COSI 134a (Fall 2014) Programming Assignment #3: Hidden Markov Models
Due: November 19 2014, 23:55 EDT

Assignment

Implement likelihood (forward), decoding (Viterbi), and supervised learning (maximum likelihood estimation) algorithms for Hidden Markov Models. Apply them to determine part-of-speech tags. Report on your implementation (design decisions, representations, difficulties) and its performance.

Distribution

The file `pa3-dist1.tgz` contains starter code based on the code distributed for PA1. As before, you may use what you like and ignore the rest. If you find that modifications are necessary to anything outside of the `test_hmm` module, please document your changes. If you're confused by anything, please ask. If you find bugs, please report them as soon as possible.

Basic Tests

Before tackling the part-of-speech task, verify that your algorithms are functioning correctly by reproducing the (corrected—see the comment in `IceCreamHMM.test_likelihood`) probabilities and variable values for the Eisner ice cream example in chapter 6 of Jurafsky & Martin. You may use the distributed test case as a starting point, but will probably need to tweak it to conform to your HMM implementation's interface. The decoding test does not check for specific values of the Viterbi trellis; you are responsible for adding such a test or checking the values manually.

Tagging

Once your HMM implementation is working, you can apply it to a somewhat more challenging task: supervised learning of part-of-speech (POS) tags. By default, you will use tagged sentences from the Penn Treebank (via the NLTK), but any similar corpus (e.g., Brown) should work about equally well. You will need to add (at least) smoothing and unknown-word handling to your HMM implementation in order to complete this task; be sure to document your strategies for both in your report.

There are a range of parameters to experiment with in the supplied POS test case: you can collapse the tag-set space (e.g., by taking first letters only), use different canonicalization strategies for the words (e.g., none, case folding, lemmatization), and adjust the size of the corpus and the train/test split. Play with a few variations and note the results in your report.

Extra Credit

You may also choose to implement unsupervised learning using the forward-backward algorithm (i.e., Baum-Welch). Start with learning parameters for the Eisner ice cream model first, then move on to POS tags. Document your results. *This is not a required part of the assignment.*

Grading

Your grade will be based on the correctness of your HMM algorithm implementation (60%), its performance on the POS tagging task (10%), code clarity and style (20%), and the quality of your report (10%). Successful unsupervised learning is worth an additional 10%.