

JHU Summer School 2006
Lab Session for July 6
Statistical Machine Translation
organized by Philipp Koehn and Chris Callison-Burch

Required Assignment

We discussed in the lecture today IBM Model 1. Implement the EM algorithm for IBM Model 1 in your favorite programming language. Test it on two corpora:

1. the toy corpus `/home/ws06/pkoehn/lab/toy.[ef]`
2. a segment of the Europarl corpus `/home/ws06/pkoehn/lab/europarl.*`
(pick the language pair you are most comfortable with: en=English, fr=French, es=Spanish, de=German)

Your program should output two different things:

1. A table containing the word translation probabilities that were learned
2. The most likely alignment (the Viterbi alignment) for each sentence pair in the training data

Optional Assignments

1. **Evaluate your word alignments** — Chris Callison-Burch has a tool that allows you to view and edit your word alignments. Take a look at <http://demo.linearb.co.uk:8080/sandbox/start.html> for a demonstration. If you put your Viterbi word alignments into the format below, Chris will upload them for you so that you can view the output of your program and make manual corrections to it.

The format of the Viterbi alignments should be the following:

- One sentence pair per line
- A list of alignment points separated by commas
- With alignment point specified as the index of the English word (starting from zero), a period, and the index of the foreign word.

For example `0.0,1.2,2.1` would mean that the English word at position 0 is aligned with the foreign word at position zero, English word at position one is aligned with the foreign word at position two, and the English word at position two is aligned with the French word at position one. Ask Chris if you need help with this.

Once your word alignments are up, take a look at them and see if you can identify what mistakes your program is making.

2. **Implement a hill-climbing method for training** — The algorithm for computing IBM Model alignments uses a trick that allows it to efficiently score the exponential number of possible alignments between words in the sentence. Later IBM Models are not able to exploit this trick, and have to resort to a hill-climbing method to find the highest probability alignment. This method works like this:

- find an initial alignment (e.g. randomly choosing alignment points)
- iterate: if changing an alignment from one English word at position j from $a(j)$ to $a'(j)$ results in a higher probability alignment, apply this change
- stop, if no improvement is possible anymore

Implement this hill-climbing method for IBM Model 1.

3. **Implement a symmetrization heuristic** — Recall that in IBM Model 1 the alignment function $a(j) = i$ maps a (*English*) target position j to a (*foreign*) source position i . It is not possible to map an English word to multiple foreign words, although this may sometimes be desirable. We discussed symmetrization methods such a *growing heuristic*, or using intersection or union of alignment points. Run IBM Model 1 training in both directions (foreign \rightarrow English, English \rightarrow foreign) and implement a symmetrization method that merges the two alignments.