Faster, Better, or Both! Learning Priority Functions for Decoding

Jiarong Jiang
University of Maryland College Park
jiarong@umiacs.umd.edu

Adam Teichert
Johns Hopkins University
teichert@jhu.edu

Hal Daumé III
University of Maryland College Park
hal@umiacs.umd.edu

Jason Eisner
Johns Hopkins University
jason@cs.jhu.edu

Abstract

Users want natural language processing (NLP) systems to be both fast and accurate, but speed often comes at the cost of quality. The field has been manually exploring various speed-accuracy tradeoffs (for particular problems and datasets). We propose reinforcement learning methods as a way to automate such exploration. As an example, we demonstrate the approach in the context of agenda-based syntactic parsing (Kay, 1986). Using a small number of features we automatically learn weights that achieve competitive parsing accuracies at significant improvements in speed over our baseline.

Decoders (e.g. parsers, taggers, or translation systems) often rely on prioritization and pruning heuristics to speed up the search for good solutions given the input. We wish to replace the manual search for general heuristics with an automatic approach that *learns* heuristics that optimize an arbitrary user-defined measure of quality. Here, we explore quality measures that reflect tradeoffs between speed and accuracy. Our objective measure is **reward** $=$ **accuracy** $+ \alpha \cdot$ **speed**, and we can choose an α that reflects our true preferences, or investigate different values of α.

We formulate the learning of a faster and "better" decoder as a reinforcement learning problem. For our parsing example, the state consists of the full current chart and agenda (so the state space is astronomically large). The agent controls which item (constituent) to pop next from the agenda. Thus, the possible actions correspond to items currently on the agenda. When the agent pops an item y, the environment responds *deterministically* by adding y to the chart, combining y as licensed by the grammar with various adjacent items z that are already in the chart, and placing each resulting new item x back on the agenda. The environment positively rewards the agent for the accuracy of that parse and negatively rewards for the time that it took.

During training, there is an enormous number of different parsing trajectories leading to different parses and which leads to a large variance in our estimate of the gradient of the expected reward. We introduce a δ-optimal policy to explore the space using an "oracle" agenda (an agenda that mimics the work of the regular agenda but only allows items that appear in the gold tree to be enqueued). Whenever we need to choose an action, with probability $1 - \delta$, we allow the parser to explore the space according to its current model, and with probability δ, we force it to take an action from the oracle agenda. If $\delta = 0$, the agent follows standard Boltzmann exploration, and if $\delta = 1$, the learning process becomes similar to learning a classifie at each time step (Daumé III et al., 2009). We allow the exploration rate $1 - \delta$ to increase over time, so finally the model can choose good actions without the help of oracle trees.

We use a simple set of features: the inside score, a Boolean bit for whether the constituent touches the beginning/end of the sentence, the length of the constituent, the ratio of constituent length to sentence length, $\log p$(current constituent label | POS tag for previous/next word), the case pattern of the first word in the span and the case pattern of the word just before/after the span and the punctuation pattern inside the span (Liang et al., 2008). Our experiments show that by using increasing exploration rate ($\delta = 0.8$) and different αs, we achieve a parser that is more than 10 times faster than the HA* parser with only 2-3% accuracy loss.

References

Hal Daumé III, John Langford, and Daniel Marcu. 2009. Search-based structured prediction. *Machine Learning Journal*, 75(3):297–325.

Percy Liang, Hal Daumé III, and Dan Klein. 2008. Structure compilation: Trading structure for features. In *International Conference on Machine Learning (ICML)*, Helsinki, Finland.