Computational Linguistics

Assignment 5 (2019-12-20)

Winter Semester 2019/20 – Prof. Dr. Alexander Koller

Word alignments

Aligning sentences and words are central tasks in statistical machine translation (SMT). In this assignment, you get to implement a word aligner. Given pairs of aligned sentences in two languages, source and target, the goal is to align source words to their target translations. The resulting alignments might contain unaligned or multiply aligned words, i.e., the word alignments are generally `m:n`, which makes the task challenging.

This assignment is almost identical to the first assignment of a well-known online course on SMT, which is available here: `http://mt-class.org/jhu/hw1.html`. Read this website, and note in particular the link to a tutorial by Adam Lopez on how to implement IBM Model 1. The key points are as follows:

1. Clone the repository from `https://github.com/alopez/en600.468.git`, using Git. Observe that the repository contains some code and a dataset of 100,000 English-French sentence pairs (`hansards.e` and `hansards.f`) in the folder `aligner`. The first 37 sentence pairs are manually aligned, and these manual alignments are encoded in file `hansards.a`.
2. Get to know the code of the aligner (folder `aligner`). It provides a very simple baseline system; test it through the provided command-line interface. Submit the Alignment Error Rate (AER) of the baseline system. Observe that it is terrible.
3. Your task is to improve over the baseline by implementing an aligner based on IBM Model 1 (see *The Challenge* section of the JHU course for a more detailed description). Your program should learn the parameters $P(e|f)$ of Model 1 from the given data, and then use them to compute optimal alignments. Submit the AER for your implementation. Feel free to use NLTK if you find it helpful, but note that anything in the `nltk.align` package is disallowed in this assignment.

4. In addition to what is required in the JHU assignment, experiment also with an off-the-shelf aligner of your choice. GIZA++ used to be the standard, but is now hard to compile. Depending on your preference of programming language, you might try MGIZA, fast_align, or the Berkeley aligner (see the website for links). Compare your IBM Model 1 to the implementation of Model 1 in the off-the-shelf aligner (if available) and another Model $i > 1$ of your choice.

Extra credit: Implement an aligner that improves over your implementation of IBM Model 1. Some ideas are suggested in the JHU homework assignment.

Submissions: Submit your code and document all your evaluation results. Submit at least one alignment visualization from your system in comparison to the baseline system and the off-the-shelf aligner so we can discuss it in class.

Turn in before class on 2020-01-17 on Piazza.