COMP 790.139 (Fall 2017)
Natural Language Processing

Lecture 4: Syntactic Parsing (Constituent, Dependency, CCG, etc.)

Mohit Bansal

(various slides adapted/borrowed from courses by Dan Klein, Richard Socher, Chris Manning, JurafskyMartin-SLP3, others)
Announcements

- Chapter section summary due Sunday Sep24 midnight

- Coding-HW1 (on word vector training+evaluation +visualization) will be release in 1-2 days – TA Yixin will give overview of the homework today!

- TA Yixin Nie’s office hours: 2.30-3.30pm Wednesdays (SN-372; might move to 2nd floor reading room)
Coding HW1 (TA Yixin Nie’s presentation)
Syntactic Parsing
Constituent Parsing
Syntactic Parsing -- Constituent

- Phrase-structure parsing or Bracketing

![Syntax Tree]

- Demos: [http://tomato.banatao.berkeley.edu:8080/parser/parser.html](http://tomato.banatao.berkeley.edu:8080/parser/parser.html)
Probabilistic Context-free Grammars

A context-free grammar is a tuple \(<N, T, S, R>\)

- **\(N\)**: the set of non-terminals
  - Phrasal categories: S, NP, VP, ADJP, etc.
  - Parts-of-speech (pre-terminals): NN, JJ, DT, VB

- **\(T\)**: the set of terminals (the words)

- **\(S\)**: the start symbol
  - Often written as ROOT or TOP
  - *Not* usually the sentence non-terminal **S**

- **\(R\)**: the set of rules
  - Of the form \(X \rightarrow Y_1 Y_2 \ldots Y_k\), with \(X, Y_i \in N\)
  - Examples: \(S \rightarrow NP \ \text{VP}\), \(\text{VP} \rightarrow \text{VP} \ \text{CC} \ \text{VP}\)
  - Also called rewrites, productions, or local trees
A PCFG:

- Adds a top-down production probability per rule \( P(Y_1 Y_2 \ldots Y_k \mid X) \)
- Allows us to find the ‘most probable parse’ for a sentence
- The probability of a parse is just the product of the probabilities of the individual rules
Treebank PCFG

- Can just count the frequency of each rule and normalize (but not very effective)

<table>
<thead>
<tr>
<th>Model</th>
<th>F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline</td>
<td>72.0</td>
</tr>
</tbody>
</table>

[Charniak, 1996]
Real Treebank Examples

Long, complex sentences with several clauses, nested prepositions, etc.

( (S (NP-SBJ The move)
   (VP followed
     (NP (NP a round)
       (PP of
         (NP (NP similar increases)
           (PP by
             (NP other lenders))
           (PP against
             (NP Arizona real estate loans))))))
  ,
  (S-ADV (NP-SBJ *)
   (VP reflecting
     (NP (NP a continuing decline)
       (PP-LOC in
         (NP that market))))))
  ).)
Grammar Refinement

- Conditional independence assumptions often too strong! Not every NP expansion can fill every NP slot

- Better results by enriching the grammar e.g.,
  - Lexicalization [Collins, 1999; Charniak, 2000]
Grammar Refinement

- Conditional independence assumptions often too strong! Not every NP expansion can fill every NP slot

- Better results by enriching the grammar e.g.,
  - Lexicalization [Collins, 1999; Charniak, 2000]
Grammar Refinement

Conditional independence assumptions often too strong! Not every NP expansion can fill every NP slot

Better results by enriching the grammar e.g.,

- Lexicalization [Collins, 1999; Charniak, 2000]
- Latent Tag-splitting [Matsuzaki et al., 2005; Petrov et al., 2006]
CKY (or CYK) Parsing Algorithm (Bottom-up)

bestScore(s)
   for (i : [0,n-1])
      for (X : tags[s[i]])
         score[X][i][i+1] = tagScore(X,s[i])
   for (diff : [2,n])
      for (i : [0,n-diff])
         j = i + diff
         for (X->YZ : rule)
            for (k : [i+1, j-1])
               score[X][i][j] = max{score[X][i][j], score(X->YZ) * score[Y][i][k] * score[Z][k][j]}

