CS 224N: Machine Translation (PA2)

Sandeep Sripada(ssandeep) & **Venu Gopal Kasturi(venuk)**

1 Introduction

For Programming Assignment-2, the task was to implement several word-level alignment systems. We implemented a base alignment system which matches up words based on superficial statistics and IBM models 1,2,3 based on the paper by [Brown1993]. We experimented with a few variations of IBM Models like diagonal assignment in case of ambiguity, bucketing in model 2 for distortion probability estimation and fixed probability NULL handling etc. The systems were trained and tested on on the Hansard corpus, consisting of parallel English and French sentences. The alignment system was combined with language models (best one from PA1), a greedy decoder (to find the most probable English sentence) to build a complete statistical machine translation package.

The best alignment system was **Model 2 with buckets** with an AER of **0.26**, Precision of **0.7**, and Recall of **0.75** with **500k** training size. The results on the 10k dataset are as shown in Table 1. The plot of the scores of various models as the training set is varied is in Figure 1, Figure 2, Figure 3.

2 Description and Analysis

2.1 Model 0 - Surface Statistics Word Aligner

2.1.1 Implementation

This model works on the very simple heuristic of using 'Point wise mutual information' from the set of sentence pairs. It assumes equal probability for any word in the target language to align to a word in the source language while training from a bilingual corpus. The best alignment for a pair of sentences is obtained by aligning an english word with a french word for which $\frac{P(f,e)}{P(f)P(e)}$ is maximum.

2.1.2 Improvements

The draw back of this alignment is that many french words align to the same english word in most of the sentences because the maximum pointwise mutual information was higher for the same word.

For example in Figure 4, we see that all the french words are aligned to 'overproduction'. In order to overcome this problem, we modified the model using the heuristic that each pair of 'free' french and english words are aligned to one another on the decreasing order of their 'Point wise mutual information'. This would make sure that once a word has been selected for alignment, it would not figure in subsequent selections as it would be marked 'not free'. This significantly improved the accuracy of the alignment model and sample alignments for the same sentence used above is shown in Figure 5.

The basic surface statistics model on a training set size of 10k had an AER of 0.67, precision of 0.33 and recall of 0.33. Using the improved model, our scores for the same training set size was **0.36, 0.63, 0.67** respectively.

2.1.3 Analysis

This model allows for out of order alignments i.e. non-diagonal alignments would also be selected if they have a high PWI score.

The problem with this generative model is that it does not have the provision for a single english word generating multiple french words. Hence this results in low recall when the length of the french sentence is larger than the length of the english sentence as the remaining words align to NULL.

2.2 IBM Model 1

2.2.1 Implementation

In Model 1, we use EM algorithm to learn the translation probabilities $t(f|e)$. We start with uniform translation probabilities and using that we estimate the probabilities of the alignments a for a sentence pair (f, e) using:

$$P(A, F|E) = P(A|E) * P(F|E, A) \tag{1}$$

which can be written as:

$$P(a, f|e) = \prod_{j=1}^{J} (P(a_i|i, J, I) * t(f_j|e_{a_j}) \tag{2}$$

where I=length of the english sentence, J=length of the french sentence, e_{a_j} is the english word aligned to the french word f_j, t is the translation probability. We estimate the translation probabilities from each possible

Table 1: AER, Precision, Recall values for models with 10k sentences (with results of no additional training) FD: Fixed distortion probability, DA: Diagonal assignment, FNP: Fixed NULL probability

Model	AER	Precision	Recall
M0 Base	0.48	0.53	0.5
M0 Improved	0.36	0.63	0.67
M1 FD+DA	0.36	0.58	0.74
M1 FD+DA+FNP	0.32	0.62	0.71
M2 Buckets	0.38	0.6	0.63
M3	0.36	0.53	0.5

Table 2: AER, Precision, Recall values for different variations of IBM Model 1 (10k training sentences) FD: Fixed distortion probability, DA: Diagonal assignment, FNP: Fixed NULL probability

Variation	AER	Precision	Recall
Base	0.48	0.51	0.53
FD+DA	0.36	0.59	0.72
FD+DA+FNP	0.32	0.63	0.72

alignment and combine those estimates weighted by the probability of the alignment and use these again in a loop till EM converges.

