Constituency Parsing

李宏毅 Hung-yi Lee
<table>
<thead>
<tr>
<th></th>
<th>One Sequence</th>
<th>Multiple Sequences</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>One Class</strong></td>
<td>Sentiment Classification, Stance Detection, Veracity Prediction, Intent Classification, Dialogue Policy</td>
<td>NLI, Search Engine Relation Extraction</td>
</tr>
<tr>
<td><strong>Class for each Token</strong></td>
<td>POS tagging, Word segmentation, Extractive Summarization, Slotting Filling, NER</td>
<td></td>
</tr>
<tr>
<td><strong>Copy from Input</strong></td>
<td></td>
<td>Extractive QA</td>
</tr>
<tr>
<td><strong>General Sequence</strong></td>
<td>Abstractive Summarization, Translation, Grammar Correction, NLG</td>
<td>General QA, Chatbot, State Tracker, Task Oriented Dialogue</td>
</tr>
<tr>
<td><strong>Other?</strong></td>
<td>Parsing, Coreference Resolution</td>
<td></td>
</tr>
</tbody>
</table>
Constituency Parsing

• Some text spans are constituents ("units")
• Each constituent has a label.
Constituency Parsing - Labels

<table>
<thead>
<tr>
<th>Label</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADJP</td>
<td>Adjective phrase</td>
</tr>
<tr>
<td>ADVP</td>
<td>Adverb phrase</td>
</tr>
<tr>
<td>NP</td>
<td>Noun phrase</td>
</tr>
<tr>
<td>PP</td>
<td>Prepositional phrase</td>
</tr>
<tr>
<td>S</td>
<td>Simple declarative clause</td>
</tr>
<tr>
<td>SBAR</td>
<td>Subordinate clause</td>
</tr>
<tr>
<td>SBARQ</td>
<td>Direct question introduced by wh-element</td>
</tr>
<tr>
<td>SINV</td>
<td>Declarative sentence with subject-aux inversion</td>
</tr>
<tr>
<td>SQ</td>
<td>Yes/no questions and subconstituent of SBARQ excluding wh-element</td>
</tr>
<tr>
<td>VP</td>
<td>Verb phrase</td>
</tr>
<tr>
<td>WHADVP</td>
<td>Wh-adverb phrase</td>
</tr>
<tr>
<td>WHNP</td>
<td>Wh-noun phrase</td>
</tr>
<tr>
<td>WHPP</td>
<td>Wh-prepositional phrase</td>
</tr>
<tr>
<td>X</td>
<td>Constituent of unknown or uncertain category</td>
</tr>
<tr>
<td>*</td>
<td>“Understood” subject of infinitive or imperative</td>
</tr>
<tr>
<td>0</td>
<td>Zero variant of that in subordinate clauses</td>
</tr>
<tr>
<td>T</td>
<td>Trace of wh-constituent</td>
</tr>
</tbody>
</table>

+ All POS tags
Constituency Parsing

- Each word is a constituent (their labels are POS tags)
- Some consecutive constituents can form a larger one.
Constituency Parsing

Each constituent is a node.

Form a tree

- Each word is a constituent (their labels are POS tags)
- Some consecutive constituents can form a larger one.
Chart-based Approach

CKY chart parsing

Figure 13.4 Completed parse table for Book the flight through Houston.

Chart-based

- **NP**: deep, learning
- **VP**: is, very, powerful
- **S**: Constituent?
  - **binary classification**
  - **multi-class classification**
- **Classifier**: Which label?
  - span: $w_2, w_3, w_4, w_5, \ldots$
Chart-based

- **Constituent?**
  - **Which label?**
  - **Classifier**
  - **deep learning**

- **Constituent?**
  - **Which label?**
  - **Classifier**
  - **is very powerful**

- **S**
- **VP**
- **ADJV**

**YES**
- **Classifier**
- **deep learning**
Chart-based

Chart-based deep learning is very powerful

NO
Constituent?

Don’t Care
Which label?

NO
Constituent?

Don’t Care
Which label?

Classifier

Classifier

deep learning is very powerful
Chart-based – Classifier

Yes/No

Constituent?

Span Feature Extraction

Which Label?

Label

Pre-trained Model

ELMO, BERT ...