[Cocke, 1970; Kasami, 1965; Younger, 1967]
function CKY-PARSE(words, grammar) returns table

for j ← from 1 to LENGTH(words) do
  for all \{A | A \to words[j] \in grammar\}  
    table[j-1, j] ← table[j-1, j] \cup A  
  for i ← from j - 2 downto 0 do
    for k ← i + 1 to j - 1 do
      for all \{A | A \to BC \in grammar\ and B \in table[i, k] and C \in table[k, j]\}  
        table[i, j] ← table[i, j] \cup A
Latent Variable Grammars

Parse Tree $T$
Sentence $w$

Derivations $t : T$

Parameters $\theta$

Grammar $G$

<table>
<thead>
<tr>
<th>Rule</th>
<th>Question</th>
</tr>
</thead>
<tbody>
<tr>
<td>$S_0 \rightarrow NP_0\ VP_0$</td>
<td>?</td>
</tr>
<tr>
<td>$S_0 \rightarrow NP_1\ VP_0$</td>
<td>?</td>
</tr>
<tr>
<td>$S_0 \rightarrow NP_0\ VP_1$</td>
<td>?</td>
</tr>
<tr>
<td>$S_0 \rightarrow NP_1\ VP_1$</td>
<td>?</td>
</tr>
<tr>
<td>$S_1 \rightarrow NP_0\ VP_0$</td>
<td>?</td>
</tr>
<tr>
<td>$\ldots$</td>
<td></td>
</tr>
<tr>
<td>$S_1 \rightarrow NP_1\ VP_1$</td>
<td>?</td>
</tr>
<tr>
<td>$\ldots$</td>
<td></td>
</tr>
<tr>
<td>$NP_0 \rightarrow PRP_0$</td>
<td>?</td>
</tr>
<tr>
<td>$NP_0 \rightarrow PRP_1$</td>
<td>?</td>
</tr>
<tr>
<td>$\ldots$</td>
<td></td>
</tr>
</tbody>
</table>

Lexicon

<table>
<thead>
<tr>
<th>Rule</th>
<th>Question</th>
</tr>
</thead>
<tbody>
<tr>
<td>$PRP_0 \rightarrow She$</td>
<td>?</td>
</tr>
<tr>
<td>$PRP_1 \rightarrow She$</td>
<td>?</td>
</tr>
<tr>
<td>$\ldots$</td>
<td></td>
</tr>
<tr>
<td>$VBD_0 \rightarrow was$</td>
<td>?</td>
</tr>
<tr>
<td>$VBD_1 \rightarrow was$</td>
<td>?</td>
</tr>
<tr>
<td>$VBD_2 \rightarrow was$</td>
<td>?</td>
</tr>
<tr>
<td>$\ldots$</td>
<td></td>
</tr>
</tbody>
</table>

[ Petrov et al., 2006 ]
Learning Latent Splits (Inside-Outside)

- Forward-backward (last week) but for trees

EM algorithm:
- Brackets are known
- Base categories are known
- Only induce subcategories

He was right.

Brackets are known
Base categories are known
Only induce subcategories

Forward-backward (last week) but for trees

[ Petrov et al., 2006 ]
DT Tag Splits Example

DT

\[
\begin{array}{c}
\text{the} (0.50) \\
\text{a} (0.24) \\
\text{The} (0.08)
\end{array}
\]

\[
\begin{array}{c}
\text{a} (0.61) \\
\text{the} (0.19) \\
\text{an} (0.11)
\end{array}
\]

\[
\begin{array}{c}
\text{the} (0.80) \\
\text{The} (0.15) \\
\text{a} (0.01)
\end{array}
\]

\[
\begin{array}{c}
\text{this} (0.39) \\
\text{that} (0.28) \\
\text{That} (0.11)
\end{array}
\]

\[
\begin{array}{c}
\text{some} (0.20) \\
\text{all} (0.19) \\
\text{those} (0.12)
\end{array}
\]