For getting the best (Viterbi) alignment in model 1, we take the best alignment of each word independent of the decision of the best alignments of neighboring words.

$$\hat{A} = \arg\max_{a_j} t(f_j|e_{a_j}) * P(a_i|i, J, I) \quad (3)$$

The distortion probabilities in the above case are handled/calculated as explained in the following section. The convergence condition for EM is that the new estimates and old estimates do not differ (euclidean distance) by more than 10^{-3}. For model 2, we have a limit on the number of iterations and keep a hard limit of 50.

2.2.2 Improvements

One change that reduced the AER was to make assignments along the diagonal in case of a tie in translation probabilities between two possible candidates. So from among the set of candidates whose translation probabilities are equal and the maximum, the one which is closest to the diagonal is chosen as a valid alignment. This is much better than just choosing the first one or the last one depending on the way comparison is made.

(Eg: (word:score pairs) w_1:10, w_2:10, w_3:10 are the set of candidates, then w_1 would be chosen if the condition (greater than) returns the first to cross the default value else w_3 (greater than equals)).

NULL word handling: We handle distortion probability of NULL word by assigning it a fixed probability of 0.1 and distribute the remaining 0.9 over all the remaining destination words in the sentence. So in EM, the distortion probability used is 0.1 in case of a NULL word and $\frac{0.9}{len_e}$ in case of others. We also tried using uniform distortion probability of $\frac{1}{len_e+1}$ (adding 1 for NULL word) but the previous method performed better. The value of 0.1 was set after trying out a few probabilities.

In case of NULL in translation probabilities, we handle it as follows:

1. Fixed probability: The training proceeds without the NULL token and after EM converges, we assign NULL a constant probability of $\frac{0.1}{V_f}$. So, $t(f_j|NULL) = \frac{0.1}{V_f}$ where V_f=number of french words encountered. The value of 0.1 was decided after trying out a few probabilities based on the output from IBM base model 1.

2. The NULL translation probability is learnt along with all the other words by placing a NULL token in the english sentence while training. This case does not warrant explicit handling for NULL.

The first method of handling NULL worked better as seen in Table2 on 10k training sentences. If translation probability of NULL was learnt along with the combinations, then the probability of a french word given NULL becomes quite high thereby overpowering actual probabilities of the right translation word.

2.2.3 Analysis

With the base model 1, the alignments did not match '.' with a '.' but rather matched it with a NULL. This happened with all sentences and upon inspection the reason turned out to be that the NULL probability was way to high and was overpowering most legit translations. This is one of the problems with model 1 but we bypassed this by using the fixed null probability tweak. This resulted in more matches and better scores.

Also, with the diagonal alignments, appropriate matches were predicted at places where possible. If this was not set, then the assignments were mostly either skewed towards the beginning or towards the end of the sentence.

2.2.4 Problems

Contextual usages of words like 'ce' for this/what/it/that/those is always wrongly assigned to either '.' or NULL. This may be because of the distribution of probability mass over all possible usages which reduces the value leading to other possibilities being selected.

Function words in french like 'ce', 'se', 'à', 'que', 'on', 'de' etc. are mapped to some occurrence of an english counterpart like 'the', 'if', 'of' etc. even when they were supposed to be generated from NULL.

2.3 Model 2

2.3.1 Implementation

Model 2 proceeds on the same lines as model 1 except for a difference in the way distortion probabilities are handled. We implemented two variants of Model 2: (a) Count based learning of probabilities, (b) Counts based translation probabilities, bucket based estimation of distortion probabilities.

In the count based method (a), we perform the same count update mechanism for translation probabilities as in model 1. Similar method is used for counting the distortion counts according to the formula:

$$d(a_i, i, len_e, len_f) += \frac{P(a_i|i, len_e, len_f) * t(f_j|e_{a_j})}{total_i} \quad (4)$$

where $total_i$ is the amassed count for the given french word. This is useful for quick normalization rather than running through the countermap structure to get the counts, which can be quite expensive. For the first EM iteration, we use a constant distortion probability of 1 and for all subsequent iterations the normalized count is used.

Model 2 also has the framework in place for 'transfer of parameters' to happen from model 1 where the initial values would be taken from the training of model 1. This did not have a major effect in AER scores but might be extremely useful in cases where the number of iterations cannot be large for EM to converge (real-time systems).