Pre-trained Model

w₁

w₂

w₃

w₄

w₅

w₆

w₇

w₈

w₉

w₁₀
Chart-based

• Given a sequence with N tokens, then run the classifier N(N-1)/2 times ……

Contradiction!
Inference

Enumerate all possible trees, and use the classifier to give scores. where you need CKY algorithm

[Stern, et al., ACL’17]
Transition-based

<table>
<thead>
<tr>
<th>Stack</th>
<th>Buffer</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td></td>
<td>NT(S)</td>
</tr>
<tr>
<td>1</td>
<td>(S)</td>
<td>NT(NP)</td>
</tr>
<tr>
<td>2</td>
<td>(S</td>
<td>(NP</td>
</tr>
<tr>
<td>3</td>
<td>(NP</td>
<td>The</td>
</tr>
<tr>
<td>4</td>
<td>(S</td>
<td>(NP</td>
</tr>
<tr>
<td>5</td>
<td>(S</td>
<td>(NP</td>
</tr>
<tr>
<td>6</td>
<td>(S</td>
<td>(NP The hungry cat)</td>
</tr>
<tr>
<td>7</td>
<td>(S</td>
<td>(NP The hungry cat)</td>
</tr>
<tr>
<td>8</td>
<td>(S</td>
<td>(NP The hungry cat)</td>
</tr>
<tr>
<td>9</td>
<td>(S</td>
<td>(NP The hungry cat)</td>
</tr>
<tr>
<td>10</td>
<td>(S</td>
<td>(NP The hungry cat)</td>
</tr>
<tr>
<td>11</td>
<td>(S</td>
<td>(NP The hungry cat)</td>
</tr>
</tbody>
</table>
Transition-based [Dyer, et al., NAACL’16]

Stack (empty at the beginning)

Buffer deep learning is very powerful

Actions

CREATE(X)
Create a constitute X
(X = NP, VP ...)

SHIFT
More a token from buffer to stack

REDUCE
Close a constitute
Transition-based

(empty at the beginning)

CREATE(S)

deep  learning  is  very  powerful
Transition-based deep learning is very powerful.
RNN Grammar

CREATE(X)  SHIFT  REDUCE

RNN  Network  RNN

Stack  Previous actions  Buffer
RNN Grammar – Training

- typical classification task
- RL is not needed

Ground truth
# Grammar as a Foreign Language

<table>
<thead>
<tr>
<th>Oriol Vinyals*</th>
<th>Lukasz Kaiser*</th>
</tr>
</thead>
<tbody>
<tr>
<td>Google</td>
<td>Google</td>
</tr>
<tr>
<td><a href="mailto:vinyals@google.com">vinyals@google.com</a></td>
<td><a href="mailto:lukaszkaiser@google.com">lukaszkaiser@google.com</a></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Terry Koo</th>
<th>Slav Petrov</th>
<th>Ilya Sutskever</th>
</tr>
</thead>
<tbody>
<tr>
<td>Google</td>
<td>Google</td>
<td>Google</td>
</tr>
<tr>
<td><a href="mailto:terrykoo@google.com">terrykoo@google.com</a></td>
<td><a href="mailto:slav@google.com">slav@google.com</a></td>
<td><a href="mailto:ilyasu@google.com">ilyasu@google.com</a></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Geoffrey Hinton</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Google</td>
<td></td>
</tr>
<tr>
<td><a href="mailto:geoffhinton@google.com">geoffhinton@google.com</a></td>
<td></td>
</tr>
</tbody>
</table>

[Vinyals, et al., NIPS’15]
Deep learning is very powerful.

Of course, you can try other tree traversal approaches [Liu, et al., TACL’17].

Tree to Sequence

(S (NP deep learning) (VP is (ADJV very powerful))

Seq2seq!
Seq2seq v.s. RNN grammar

(S (NP deep learning) (VP is (ADJV very powerful)))

[Vinyals, et al., NIPS’15]

CREATE(S) CREATE(NP) SHIFT SHIFT REDUCE

CREATE(VP) SHIFT CREATE(ADJV) SHIFT SHIFT

REDUCE REDUCE REDUCE

[Dyer, et al., NAACL’16]
Unsupervised Parsing?

Can we find parsing trees without label data?

YES!

Reference: https://youtu.be/YIuBHB9Ejok
Reference

- [Stern, et al., ACL’17] Mitchell Stern, Jacob Andreas, Dan Klein, A Minimal Span-Based Neural Constituency Parser, ACL, 2017