Suggested Readings

Shubham Gupta

January 20, 2020

1 Incrementality in Deterministic Dependency Parsing

- Why incrementality?
  - Practical and can be used in real time applications
  - Theoritical i.e human parsing is largely incremental, so it makes sense to try it out in the text as well.
  - **Core aim**: Build a complete syntactic analysis for the input string
  - Dependencies can be done by labelling each word in the sentence by POS tags and the arcs between each of these words with their gramatical function
  - Parser configs represented by triples (S, I, A). S = Stack, I = List of input tokets, A = current arc relation in dependency graph
  - **Incremental**: For bottom up parsers, idea is to have a term related to graph at all times. Number of elements in stack is not greater than 2
  - **Arc eager parsing**: Do both bottom-up and top-down parsing. Left dependents bottom-up and right dependents top-down
  - In standard algorithm, Right-redice is bottom-up approach i.e it is a Right-Arc + Reduce function.
  - **Incremental for arc parsing**: Enforce that the graph should be connected at all times, since two tokens need not be present on the stack to create the dependency
  - *Top Down parsing*: Head is attached to dependant before dependent is attached to its other dependents
  - *Bottom up parsing*: Dependent is attached to its head before head is attached to its parent's head.
  - **Performance**: Evaluation based on arc-eager parsing with memory-based classifier for predicting next transition
  - 90% of all confihs ave no more than three connected components on the stack.
  - Sentences that violate the incremental parsing property are because:
    - Can't be parse into a well formed tree i.e single projective dependency tree

2 Conclusion

- This paper shows that strict incremental parsing is not acheivble with the framework presented.
- HOWEVER, arc-eager parsing algorithm is optimial for incremental parsing.