While calculating the best alignment, the same procedure as Model 1 is employed except for the inclusion of the distortion probability:

$$\hat{A} = \arg\max_{a_j} t(f_j|e_{a_j}) * P(a_i|i, I, J) \quad (5)$$

Table 3: AER on varying bucket sizes on IBM Model 2, 10k dataset

Bucket Size	AER
30	0.40
20	0.38
10	0.39
5	0.39

In the count, bucketing based method (b), the translation probabilities are calculated exactly the same but there is a difference in the way distortion parameters are estimated. In model 2, the distortion probabilities are not independent of i, so we use a bucketing mechanism to accumulate counts based on the values of english position, french position, english length and french length.

$$P(a_i|i, len_f, len_e) \propto d(bucket(a_i - i * \frac{len_e}{len_f})) \quad (6)$$

where the distortion probability is parameterized by the displacement from the diagonal.

Buckets We have tried using: (a) absolute displacement bucketing where the counts of all terms at equal displacement on either side of the diagonal are binned together and (b) exact displacement bucketing where the terms to the left of the diagonal are binned separately from the ones to the right of the diagonal. The results of the latter method were better when compared to the former as more information is captured by maintaining the exact displacement. The results of varying the bucket size is as shown in Table 3.

The most important part here is to maintain the bucket distribution to be a proper probability distribution i.e. since $\sum_{a_i} P(a_i|i, len_f, len_e) = 1$, we need to ensure that the bucket distribution for a fixed i, len_e, len_f needs to sum to 1. To handle this, we maintained an entry for each i, len_e, len_f that occurs in our training sentences that holds the corresponding bin counts for various indices. This helped us in easily adding a NULL token at the end with a constant probability of 0.1. The NULL token was added to each of entries as a separate key in the bin counter. Also, the probability of NULL for each entry in the countermap is equal to the overall mass of null token divided by the total number of entries among which it should be shared. This ensures that the resulting distribution is a probability distribution even after adding NULL. The overall mass of the null token was fixed to 0.1 after trying out various values.

2.3.2 Improvements

We continued to use the case where diagonal alignments are preferred among candidates with equal probabilities

of being generated from english words. The introduction of bucketing proved to be effective when compared to the count based method. In the count based method, the probabilities were scattered over various lengths of english and french sentences (the data was sparse), whereas in the bucketing case the final counts were placed based on the bucket index returned.

Code optimizations:

- While maintaining a count for the distortion probabilities, rather than initializing the structure to 1 for all possible values, we simply used a flag to indicate whether it is an initial iteration or not. Using this appropriate starting probability was used.
- For normalization we used an additional map to maintain the appropriate counts rather than accessing the structure again to get the total counts. This speeds up the execution by quite a bit.

2.3.3 Analysis

The french word 'à' is mapped to many types english words in different sentences like to/by/in/of and the model is unable to guess that correctly.

2.4 Model 3

2.4.1 Implementation

Model 3's generative story includes the following parameters:

- $t(f|e)$: probability of an english word produces a french word.
- $n(\phi_i|e_i)$: probability of the ei th word in the english sentence to have a fertility of ϕ_i.
- $d(j|i, I, J)$: the probability of a french word generated in the 'j' th index given the index 'i' in th english sentence.
- $p1$: the probability of generating a word from the NULL.

IBM Model 3 needs aligned sentence pairs to learn its various parameters. For this we have used Model 1 to give us a set of alignments which are used to train the model and then employed Expectation Maximization algorithm to estimate various parameters.

Process followed:

1. Train Model 1 with the input pair of sentences
2. For each sentence pair, calculate a subset of possible alignments using Model 1, including the Best alignment possible
3. Learn the translation probabilities learnt from Model 1 as the initial values for Model 3 as well
4. Initialize n, d, p values uniformly
5. Take the best alignments of the sentence pairs from Model 1
6. Run EM Algorithm to :
   (a) Calculate the probabilities of the alignments using the parameters being learnt
   (b) Calculate the parameter values using the alignments and the freshly calculated probabilities
7. Iterate until convergence

Difficulties faced during the training phase:

- For calculating the alignment probability using Model 3, we have used the simplification that higher order fertility factorials ($> 12!$) are trimmed (to $12!$).
- We have to smooth the probability distribution even though we are provided test set as part of the train set, because we are going to calculate various other possible alignments.
- To obtain a smoothed probability distribution for the parameter values, we are estimating the number of possible unseen events and allotting a probability mass of 0.2 and then distributing it uniformly amongst those unseen events. And instead of storing each unseen event's probability explicitly (which is leading to huge blow up in the size of the map required), we are recognizing it as an unseen event and returning the probability for that directly.