DT-1  DT-2  DT-3  DT-4

[ Petrov et al., 2006 ]
### Other Learned Splits

#### Proper Nouns (NNP):

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>NNP-12</td>
<td>John</td>
<td>Robert</td>
<td>James</td>
</tr>
<tr>
<td>NNP-2</td>
<td>J.</td>
<td>E.</td>
<td>L.</td>
</tr>
<tr>
<td>NNP-1</td>
<td>Bush</td>
<td>Noriega</td>
<td>Peters</td>
</tr>
<tr>
<td>NNP-15</td>
<td>New</td>
<td>San</td>
<td>Wall</td>
</tr>
<tr>
<td>NNP-3</td>
<td>York</td>
<td>Francisco</td>
<td>Street</td>
</tr>
</tbody>
</table>

#### Personal pronouns (PRP):

<table>
<thead>
<tr>
<th>PRP-0</th>
<th>it</th>
<th>He</th>
<th>I</th>
</tr>
</thead>
<tbody>
<tr>
<td>PRP-1</td>
<td>it</td>
<td>he</td>
<td>they</td>
</tr>
<tr>
<td>PRP-2</td>
<td>it</td>
<td>them</td>
<td>him</td>
</tr>
</tbody>
</table>

[ Petrov et al., 2006 ]
Other Learned Splits

- Relative adverbs (RBR):

<table>
<thead>
<tr>
<th>RBR-0</th>
<th>further</th>
<th>lower</th>
<th>higher</th>
</tr>
</thead>
<tbody>
<tr>
<td>RBR-1</td>
<td>more</td>
<td>less</td>
<td>More</td>
</tr>
<tr>
<td>RBR-2</td>
<td>earlier</td>
<td>Earlier</td>
<td>later</td>
</tr>
</tbody>
</table>

- Cardinal Numbers (CD):

<table>
<thead>
<tr>
<th>CD-7</th>
<th>one</th>
<th>two</th>
<th>Three</th>
</tr>
</thead>
<tbody>
<tr>
<td>CD-4</td>
<td>1989</td>
<td>1990</td>
<td>1988</td>
</tr>
<tr>
<td>CD-11</td>
<td>million</td>
<td>billion</td>
<td>trillion</td>
</tr>
<tr>
<td>CD-0</td>
<td>1</td>
<td>50</td>
<td>100</td>
</tr>
<tr>
<td>CD-3</td>
<td>1</td>
<td>30</td>
<td>31</td>
</tr>
<tr>
<td>CD-9</td>
<td>78</td>
<td>58</td>
<td>34</td>
</tr>
</tbody>
</table>

[Petrov et al., 2006]
## Latent PCFG Results

<table>
<thead>
<tr>
<th></th>
<th>≤ 40 words F1</th>
<th>all F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>ENG</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Charniak&amp;Johnson ‘05 (generative)</td>
<td>90.1</td>
<td>89.6</td>
</tr>
<tr>
<td>Split / Merge</td>
<td>90.6</td>
<td>90.1</td>
</tr>
<tr>
<td>GER</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Dubey ‘05</td>
<td>76.3</td>
<td>-</td>
</tr>
<tr>
<td>Split / Merge</td>
<td>80.8</td>
<td>80.1</td>
</tr>
<tr>
<td>CHN</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Chiang et al. ‘02</td>
<td>80.0</td>
<td>76.6</td>
</tr>
<tr>
<td>Split / Merge</td>
<td>86.3</td>
<td>83.4</td>
</tr>
</tbody>
</table>

[Source: Petrov et al., 2006]
Evaluating Constituent Parsers

\[
\text{labeled recall: } = \frac{\text{# of correct constituents in hypothesis parse of } s}{\text{# of correct constituents in reference parse of } s}
\]

\[
\text{labeled precision: } = \frac{\text{# of correct constituents in hypothesis parse of } s}{\text{# of total constituents in hypothesis parse of } s}
\]

\[
F_\beta = \frac{(\beta^2 + 1)PR}{\beta^2 P + R}
\]

\[
F_1 = \frac{2PR}{P + R}
\]

cross-brackets: the number of constituents for which the reference parse has a bracketing such as ((A B) C) but the hypothesis parse has a bracketing such as (A (B C)).
Other Results

