# CS224N Fall 2014 - Project assignment 1

SUNet ID: ehrhardn
Name: Nicolas Ehrhardt
Collaborators: None

By turning in this assignment, I agree by the Stanford honor code and declare that all of this is my own work.

## 1 Word alignments

Here is the table of the results obtained on the test set.

| PMI | French-English | Hindi-English | Chinese-English |
|---|---|---|---|
| Precision | 0.3117 | 0.1971 | 0.2032 |
| Recall | 0.3026 | 0.1568 | 0.1814 |
| F1 | 0.3070 | 0.1747 | 0.1917 |
| **AER** | **0.6916** | **0.8253** | **0.8083** |
| Model 1 | French-English | Hindi-English | Chinese-English |
| Precision | 0.5970 | 0.4665 | 0.4291 |
| Recall | 0.7422 | 0.3714 | 0.3831 |
| F1 | 0.6617 | 0.4135 | 0.3552 |
| **AER** | **0.3700** | **0.5864** | **0.5952** |
| Model 2 | French-English | Hindi-English | Chinese-English |
| Precision | 0.6264 | 0.4687 | 0.4328 |
| Recall | 0.7400 | 0.3772 | 0.3864 |
| F1 | 0.6785 | 0.4180 | 0.4083 |
| **AER** | **0.3222** | **0.5752** | **0.5917** |

Table 1: Performance of PMI model and IBM model 1 & 2

As expected PMI has the lowest AER for all languages, IBM Model 1 performs better than PMI and Model 2 slightly improves the AER from Model 1. The hindi language seems to see almost no improvement from model 1 to model 2 in terms of F1 score. This is probably related to the fact that we don't have a lot of sentences available for this language.

All of these approaches seems to perform the best on French-English translations. One would probably argue that since these two language share a good amount of vocabulary (English has a lot of French word, especially in some specific area) simple word to word translations system like the one we have implemented will yield better results. On the other hand, Indian or Chinese translation will induce a lot of reordering of the words of the sentence before they can be translated in a word to word fashion. Although Model 2 is meant to account for reordering, it won't be able to capture semantic structure like the one we have seen in [David Chiang. 2005. A hierarchical phrase-based model for statistical machine translation].

## 1.1 Development details

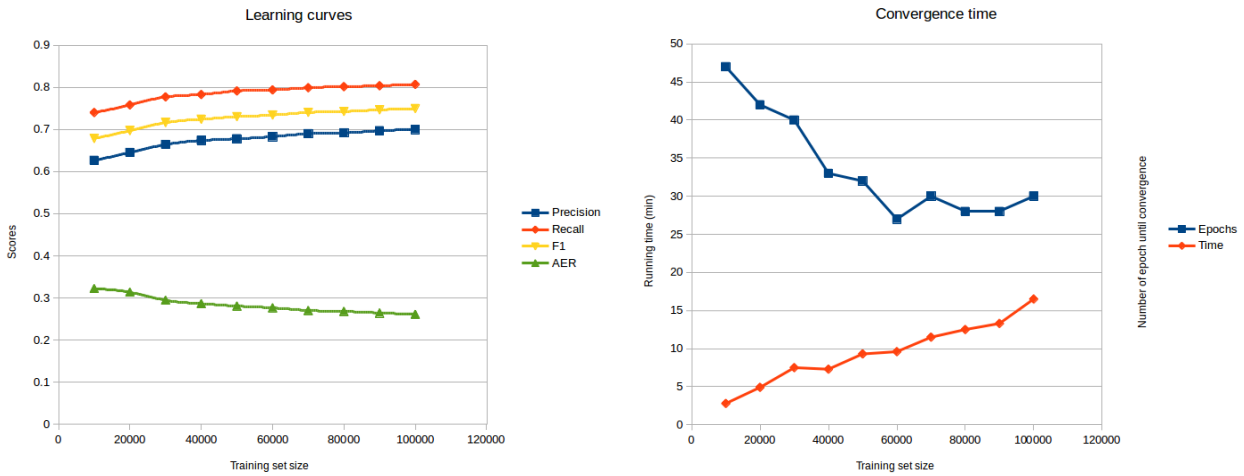
I used two stopping criterion. The first is a hard threshold on the number of iterations, in my case, 50 iterations. The second is a stopping criteria based on the average square distance between distributions. If this threshold is inferior to $10^{-6}$ then the training is stopped. Using this criteria, IBM model 1 runs in around 10 iterations and model 2 around 30 iterations.

I also observed that my results depended a lot on the initialization of the probabilities for model 2. Surprisingly, initializing the lexical probabilities to the values of model 1 does now always yield to the best outcome on our test set (sometimes initializing uniformly gave me better results). However, it consistently decreased the number of iterations to converge (usually around 10 less). I have to say that I was still initializing the position probabilities at random which has to be factored in this statement.

