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

- Phrase-structure parsing or Bracketing

```
S
 /  
NP   VP
 / 
NNP  VBD
|  /  
John met NP
|   
|   /  
|   her
```

- Demos: [https://parser.kitaev.io/](https://parser.kitaev.io/)
### Grammar and Lexicon

<table>
<thead>
<tr>
<th>Grammar</th>
<th>Lexicon</th>
</tr>
</thead>
<tbody>
<tr>
<td>(S \rightarrow NP \ VP)</td>
<td>(Det \rightarrow that \</td>
</tr>
<tr>
<td>(S \rightarrow Aux \ NP \ VP)</td>
<td>(Noun \rightarrow book \</td>
</tr>
<tr>
<td>(S \rightarrow VP)</td>
<td>(Verb \rightarrow book \</td>
</tr>
<tr>
<td>(NP \rightarrow Pronoun)</td>
<td>(Pronoun \rightarrow I \</td>
</tr>
<tr>
<td>(NP \rightarrow Proper-Noun)</td>
<td>(Proper-Noun \rightarrow Houston \</td>
</tr>
<tr>
<td>(NP \rightarrow Det Nominal)</td>
<td>(Aux \rightarrow does)</td>
</tr>
<tr>
<td>Nominal \rightarrow Noun</td>
<td>Preposition \rightarrow from \</td>
</tr>
<tr>
<td>Nominal \rightarrow Nominal Noun</td>
<td></td>
</tr>
<tr>
<td>Nominal \rightarrow Nominal PP</td>
<td></td>
</tr>
<tr>
<td>(VP \rightarrow Verb)</td>
<td></td>
</tr>
<tr>
<td>(VP \rightarrow Verb \ NP)</td>
<td></td>
</tr>
<tr>
<td>(VP \rightarrow Verb \ NP \ PP)</td>
<td></td>
</tr>
<tr>
<td>(VP \rightarrow Verb \ PP)</td>
<td></td>
</tr>
<tr>
<td>(VP \rightarrow VP \ PP)</td>
<td></td>
</tr>
<tr>
<td>(PP \rightarrow Preposition \ NP)</td>
<td></td>
</tr>
</tbody>
</table>

**Figure 12.1** The \(\mathcal{L}_1\) miniature English grammar and lexicon.
Figure 12.2  Two parse trees for an ambiguous sentence. The parse on the left corresponds to the humorous reading in which the elephant is in the pajamas, the parse on the right corresponds to the reading in which Captain Spaulding did the shooting in his pajamas.
A context-free grammar is a tuple $\langle N, T, S, R \rangle$

- $N$ : 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: $S \rightarrow NP \ VP$, $VP \rightarrow VP CC VP$
  - Also called rewrites, productions, or local trees
Probabilistic Context-free Grammars

- 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
Figure 13.2 Two parse trees for an ambiguous sentence. The parse on the left corresponds to the sensible meaning “Book a flight that serves dinner”, while the parse on the right corresponds to the nonsensical meaning “Book a flight on behalf of ‘the dinner’”.

Let's formalize this intuition that picking the parse with the highest probability is the correct way to do disambiguation. Consider all the possible parse trees for a given sentence $S$. The string of words $S$ is called the yield of any parse tree over $S$.

Thus, out of all parse trees with a yield of $S$, the disambiguation algorithm picks the parse tree that is most probable given $S$:

$$\hat{T}(S) = \arg\max_{T \in TS} P(T | S)$$

By definition, the probability $P(T | S)$ can be rewritten as $P(T, S) / P(S)$, thus leading to

$$\hat{T}(S) = \arg\max_{T \in TS} P(T, S) / P(S)$$
Probabilistic Context-free Grammars

\[ P(T_{left}) = 0.05 \times 0.20 \times 0.20 \times 0.75 \times 0.30 \times 0.60 \times 0.10 \times 0.40 = 2.2 \times 10^{-6} \]

\[ P(T_{right}) = 0.05 \times 0.10 \times 0.20 \times 0.15 \times 0.75 \times 0.75 \times 0.30 \times 0.60 \times 0.10 \times 0.40 = 6.1 \times 10^{-7} \]
Treebank PCFG Probabilities

- Simple MLE approach: just count the frequency of each rule and normalize (similar to what we did for tagging)

\[
P(\alpha \rightarrow \beta | \alpha) = \frac{\text{Count}(\alpha \rightarrow \beta)}{\sum_{\gamma} \text{Count}(\alpha \rightarrow \gamma)} = \frac{\text{Count}(\alpha \rightarrow \beta)}{\text{Count}(\alpha)}
\]