- **Collins, 1999** → 88.6 F1 (generative lexical)
- **Charniak and Johnson, 2005** → 89.7 / 91.3 F1 (generative lexical / reranking)
- **Petrov et al., 2006** → 90.7 F1 (generative unlexical)
- **McClosky et al., 2006** – 92.1 F1 (generative + reranking + self-training)
Syntactic Ambiguities

- I saw the old man with a telescope
- I shot an elephant in my pajamas
- I cleaned the dishes in my pajamas
- I cleaned the dishes in the sink
The board approved [its acquisition] [by Royal Trustco Ltd.] [of Toronto] [for $27 a share] [at its monthly meeting].
Attachment Ambiguity Types

- **Prepositional phrases:**
  *They cooked the beans in the pot on the stove with handles.*

- **Particle vs. preposition:**
  *The puppy tore up the staircase.*

- **Complement structures**
  *The tourists objected to the guide that they couldn’t hear.*
  *She knows you like the back of her hand.*

- **Gerund vs. participial adjective**
  *Visiting relatives can be boring.*
  *Changing schedules frequently confused passengers.*
Attachment Ambiguity Types

- Modifier scope within NPs
  - *impractical design requirements*
  - *plastic cup holder*

- Multiple gap constructions
  - *The chicken is ready to eat.*
  - *The contractors are rich enough to sue.*

- Coordination scope:
  - *Small rats and mice can squeeze into holes or cracks in the wall.*
World Knowledge is Important

Clean the dishes in the sink.
Web Features for Syntactic Parsing

Dependency:

They considered running the ad during the Super Bowl.

Constituent:

[Nakov and Hearst 2005; Pitler et al., 2010; Bansal and Klein, 2011]
They considered running the ad during the Super Bowl.

Web Ngrams

\[
\text{count}((\text{running it during})) > \text{count}((\text{considered it during}))
\]

- 7-10% relative error reduction over 90-92% parsers

[Bansal and Klein, 2011]
Visual Recognition Cues

- Joint parsing and image recognition

*the mug on the table with a crack*
Visual Recognition Cues

- Joint parsing and image recognition

- the mug on the table with a crack

- red chair and table

- light green table
We present an approach to simultaneously perform semantic segmentation and prepositional phrase attachment resolution for captioned images. Some ambiguities in language cannot be resolved without simultaneously reasoning about an associated image. If we consider the sentence "I shot an elephant in my pajamas", looking at language alone (and not using common sense), it is unclear if it is the person or the elephant wearing the pajamas or both. Our approach produces a diverse set of plausible hypotheses for both semantic segmentation and prepositional phrase attachment resolution that are then jointly reranked to select the most consistent pair. We show that our semantic segmentation and prepositional phrase attachment resolution modules have complementary strengths, and that joint reasoning produces more accurate results than any module operating in isolation. Multiple hypotheses are also shown to be crucial to improved multiple-module reasoning. Our vision and language approach significantly outperforms the Stanford Parser (De Marneffe et al., 2006) by 17.91% (28.69% relative) and 12.83% (25.28% relative) in two different experiments. We also make small improvements over DeepLab-CRF (Chen et al., 2015).
Dependency Parsing
Dependency Parsing

- Predicting directed head-modifier relationship pairs

Can convert (lexicalized) constituent tree to dependency tree (each local binary tree gives us a dependency attachment from head to modifier)

Constituent Parse (with head words)  Dependency Parse
**Dependency Parsing**

<table>
<thead>
<tr>
<th>Clausal Argument Relations</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NSUBJ</td>
<td>Nominal subject</td>
</tr>
<tr>
<td>DOBJ</td>
<td>Direct object</td>
</tr>
<tr>
<td>IOBJ</td>
<td>Indirect object</td>
</tr>
<tr>
<td>CCOMP</td>
<td>Clausal complement</td>
</tr>
<tr>
<td>XCOMP</td>
<td>Open clausal complements</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Nominal Modifier Relations</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NMOD</td>
<td>Nominal modifier</td>
</tr>
<tr>
<td>AMOD</td>
<td>Adjectival modifier</td>
</tr>
<tr>
<td>NUMMOD</td>
<td>Numeric modifier</td>
</tr>
<tr>
<td>APPOS</td>
<td>Appositional modifier</td>
</tr>
<tr>
<td>DET</td>
<td>Determiner</td>
</tr>
<tr>
<td>CASE</td>
<td>Prepositions, postpositions and other case markers</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Other Notable Relations</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CONJ</td>
<td>Conjunct</td>
</tr>
<tr>
<td>CC</td>
<td>Coordinating conjunction</td>
</tr>
</tbody>
</table>

