Ghoti Phrasal Chunker

Algorithm outline

The final version (in `chunk.py`) uses an average perceptron algorithm, with the lazy-update optimization described in Sarkar 2011.

The basic idea is exactly the same as the pseudocode of the baseline provided in the homework description. The only difference is that the final, trained weight vector is the average of all the weight vectors after every step (there are $N * T$ weight vectors in total, where N is the number of sentences and T is the number of iterations, or epochs).

The naive implementation of average perceptron is to maintain a global sum weight vector to accumulate all the weight vectors and calculate the average at the end. The problem is that the total weight vector has a huge number of dimensions and we need to do a lot of vector additions on this gigantic vector. The optimization is to use "lazy" update: many dimensions of the weight vector do not change over many updates. We can thus convert all these additions into adding a multiply of the unchanged value on certain dimension by keeping track of when (at which step) a dimension is changed. This can also be seen as a "sparse" update strategy. See page 38 of Sarkar 2011 for the full pseudocode.

Finally, in order to speed up a lot of experiments we did (including finding the optimal number of epochs), we used `pypy` in place of CPython, which provided really good speedup for free. The optimal number of epochs from our experiments was around 10.

Other experimented approaches

We first tried the baseline approach. And then we implemented the naive average perceptron. It was so slow that we were unable to try any "large" number (say 5) of epochs. Thus, we modified the algorithm a bit such that we only averaged the weight vectors after an entire epoch instead of after each sentence, which actually gave decent improvement as well.

Acknowledgement

- Anoop Sarkar, the course web page
- Michael Collins, *Discriminative Training Methods for Hidden Markov Models: Theory and Experiments with Perceptron Algorithms*. EMNLP 2002.
- Sarkar 2011. Syntax parsing survey.