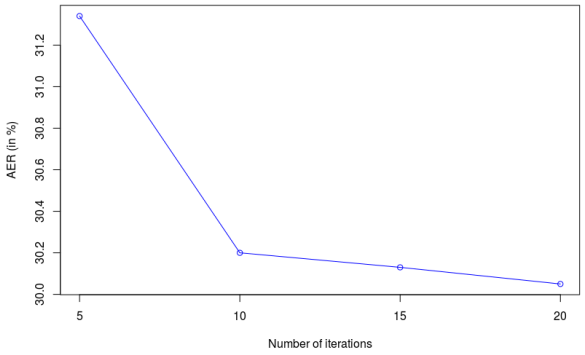

Figure 1: AER for the IBM *INT* model as the number of the iteration of EM are varied from 5 to 20. As can be seen, improvement in AER is negligible after 10 iterations

Figure 1 shows how the AER changes as the number of iterations for the EM training procedure are changed. Note that after 10 or so iterations, there is hardly any change in AER. However, the training time increases linearly with the number of iterations. Specifically, training times for 5,10,15,20 iterations are 27, 58, 71, and 91 seconds respectively.

3.3 HMM Alignment model

This section discusses experiments related to HMM models. I will discuss imapct of intialization, number of training epochs, and addition of end state probability, alongwith reporting results on large data sets.

3.3.1 Initialization of parameters

(E->F) HMM alignment model with random initialization of paramters achieves AER of 0.459 when trained on 1K sentences pairs with 6 iter-

ations of EM algorithm. On using probabilities from IBM Model 1 trained on same data to intialize emmission probabilities, AER reduces to 0.312.

3.3.2 Intersection of HMM models

Table 3 shows the how intersect hmm leads to improvement in AER. All the results are when models are trained for 6 iterations using EM. Both E->F and F->E have similar performance. Intersection model, denoted by *HMM INT* leads to a AER much lower than either of the HMM models. From hereafter, I will discuss results in HMM INT model, as it outperforms the two other models.

Method	AER	Time	BLEU
HMM $E->F$	0.312	31s	10.23
HMM $F->E$	0.263	35s	11.69
HMM INT	0.215	64s	11.68

Table 3: Impact of doing *intersection* of two HMM models trained from source to target, and another from target to source. All experiments are using data set of *1000* sentence pairs.

3.3.3 Impact of number of training epochs

Figure 2 shows log likelihood of the data for (E->F) and (F->E) models with number of iterations.

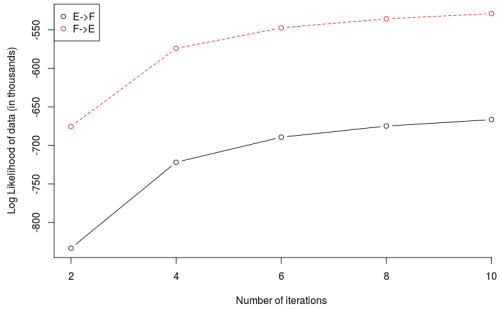

Figure 2: Log likelihood of (E->F) and (F->E) models as the number of the iteration of EM are increased. All results are when training is doneover $10K$ pairs of sentences.

Table 4 shows how the performance of the HMM INT model varies with the number of training iterations. As can be seen, there is little performance gain after 10 iterations. In fact, performance deteriorated when model is trained for 20 iterations. Based on the four values of the number of training epochs, model performs best when using 15 iterations.

Epochs	AER	BLEU
5	16.64%	18.04
10	16.28%	18.20
15	**16.27%**	**18.45**
20	16.88%	18.40

Table 4: Performance of the HMM INT model as the number of training epochs for EM algorithm are changed. All results are on taining data set of $10K$ pairs of sentences, with emmission probabilities initialized using IBM model 1 trained for 20 iterations.

3.3.4 Tuning initialization of HMM parameters

As described earlier, model 1 trained on same data is used to initialize emmission probabilities. Due to this, paramters for training of model 1 impact performance of the HMM models also. Table 5 shows how the performance of HMM INT changes as the number of training epochs of the corresponding model 1 are changed.

Epochs of intiliazing Model 1	AER
5	16.98 %
10	16.67 %
15	**16.09** %
20	16.27 %

Table 5: Performance as the number of training epochs for EM algorithm are changed for the *Model 1*, which is used to initialize paramters of HMM model.

