Namdar Kabolinejad
260893536

COMP 550 - Reading Assignment 2

The authors focus on demonstrating the potential of unlexicalized PCFG for parsing. They attempt to exploit structural context and utilize linguistically motivated state splits to break down false independence assumptions made in vanilla treebank grammar.
They initially Marovize the rules with v ≤ 2 & h ≤ 2; and proceed by splitting up the grammar, adding additional non-lexical annotations - based on linguistic understanding - and finally run the experiments using the CKY algorithm. The results show that the unlexicalized PCFG is on par with early lexicalized parsers. An advantage of the paper is that it works on the commonly used Penn Treebank dataset which makes the results comparable, however, I felt that the paper was very dense and found myself confused by the multitude of steps taken.

The main difference between lexicalized and unlexicalized parsing is in content versus function words. Function words are usually markers for selection, while content words are essentially a proxy for semantics. This means that we should differentiate between annotating nodes that have a different functional head.
In unlexicalized parsers, grammar rules aren't systematically specified to the level of lexical items. In this approach, we are trying to leverage the linguistic structure of the phrases and not the past information. For example, in unlexicalized parsing, PP[in] and PP[of] are different, but annotating using content words isn't allowed, like NP[stocks] vs NP[bonds]. One disadvantage of the paper is that some methods used look very much like lexicalization, for example, the SPLIT-VP tag annotates all VP nodes with their head tag, although the effect of this tag might be very small I think it would be better if the authors didn't include it.

I don't believe that the CYK algorithm needs to be modified to work with the techniques mentioned in the paper. This is because Markovization and annotation are operations that transform and add to the grammar rules. These steps can be applied when converting the grammar rules to CNF; then the CYK algorithm can be simply used with a new set of rules. However, this is not to say that the running time and the accuracy of the CYK algorithm doesn't change after applying the steps. For one thing, as we Markovize and annotate we are adding to the grammar rules, given that a main part of the CYK algorithm is to loop over the grammar rules, I'd expect the running time to increase as the grammar size grows. Therefore, not all splits are guaranteed to be beneficial at the end, and it's important to keep the grammar size manageable and find a compromise between the improved performance gained from annotating the data and grammar complexity.

We know that NNs can detect complex nonlinear relationships and possible interactions between predictor variables. Given these abilities, I believe NNs could be useful in extracting grammar rules and combating sparsity and could somehow replace the steps of Markovization and annotation. We know that the strong independent assumption made by traditional methods results in undesirable biases. In NNs these assumptions could be much weaker, thus providing a potential alternative to using smoothing techniques to find estimates for infrequent events. Furthermore, I believe the same concept could be extended to tag-splitting, so NNs could learn to uncover splits that might not be obvious from a linguistic approach the authors took in the paper.
A clear advantage of using NNs is that they can work well with sparse data and potentially improve the parsing result, the disadvantage is that translating the problem to fit the NNs might be hard, and given the fact that we have to use large data sets in linguistics, even with NNs, we will need high computing power. And it's not clear how much of an improvement NNs would be to traditional methods, for example, a simple Markovization.