Figure 14.2 Selected dependency relations from the Universal Dependency set. (de Marneffe et al., 2014)
### Dependency Parsing

<table>
<thead>
<tr>
<th>Relation</th>
<th>Examples with <em>head</em> and <em>dependent</em></th>
</tr>
</thead>
<tbody>
<tr>
<td>NSUBJ</td>
<td>United <em>canceled</em> the flight.</td>
</tr>
<tr>
<td>DOBJ</td>
<td>United <em>diverted</em> the <em>flight</em> to Reno.</td>
</tr>
<tr>
<td></td>
<td>We <em>booked</em> her the first <em>flight</em> to Miami.</td>
</tr>
<tr>
<td>IOBJ</td>
<td>We <em>booked</em> her the flight to Miami.</td>
</tr>
<tr>
<td>NMOD</td>
<td>We took the <em>morning</em> <em>flight</em>.</td>
</tr>
<tr>
<td>AMOD</td>
<td>Book the <em>cheapest</em> <em>flight</em>.</td>
</tr>
<tr>
<td>NUMMOD</td>
<td>Before the storm JetBlue canceled <em>1000 flights</em>.</td>
</tr>
<tr>
<td>APPOS</td>
<td>United, a <em>unit</em> of UAL, matched the fares.</td>
</tr>
<tr>
<td>DET</td>
<td>The <em>flight</em> was canceled.</td>
</tr>
<tr>
<td></td>
<td>Which <em>flight</em> was delayed?</td>
</tr>
<tr>
<td>CONJ</td>
<td>We <em>flew</em> to Denver and <em>drove</em> to Steamboat.</td>
</tr>
<tr>
<td>CC</td>
<td>We flew to Denver and <em>drove</em> to Steamboat.</td>
</tr>
<tr>
<td>CASE</td>
<td>Book the flight <em>through</em> Houston.</td>
</tr>
</tbody>
</table>

In their most general form, the dependency structures we're discussing are simply directed graphs. That is, structures $G = (V, A)$ consisting of a set of vertices $V$, and a set of ordered pairs of vertices $A$, which we'll refer to as arcs.

For the most part we will assume that the set of vertices, $V$, corresponds exactly to the set of words in a given sentence. However, they might also correspond to punctuation, or when dealing with morphologically complex languages the set of vertices might consist of stems and affixes of the kind discussed in Chapter 3. The set of arcs, $A$, captures the head-dependent and grammatical function relationships between the elements in $V$.

Further constraints on these dependency structures are specific to the underlying grammatical theory or formalism. Among the more frequent restrictions are that the structures must be connected, have a designated root node, and be acyclic or planar.

Of most relevance to the parsing approaches discussed in this chapter is the common, computationally-motivated, restriction to rooted trees. That is, a dependency tree is a directed graph that satisfies the following constraints:

1. There is a single designated root node that has no incoming arcs.
2. With the exception of the root node, each vertex has exactly one incoming arc.
3. There is a unique path from the root node to each vertex in $V$.

Taken together, these constraints ensure that each word has a single head, that the dependency structure is connected, and that there is a single root node from which one can follow a unique directed path to each of the words in the sentence.

### 14.2.1 Projectivity

The notion of projectivity imposes an additional constraint that is derived from the order of the words in the input, and is closely related to the context-free nature of human languages discussed in Chapter 11. An arc from a head to a dependent is said to be projective if there is a path from the head to every word that lies between the head and the dependent in the sentence. A dependency tree is then said to be projective if all the arcs that make it up are projective. All the dependency trees we've seen thus far have been projective. There are, however, many perfectly valid...
Pure (projective, 1\textsuperscript{st} order) dependency parsing is only cubic [Eisner, 1996]

Non-projective dependency parsing useful for Czech & other languages – MST algorithms [McDonald et al., 2005]
Transition-based Dependency Parsing

Figure 14.7 Trace of a transition-based parse.