3.3.5 Addition of end state transition parameter

Adding an end state transition, similar to starting probabilities, can help in improving the accuracy. For example, it can capture the fact the English-French are often aligned diaganolly, leading to favoritism to end hmm state sequence with words at the end of the sentence. I consider following two techniques for adding end state probabilities:

1. Considering $endProb(s)$ for every state s
2. Considering $endProb(i)$ such that $i = numOfStates\text{-}s$. This can deal with the fact that there are sentences of varying length in the data set.

The first technique does not lead to reduction, but second way of adding end state probabilities

leads to reduction in AER. However, it also lead to a slight reduction in BLEU from 18.27 to 17.93. Specifically, following results are obtained for a data set of 10K pairs of sentences:
Precision: 0.957
Recall: 0.740
AER: **15.47**%
BLEU: **17.927**

3.3.6 Experiments on large data set

#Sent. Pairs	AER	Time	BLEU	Memory
20K	14.32%	34m	19.80	971M
30K	12.91%	1h 07m	20.40	1.0G
100K	**10.30**%	**2h 45m**	**23.12**	**1.4G**

Table 6: AER, BLEU score, and training time with larger data sets.

Table 6 shows experiments on larger data sets. For 100K sentence pairs, HMM INT model takes xx minutes to train and achieves AER of 10.30%, and BLEU score of 23.12. It took 2 hours 45 minutes to train on 100K sentences, with 1.4GB memory consumption.

3.4 Supervised classification

Whether a pair of e, f should be considered an alignment or not is framed as a binary classification problem. Following features are considered for a pair of word f_j and e_i.

- (f_j, e_i) is in alignment predicted by E->F HMM Alignment model or not
- (f_j, e_i) is in alignment predicted by F->E HMM Alignment model or not
- $p(f_j|e_i)$ as per IBM model 1 (E->F)
- $p(e_i|f_j)$ as per IBM model 1 (F->E)
- $\sqrt{p(f_j|e_i)p(e_i|f_j)}$
- $|\frac{i}{E} - \frac{j}{F}|$, where E and F denote length of english and french sentences respectively.
- A bias term

Training data is collected by sampling pairs of english and french word pairs from 1000 parallel sentences. Ground truth labels are already available. Specifically, 3233 positive pair words and 2555 negative pairs of words are sampled.

An averaged perceptron is trained for the above mentioned classification problem. Training of perceptron takes less than a minute, as data set is small and features are few. At time of prediction, a $thresh$ can be varied to trade-off between precision and recall, and is thus a paramter to tune i.e. an alignment is added if $w^T f > thresh$. For $thresh = 0.3$, results are as follows:
- For 1000 sentence pairs, AER improved from 21.06% to 19.98%, an improvement of 1.08 percentage points
- For 10K sentence pairs, AER improved from 15.47% to 14.06%

Since data path is not passed to word aligner class, I had to hardcode the path to validation data ground truth. To ensure that submitted code runs while testing on any other system, I use perceptron weights in the code (there are 7 features only, and hence 7 weight values). Note that once hmm model has trained, output of which is a feature for perceptron, the training of perceptron itself takes only few seconds, which means training time is governed by time taken to train hmm. 'HMM' would run the supervised model, which also uses hmm model result as one of the features. 'HEURISTIC' would just run the hmm model

4 Conclusion

IBM model 1 was able to achieve AER of 30% for 10K sentence pairs. Intersection of HMM models achieves AER close to 16%. Initializing HMM emmission probabilities using IBM model 1, and adding end state probabilities, leads to signifcant improvement in AER. However, recall suffers in intersection HMM model due to the nature of intersection operation. A simple model like perceptron trained on very few features and on small labelled data provided improvement of about 1.5 percentage points in AER. With more sophisticated learning algorithms and features, supervised approaches might be able to perform even better. The final AER on test data is $\mathbf{13.91}\%$, and corresponding BLEU score is $\mathbf{18.3}$. Training of this model took **51 minutes**, with **903 MB** memory consumption.

References

[Church and Hanks1990] Kenneth Ward Church and Patrick Hanks. 1990. Word association norms, mutual information, and lexicography. *Computational linguistics*, 16(1):22–29.

[Dice1945] Lee R Dice. 1945. Measures of the amount of ecologic association between species. *Ecology*, 26(3):297–302.

[Janson and Vegelius1981] Svante Janson and Jan Vegelius. 1981. Measures of ecological association. *Oecologia*, 49(3):371–376.

[Vogel et al.1996] Stephan Vogel, Hermann Ney, and Christoph Tillmann. 1996. Hmm-based word alignment in statistical translation. In *Proceedings of the 16th conference on Computational linguistics-Volume 2*, pages 836–841. Association for Computational Linguistics.