Machine Translation HW6
Inflection

David Russell – drusse19 Adam Poliak – apoliak1

April 19, 2016

1 Building

Our files can be run on the CLSP grid by aggregating the data as described in the assignment into the local folder. They can then be run with `cat data/dtest.lemma | ./inflect-bigram | ./grade`.

2 Baseline

The baseline score we sought to beat was $\boxed{40765/70974 = 0.57}$.

3 Bigrams

Our first step was to implement a bigram model that includes the previous lemma for a given lemma in the training set (that is, the bigram consists of $(lemma_{i-1}, lemma_i)$) so that we use a sequential model when determining our inflections. We train this in addition to the unigram model. Inflections are then determined from the training data based on maximum likelihood estimation (counting) of the observed bigrams. At test-time, if a given bigram has not been observed, we fall back onto the unigram model, and if our model finds no inflection result for a given lemma, we simply use the base lemma we are testing on and provide no inflection. This results in a score of $\boxed{42754/70974 = 0.60}$.

4 Trigrams

We then implemented the same procedure described in the bigram model but including trigrams. Therefore, our trigrams consisted of $(lemma_{i-2}, lemma_{i-1}, lemma_i)$, and we performed MLE again, training on trigrams, bigrams, and unigrams, falling back incrementally during test-time if an observed n-gram had not been observed. This results in a score of $\boxed{42815/70974 = 0.60}$, so we had marginal improvement.

5 POS-Tagging

Next, we implemented a version that included POS tags on our unigram model. By training on each lemma to include its POS tag, we effectively train a bi-gram model of $(POS_i, lemma_i)$, and fall back on the unigram model if this lemma has not appeared with the given part of speech at test-time. This results in a score of $\boxed{42339/70974 = 0.60}$.

6 Viterbi Algorithm

Lastly, in NLP Adam had implemented a version of the Viterbi algorithm with advanced smoothing. In that assignment, the goal was to predict the weather based on the number of ice creams eaten. However, we observed that we can model this problem in a similar way, where we train a hidden markov model with Viterbi using lemmas and parts of speech to predict the inflection. Originally, our fall-back method was to assign <u>every possible</u> inflection to a given $(POS_i, lemma_i)$ combination that didn't exist in our model at test-time. However, we modified this to simply fall back on the lemma as we had in our other models. This greatly improved our computation time (which was otherwise infeasible on only 20 sentences). This results in a final score of $\boxed{49017/70974 = 0.69}$. This implementation process took several hours, it wasn't as if we simply reused code, and we consider it similar to using a machine learning library which is allowed for the 'extra credit' section of other projects, so we felt like we were justified doing this ethically :). The data file can be found at `./scripts/final_sub.txt`, and the script is under the scripts directory.