<table>
<thead>
<tr>
<th>Step</th>
<th>Stack</th>
<th>Word List</th>
<th>Action</th>
<th>Relation Added</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>[root]</td>
<td>[book, me, the, morning, flight]</td>
<td>SHIFT</td>
<td>(book → me)</td>
</tr>
<tr>
<td>1</td>
<td>[root, book]</td>
<td>[me, the, morning, flight]</td>
<td>SHIFT</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>[root, book, me]</td>
<td>[the, morning, flight]</td>
<td>RIGHTARC</td>
<td>(morning ← flight)</td>
</tr>
<tr>
<td>3</td>
<td>[root, book]</td>
<td>[the, morning, flight]</td>
<td>SHIFT</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>[root, book, the]</td>
<td>[morning, flight]</td>
<td>SHIFT</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>[root, book, the, morning]</td>
<td>[flight]</td>
<td>LEFTARC</td>
<td>(the ← flight)</td>
</tr>
<tr>
<td>6</td>
<td>[root, book, the, morning, flight]</td>
<td>[]</td>
<td>LEFTARC</td>
<td>(book → flight)</td>
</tr>
<tr>
<td>7</td>
<td>[root, book, the, flight]</td>
<td>[]</td>
<td>LEFTARC</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>[root, book, flight]</td>
<td>[]</td>
<td>RIGHTARC</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>[root, book]</td>
<td>[]</td>
<td>RIGHTARC</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>[root]</td>
<td>[]</td>
<td>Done</td>
<td></td>
</tr>
</tbody>
</table>
Parsing: Other Models and Methods


- Transition-based Dependency Parsing [Yamada and Matsumoto, 2003; Nivre, 2003]

- Tree-Insertion Grammar, DOP [Schabes and Waters, 1995; Hwa, 1998; Scha, 1990; Bod, 1993; Goodman, 1996; Bansal and Klein, 2010]

- Tree-Adjoining Grammar [Resnik, 1992; Joshi and Schabes, 1998; Chiang, 2000]

- Shift-Reduce Parser [Nivre and Scholz, 2004; Sagae and Lavie, 2005]

- Other: Reranking, A*, K-Best, Self-training, Co-training, System Combination, Cross-lingual Transfer [Sarkar, 2001; Steedman et al., 2003; Charniak and Johnson, 2005; Hwa et al., 2005; Huang and Chiang, 2005; McClosky et al., 2006; Fossum and Knight, 2009; Pauls and Klein, 2009; McDonald et al., 2011]

CCG Parsing

- Combinatory Categorial Grammars:

- Each category encodes an argument sequence (fwd/bwd slashes specify argument order/direction)

- Closely related to lambda calculus

- Captures both syntactic and semantic info

- Naturally allows meaning representation and semantic parsing (next week!)

```
John ⊨ NP
shares ⊨ NP
buys ⊨ (S\NP)/NP
sleeps ⊨ S\NP
well ⊨ (S\NP)\(S\NP)
```

```
S
   NP  S\NP
      |   |
   John (S\NP)/NP buys shares
```
Parser Reranking

- Can first get the $k$-best list of parses based on parser probability
- Then we can fire features on full tree (as opposed to local features in the parser’s dynamic program)
- Can fire non-local, global features like tree depth, width, right-branching vs left-branching, etc.
- See [Charniak and Johnson, 2005] for feature list.
Data Oriented Parsing (TIGs)

- DOP is formally a Tree-Insertion Grammar, i.e., we can rewrite a large subtree in a single step.
- Hence, this brings in derivational ambiguity.
Data Oriented Parsing (TIGs)
Neural Models for Parsing
Word Embeddings for Parsing

- Discrete or continuous, trained on large amounts of context

- **BROWN** (Brown et al., 1992):

  ![Brown Clustering Example](image)