Get Best Alignment:

To obtain the best alignment for a given sentence pair using Model 3: (a) begin with the best alignment for the same sentence pair using Model 1, (b) for each word alignment, do a greedy hill climbing search by shifting the french word's alignment to the english words which immediately precede or succeed the current alignment, to obtain a better alignment.

2.4.2 Analysis & Improvements

The problem with Model 3 has been with the fact that our initial alignments are the best alignments alone taken from Model 1, which is not ideal. We need to take a larger set of good alignments from Model 1 so that we can further estimate the parameter values better. Because of the lack of this, the parameters learnt are not accurate and they are biased towards the alignment generated by Model 1. This is resulting in the greedy hill climbing to fail as it is getting stuck in local optima. Hence the alignment returned by Model 3 is same as that returned by Model 1. The further improvement would be to obtain a richer set of good alignments as the training corpus for Model 3.

3 Decoder results

We have used Kneser-Ney trigram model as our best language model from assignment 1 and ran the decoder tests.

The analysis of the decoder results on some of the sentences selected by the greedy decoder using model 1 is as follows:

```
french string: monsieur le Orateur , ma question
  se adresse  le ministre charg de les transports .

actual english string:
Mr. Speaker , my question is
  directed to the Minister of Transport .

Decoded english string: the Speaker , my question
  of the Minister of the window .
```

Model 1 does not take order of the words into account and considers the alignment of each word to be placed anywhere. Hence the generated words are jumbled.

```
  french string: en outre , elle pourrait constituer
    une srieuse menace pour la Confdration et le
    unit nationale .

  actual english string:
in addition , it could become
    a serious threat to Confederation and
    national unity .

  guessed english string:
could she furthermore , a
    serious unfit for the and in the national unity
```

Here we can notice that the we are getting synonyms for the content words like 'in addition' , 'furthermore' etc. The guessed sentences are in fact in the context of the actual sentences but they are syntactically wrong because of Model 1's lack of provision for word ordering.

We illustrate the examples below to show that the guessed english strings do give us some information about the context but not the entire meaning.

```
  french string: il y a beaucoup  dire sur les
    deux aspects de cette question .

  actual english string: there is a lot to be
    said on both sides of that question .

  guessed english string: there has to say on
    this issue of it .
```

Functional words are not translated properly but content words are translated to their synonyms.

The BLEU-4 scores for IBM Model 1 is 0.056 and the log Ngram values are: -0.60 (1Gram)/-2.05 (2Gram)/-3.13 (3Gram)/-4.46 (4Gram).

4 Future Work

- If rare combinations occur with count (frequency) less than a particular threshold, then trust those pairings. This would preserve the pairings rather than selecting NULL or other combinations.
- Since the models are not able to correctly guess the mapping of functional words, trying phrase based translation might help in overcoming that problem.
- Rather than using rough diagonal displacements to generate bucket index, one could use more information like word structure while generating bucket indices for unequal lengths of sentences.
- Another possibility is to use the sentence structure (syntactic) to first parse the french sentence and then use the word types to generate appropriate english words. This way the content words, function words might be mapped correctly.

5 Contributions

Sandeep Sripada: Worked on Models 1, 2 and improvements. Worked on the report for corresponding models.
Venu Gopal Katuri: Worked on Models 0, 3 and improvements. Worked on the report for corresponding models.

References

[Brown1993] The mathematics of statistical machine translation: parameter estimation.
Peter F. Brown, Vincent J. Della Pietra, Stephen A. Della Pietra, and Robert L. Mercer.
Computational Linguistics, v.19 n.2, June 1993.

