Combining (Second-Order) Graph-Based and Headed-Span-Based Projective Dependency Parsing

Songlin Yang, Kewei Tu
April 20, 2022
SIST, ShanghaiTech University
Dependency parsing

- First-order graph-based parsers cannot capture sufficient subtree information.
- Headed-span-based parsers capture subtree information in headed spans but ignore arc scores.
- We want to incorporate both arc and headed span scores into parsing.

Figure 1: An example projective dependency parse tree with all its headed spans.
Direct modeling with $O(n^4)$ modified Eisner-Satta algorithm

- Our modification is colored in red.
- An “unfinished” span can absorb a child span to form a larger span, while a “finished” span (double underlined) cannot expand anymore and thus corresponds to a headed-span for the given headword.
$O(n^3)$ modified (second-order) Eisner algorithm

- Apply the head-splitting trick: $s^\text{span}_{i, r_i, i} = s^\text{left}_{i, r_i} + s^\text{right}_{i, r_i}$
## Experiments on PTB and CTB

| Model Configuration | PTB (UAS) | PTB (LAS) | CTB (UAS) | CTB (LAS) |
|---------------------|-----------|-----------|-----------|-----------|
|                     | +BERT$_{\text{large}}$ | +BERT$_{\text{base}}$ |           |           |
| Biaffine+MM$^\dagger$ | 97.22     | 95.71     | 93.18     | 92.10     |
| Span                | 97.24     | 95.73     | 93.33     | 92.30     |
| 1O+Span             | 97.26     | 95.68     | 93.56     | 92.49     |
| 1O+Span+HeadSplit   | 97.30     | 95.77     | 93.46     | 92.42     |
| Biaffine+2O+MM      | 97.28     | 95.73     | 93.42     | 92.34     |
| 2O+Span+HeadSplit   | 97.23     | 95.69     | 93.57     | 92.47     |

For reference

| Model Configuration | PTB (UAS) | PTB (LAS) | CTB (UAS) | CTB (LAS) |
|---------------------|-----------|-----------|-----------|-----------|
| MFVI2O              | 96.91     | 95.34     | 92.55     | 91.69     |
| HierPtr             | 97.01     | 95.48     | 92.65     | 91.47     |
|                      | +XLNet$_{\text{large}}$ | +BERT$_{\text{base}}$ |           |           |
| HPSG$^\flat$        | 97.20     | 95.72     | -         | -         |
| HPSG+LAL$^\flat$    | 97.42     | 96.26     | 94.56     | 89.28     |
### Experiments on UD

|                       | bg  | ca  | cs  | de  | en  | es  | fr  | it  | nl  | no  | ro  | ru  | Avg |
|-----------------------|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| **+ BERT<sub>multilingual</sub>** |     |     |     |     |     |     |     |     |     |     |     |     |     |
| Biaffine+MM↑          | 90.30 | 94.49 | 92.65 | **85.98** | 91.13 | 93.78 | 91.77 | 94.72 | 91.04 | 94.21 | 87.24 | 94.53 | 91.82 |
| Span                  | 91.10 | 94.46 | 92.57 | 85.87 | **91.32** | 93.84 | 91.69 | 94.78 | 91.65 | 94.28 | 87.48 | 94.45 | 91.96 |
| 1O+Span               | 91.44 | 94.54 | 92.68 | 85.75 | 91.23 | 93.84 | 91.67 | **94.97** | **91.81** | 94.35 | 87.17 | 94.49 | 91.99 |
| 1O+Span+Headsplit     | 91.46 | 94.53 | 92.63 | 85.78 | 91.25 | 93.77 | **91.91** | 94.88 | 91.59 | 94.18 | 87.45 | 94.47 | **92.00** |
| Biaffine+2O+MM        | 91.58 | 94.48 | **92.69** | 85.72 | 91.28 | 93.80 | 91.89 | 94.88 | 91.30 | 94.23 | 87.55 | **94.55** | **92.00** |
| 2O+Span+Headsplit     | **91.82** | **94.58** | 92.59 | 85.65 | 91.28 | **93.86** | 91.80 | 94.75 | 91.50 | **94.40** | **87.71** | 94.51 | **92.04** |
| **For reference**      |     |     |     |     |     |     |     |     |     |     |     |     |     |
| MFVI2O                | 91.30 | 93.60 | 92.09 | 82.00 | 90.75 | 92.62 | 89.32 | 93.66 | 91.21 | 91.74 | 86.40 | 92.61 | **90.61** |
Summary of experiments

- 😊 Second-order parsing is still useful in the deep learning age.
  - What about third-order parsing?
- 😊 1O+Span and 1O+Span+Headsplit outperform Span and 1O
  - Combining headed-span-based and first-order graph-based methods is useful.
- 😊 1O+Span and 1O+Span+Headsplit have very similar performance.
  - It is more advantageous to apply the head-splitting trick to enjoy a lower parsing complexity.
- 😞 2O+Span+Headsplit has no clear advantage over 2O
  - Maybe the utility of adjacent sibling information and headed span information is overlapping.
Questions?