  - apple \(\rightarrow\) 000
  - pear \(\rightarrow\) 001
  - Apple \(\rightarrow\) 010
  - IBM \(\rightarrow\) 011
  - bought \(\rightarrow\) 100
  - run \(\rightarrow\) 101
  - of \(\rightarrow\) 110
  - in \(\rightarrow\) 111

- **SKIPGRAM** (Mikolov et al., 2013):

  ![Skipgram Model](image)

  - INPUT: \(w(t)\)
  - PROJECTION: \(w(t-2), w(t-1), w(t+1), w(t+2)\)
  - OUTPUT:
    - apple \(\rightarrow\) [0.65, 0.15, -0.21, 0.15, 0.70, -0.90]
    - pear \(\rightarrow\) [0.51, 0.05, -0.32, 0.20, 0.80, -0.95]
    - Apple \(\rightarrow\) [0.11, 0.33, 0.51, -0.05, -0.41, 0.50]

[Koo et al., 2008; Bansal et al., 2014]
Word Embeddings for Parsing

- Condition on dependency context instead of linear, then convert each dependency to a tuple:

\[
\text{dep label} \quad \text{grandparent} \quad \text{parent} \quad \text{child} \quad \text{dep label} \\
\begin{bmatrix} PMOD_{<L>} \quad \text{regulation}_{<G>} \quad \text{of} \quad \text{safety}\quad PMOD_{<L>} \end{bmatrix}
\]

- 10% rel. error reduction over 90-92% parsers

[Mr., Mrs., Ms., Prof., III, Jr., Dr.]
[Jeffrey, William, Dan, Robert, Stephen, Peter, John, Richard, ...]
[Portugal, Iran, Cuba, Ecuador, Greece, Thailand, Indonesia, ...]

[his, your, her, its, their, my, our]
[Your, Our, Its, My, His, Their, Her]
[truly, wildly, politically, financially, completely, potentially, ...]
Neural Dependency Parser

[Chen and Manning, 2014; CS224n]

**Softmax probabilities**

Output layer $y$

$y = \text{softmax}(Uh + b_2)$

Hidden layer $h$

$h = \text{ReLU}(Wx + b_1)$

Input layer $x$

lookup + concat

Cross-entropy error will be back-propagated to the embeddings.
Neural Dependency Parser

Further developments in transition-based neural dependency parsing

This work was further developed and improved by others, including in particular at Google

- Bigger, deeper networks with better tuned hyperparameters
- Beam search
- Global, conditional random field (CRF)-style inference over the decision sequence

Leading to SyntaxNet and the Parsey McParseFace model

https://research.googleblog.com/2016/05/announcing-syntaxnet-worlds-most.html

<table>
<thead>
<tr>
<th>Method</th>
<th>UAS</th>
<th>LAS (PTB WSJ SD 3.3)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chen &amp; Manning 2014</td>
<td>92.0</td>
<td>89.7</td>
</tr>
<tr>
<td>Weiss et al. 2015</td>
<td>93.99</td>
<td>92.05</td>
</tr>
<tr>
<td>Andor et al. 2016</td>
<td>94.61</td>
<td>92.79</td>
</tr>
</tbody>
</table>
Neural Constituent Parser

Compositional Vector Grammar (CVG)

\[ p^{(1)} = f \left( W^{(B,C)} \begin{bmatrix} b \\ c \end{bmatrix} \right) \]

\[ s \left( p^{(1)} \right) = (v^{(B,C)})^T p^{(1)} + \log P(P_1 \rightarrow B \ C) \]

\[ P((P_1, p_1) \rightarrow (B, b)(C, c)) = P(p_1 \rightarrow b \ c|P_1 \rightarrow B \ C)P(P_1 \rightarrow B \ C) \]

\[ p^{(2)} = f \left( W^{(A,P_1)} \begin{bmatrix} a \\ p^{(1)} \end{bmatrix} \right) \]

\[ s \left( p^{(2)} \right) = (v^{(A,P_1)})^T p^{(2)} + \log P(P_2 \rightarrow A \ P_1) \]

Goodness of a tree is measured in terms of its score and the CVG score of a complete tree is the sum of the scores at each node →

\[ s(CVG(\theta, x, \hat{y})) = \sum_{d \in N(\hat{y})} s \left( p^d \right) \]