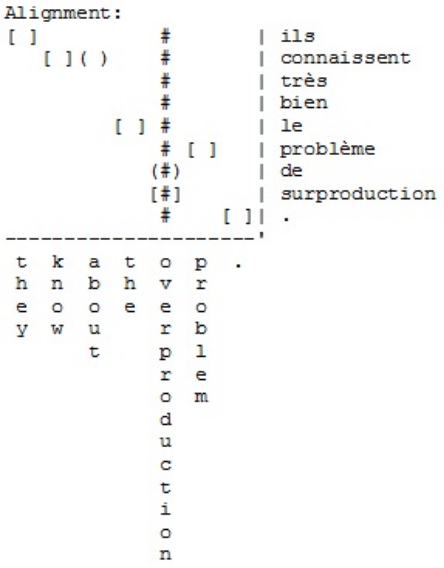

Figure 4: Alignment using simple PWMI

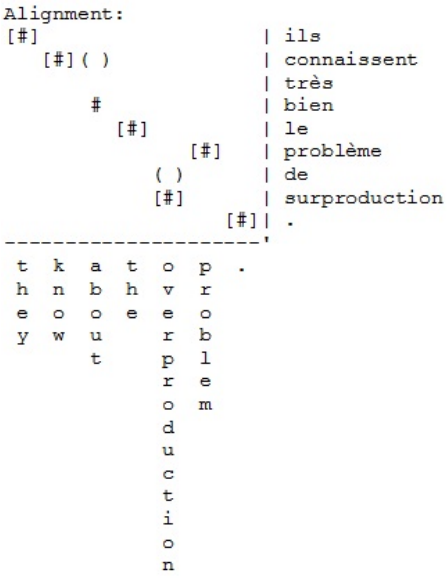

Figure 5: Alignment using modified PWMI

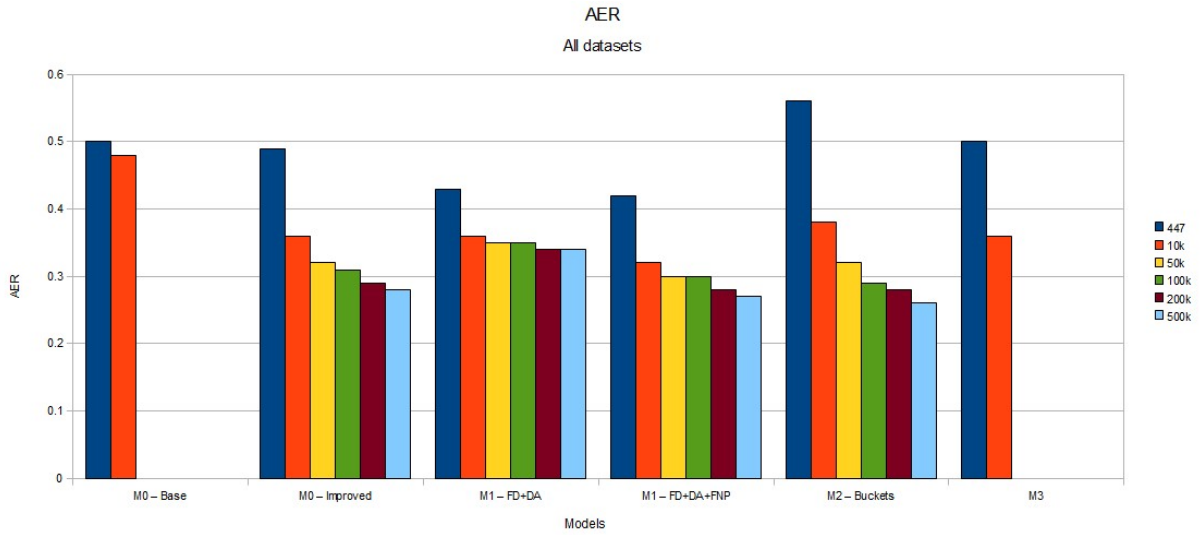