As for my implementations of IBM models, the worst-case complexity of each period is in $O(\max(N_p \times M_f \times M_e, N_f \times N_e))$ where:

- $N_p$ is the number of sentence pairs
- $N_f$ is the number of French tokens
- $N_e$ is the number of English tokens
- $M_f$ is the maximum number of tokens in a French sentence
- $M_e$ is the maximum number of tokens in an English sentence

## 1.2 Learning curves [bonus]




(a) Learning curves

(b) Running time curves

Figure 1: Algorithm performance

I implemented a multi-threaded version of my code to be able to run the model 1 and 2 much faster. I observed a 2× performance gain for the small dataset of 10,000 training

examples. My optimization is quite simple. Since the E-step consists in adding the probabilities to the counts, and this operation can be done in parallel. We can simply send a fraction of the sentence pairs to different workers who will concurrently increase the counts before the normalization of our conditional probabilities.

I used and `ExecutorService` with 10 threads. I had to make sure that I was using a `ConcurrentHashMap` otherwise the thread would have modified a map that is not implemented to handle parallel requests. Unfortunately, I started to get an `OutOfMemory` error at $90,000$ examples, which I solved by implementing in-place normalization of counts.

The convergence time seems to be roughly linear with the number of training examples. The number of epochs necessary is diminishing. This is expected since the algorithm sees more pairs at each iterations, hence each estimation step is more efficient.

### 1.3 Quality of the results

The most common mistake made by all the models is assigning a word to several different words that are not related. This tells me that the training set size needs to be much bigger for the model to be able to behave correctly in sentences including words that are not common. Increasing the size of the training set seems to indeed help solving this particular problem. Very common words/articles such as `le` and `la` translated to `the` are especially bad. Model 2 is supposed to help solving this issue, but for sentences that include the same article twice, the algorithm sometimes get confused and do not assign the matching article. Since articles are overwhelmingly present in both languages, the position probabilities might not be sufficient to help the algorithm to make a smart decision.

On the other hand, very rare words are also badly aligned, which basically suggests again that the algorithm needs to see the word in context more time to be able to make a smarter decision. Finally the algorithm seems to be able to make a decent word-to-word translation for common words. For example, both `earn` and `income` were aligned to the French word `revenue` which is a smart decision.

## 2 MT System Training

After taking a close look at the translation in the baseline system and in the reference, I realized how hard the problem was. The reference translations often modify the structure from passive to active. Ex:

```
[c' est le coran qui va unir] [ la place de la personnalit du dictateur]
[la socit compose de tribus opposes] .
[instead of a dictator] , [a society consisting of competing tribes]
[will be united by koran] .
```

## 2.1 My features

### 2.1.1 Pronoun "On"

In French, the pronoun `On` is technically translated to `It` or to a passive form. However, it is usually used when a person is talking about the group as `We`. Therefore, I added an indicator feature fired when both the source has `On` and the translation `We`. This managed to slightly improve a couple of sentence, including this passage:

```
[FR]         finalement je comprends pourquoi on s' efforce d' apprendre ces rgles
[REF]        here i finally understood why we drill all the definitions and rules
[BASELINE ] i understand why efforts to learn these rules
[W/FEATURE] finally i understand why we trying to learn these rules
```

However, this mainly depends on the context. Using `On` like a plural pronoun is not a very high level of language in French and therefore, this feature might help the prediction or make it worth depending on the level of language that is expected. Going forward, this led me to think that it could be very interesting to include a feature of topics for translation engine. This feature would indicate if this is a newspaper article, or a sport magazine, etc. But this goes beyond this exercise.

### 2.1.2 Identical words

One thing that I struggled with in English is to prevent myself from saying words that are the same in French but might not have the same meaning. Therefore I created an indicator feature that would fire the number of identical words in the source and the target if it is greater than 0. I was actually very pleased to see that the weight associated to my feature was positive: `FeatureSameWord:1:0.05860402429379301`. French and English actually share one third of their vocabulary, and I feel like slightly boosting identical words is probably a good idea since these words share the same root. On the other hand, it might actually corrupt some results: `recipient` means `container` in French.

### 2.1.3 Conclusion

I tried other very simple features like checking the number of matching plural words in the original words and in the target words but it decreased my BLEU score. My rule was to look at the `s` at the end of words. But verbs at the same person fall in this category in English. In general, I realized that small hand-crafted feature needed a lot of care, for instance, `children` is plural, and such a rule would lower the score of the translation `enfants -> children` even though it is actually the right one.

Overall, the best BLEU score I obtained was 15.378 which is 0.267 higher than the baseline 15.111. I got it with the two simple features described before.