Syntactic Parsing - Assignment 3

Xiajing Li

---

1 Causes of the discrepancies

Comparing the output file and the gold file, it shows most of the errors are wrongly linked to the root as its head. Based on the algorithm, only the words that are not linked with any arc would be linked with ROOT. Usually there should be only one word linked with ROOT in a sentence.

Missing token
Example: *I was able to secure a Duns # and Fed Tax ID however, I am lacking Fax and bank account numbers.*
The original conll format, # is a symbol used for comment such as sentence id and text so that the function *read_sentence* skip lines starting with this symbol and thus causes not only missing token in the sentence, but also word after this token facing index disorder. This problem would not happen when using normal conll format dataset since all lines of tokens are starting with the corresponding word index.

Non-projective sentences
Example: *You've all won!*
Non-projective sentence is the main problem of affecting the parsing accuracy since our transition-based parsing algorithm could only handle projective sentences. Those non-projective sentences have crossed-arcs in dependency trees which means some words has higher projective order than the words before it. But our parser cannot produce crossed arc because the head might be deleted in previous transition or discontinuity cannot be fixed. Therefore those words could only be linked to ROOT.

2 Possible solutions

Non-projective sentences
Handling non-projective sentences parsing is a hard and time-consuming task in transition-based dependency parsing. Basically there are two kinds of ideas in solving this problem: change sentence into projective order or change the arc of non-projective word. Current solutions of those two ideas including (1)adding SWAP operator and swap the word from stack to buffer when facing non-projectivity;(2)implementing pseduo-projective parsing.

By implementing SWAP operator based on arc-standard algorithm, when the current configuration indicates arcs between discontinued words, words between the head and the dependency would be swapped back to buffer so that discontinuity could be fixed. It change the output result.

By implementing pseduo-projective parsing, non-projective trees in gold-standard would be transformed into projective one. Dependencies with crossed arcs will be linked to the head of its original head.

3 VG:arc-standard

For the VG part, I implemented arc-standard transition system. The difference between two systems is the rules of transition and adding arcs because arc-standard system focus on two top words in stack. Beside, arc-standard system contains only three types of transition, comparing to four in arc-eager. Without "reduce" action, the parser would have less chance to connect words that are not close to each other. The results show that arc-standard transition system(78.29%) performs worse than arc-eager(99.64%) in this case because of far more missing relations.