Figure 1: AER for all models as training set is varied

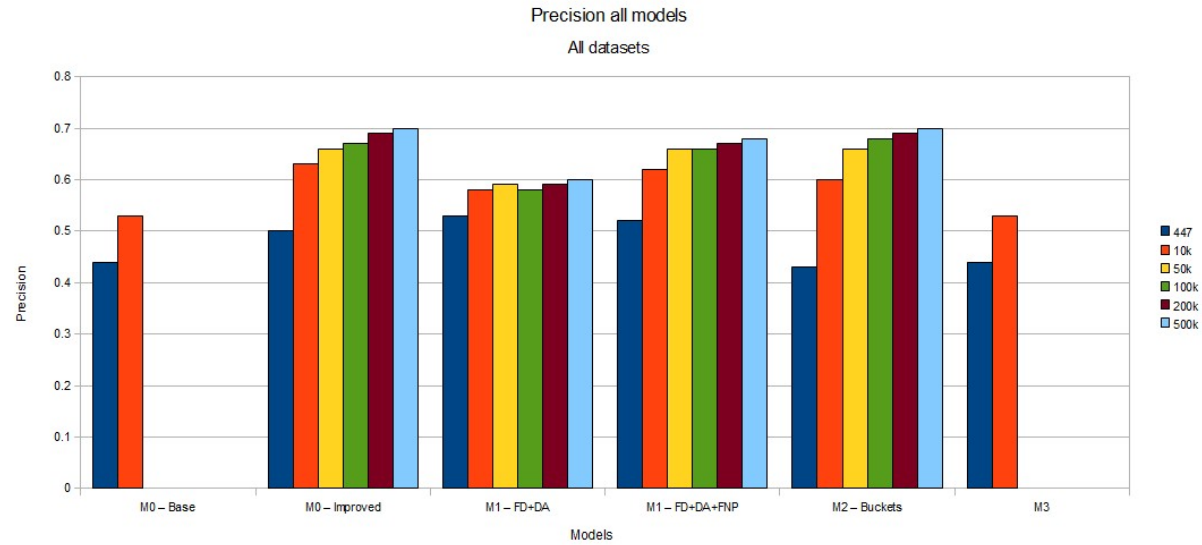

Figure 2: Precision for all models as training set is varied

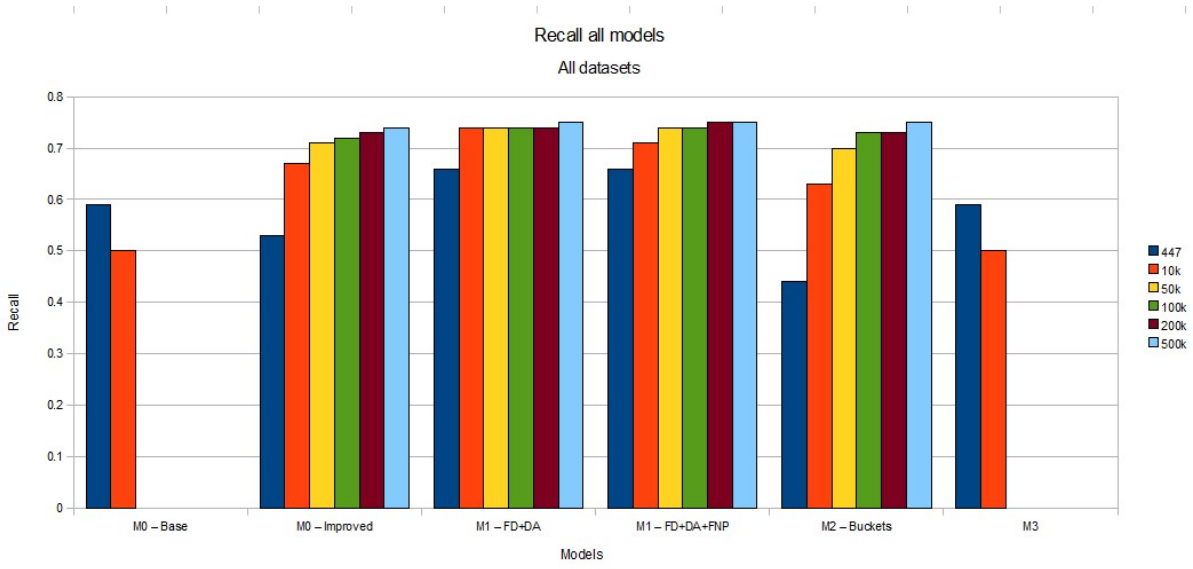

Figure 3: Recall for all models as training set is varied