[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))))))
```
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]
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]}

[CKY (or CYK) Parsing Algorithm (Bottom-up)]

[Cocke, 1970; Kasami, 1965; Younger, 1967]
prove our estimates by beginning with a parser with equal rule probabilities, then
count for each parse of a sentence and weight each of these partial counts by the
don’t know which parse to count the rules in. Instead, we need to keep a separate
as simple as this: parse the corpus, increment a counter for every rule in the parse,
generate the counts we need for computing PCFG rule probabilities by first parsing
occurs and then normalizing.

each expansion of a non-terminal by counting the number of times that expansion
tion of parse trees in English, Chinese, and other languages that is distributed by the
treebanks and the commonly used
of already parsed sentences. Recall that we introduced in Chapter 11 the idea of
abilities for the rules of a grammar. The simplest way is to use a treebank, a corpus
(13.17)
function is left as an exercise to the reader.

Figure 13.3 The probabilistic CKY algorithm for finding the maximum probability parse
of a string of num_words words given a PCFG grammar with num_rules rules in Chomsky
normal form. back is an array of backpointers used to recover the best parse. The build_tree
function is left as an exercise to the reader.
Latent Variable Grammars

Parse Tree

Grammar G

Parameters $\theta$

Sentence $w$

Derivations $t : T$

[ 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

\[
\begin{aligned}
S[X_1] \\
NP[X_2] & \quad VP[X_4] & \quad X_7 \\
PRP[X_3] & \quad VBD[X_5] & \quad ADJP[X_6] \\
He & \quad was & \quad right
\end{aligned}
\]

[Brackets are known]
[Base categories are known]
[Only induce subcategories]

[Forward-backward (last week) but for trees]

[Forward]

[Backward]

[He was right]

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

DT

- the (0.50)
- a (0.24)
- The (0.08)

DT-1
- a (0.61)
- the (0.19)
- an (0.11)

DT-2
- the (0.80)
- The (0.15)
- a (0.01)

DT-3
- this (0.39)
- that (0.28)
- That (0.11)

DT-4
- some (0.20)
- all (0.19)
- those (0.12)

[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>
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]
13.8 Evaluating Parsers

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

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

\[
F_\beta = \frac{(\beta^2 + 1)PR}{\beta^2 P + R} \quad F_1 = \frac{2PR}{P + R}
\]

**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]
Web Features for Syntactic Parsing

They considered running the ad during the Super Bowl.

- Web Ngrams

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

- 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
Visual Recognition Cues

“A dog is standing next to a woman on a couch”

Ambiguity:
(dog next to woman) on couch vs dog next to (woman on couch)

Figure 1: Overview of our approach. We propose a model for simultaneous 2D semantic segmentation and prepositional phrase attachment resolution by reasoning about sentence parses. The language and vision modules each produce \( M \) diverse hypotheses, and the goal is to select a pair of consistent hypotheses. In this example the ambiguity to be resolved from the image caption is whether the dog is standing on or next to the couch. Both modules benefit by selecting a pair of compatible hypotheses.
Dependency Parsing
Dependency Parsing

- Predicting directed head-modifier relationship pairs

Demos: [https://corenlp.run/](https://corenlp.run/)
Can convert (lexicalized) constituent tree to dependency tree (each local binary tree gives us a dependency attachment from head to modifier)
### Dependency Parsing

#### Clausal Argument Relations

<table>
<thead>
<tr>
<th>Relation</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 complement</td>
</tr>
</tbody>
</table>

#### Nominal Modifier Relations

<table>
<thead>
<tr>
<th>Relation</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>

#### Other Notable Relations

<table>
<thead>
<tr>
<th>Relation</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 shows a subset of the relations from this effort. Figure 14.3 provides some example sentences illustrating selected relations.

The motivation for all of the relations in the Universal Dependency scheme is beyond the scope of this chapter, but the core set of frequently used relations can be broken into two sets: clausal relations that describe syntactic roles with respect to a predicate (often a verb), and modifier relations that categorize the ways that words that can modify their heads.

Consider the following example sentence:

(14.2) United canceled the morning flights to Houston

The clausal relations NSUBJ and DOBJ identify the subject and direct object of the predicate cancel, while the NMOD, DET, and CASE relations denote modifiers of the nouns flights and Houston.
## 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 canceled the flight.</td>
</tr>
<tr>
<td>DOBJ</td>
<td>United diverted the <strong>flight</strong> to Reno.</td>
</tr>
<tr>
<td></td>
<td>We <em>booked</em> her the first <strong>flight</strong> to Miami.</td>
</tr>
<tr>
<td>IOBJ</td>
<td>We <em>booked her</em> the flight to Miami.</td>
</tr>
<tr>
<td>NMOD</td>
<td>We took the <strong>morning flight</strong>.</td>
</tr>
<tr>
<td>AMOD</td>
<td>Book the <strong>cheapest flight</strong>.</td>
</tr>
<tr>
<td>NUMMOD</td>
<td>Before the storm JetBlue canceled <strong>1000 flights</strong>.</td>
</tr>
<tr>
<td>APPOS</td>
<td>United, a <strong>unit</strong> of UAL, matched the fares.</td>
</tr>
<tr>
<td>DET</td>
<td>The <strong>flight</strong> was canceled.</td>
</tr>
<tr>
<td></td>
<td><strong>Which flight</strong> 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 <strong>and drove</strong> to Steamboat.</td>
</tr>
<tr>
<td>CASE</td>
<td>Book the flight <strong>through</strong> Houston.</td>
</tr>
</tbody>
</table>
Dependency Parsing

- Pure (projective, 1\textsuperscript{st} order) dependency parsing is only cubic \[ \text{[Eisner, 1996]} \]

- Non-projective dependency parsing useful for Czech & other languages – MST algorithms \[ \text{[McDonald et al., 2005]} \]

```plaintext
root  John  saw  a  dog  yesterday  which  was  a  Yorkshire  Terrier
```


Transition-based Dependency Parsing

<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></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>(book → me)</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>SHIFT</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>[root, book, the, morning, flight]</td>
<td>[]</td>
<td>LEFTARC</td>
<td>(morning ← flight)</td>
</tr>
<tr>
<td>7</td>
<td>[root, book, the, flight]</td>
<td>[]</td>
<td>LEFTARC</td>
<td>(the ← flight)</td>
</tr>
<tr>
<td>8</td>
<td>[root, book, flight]</td>
<td>[]</td>
<td>RIGHTARC</td>
<td>(book → flight)</td>
</tr>
<tr>
<td>10</td>
<td>[root]</td>
<td>[]</td>
<td>Done</td>
<td></td>
</tr>
</tbody>
</table>
 Parsing: Other Models and Methods

- Combinatory Categorial Grammar [Steedman, 1996, 2000; Clark and Curran, 2004] (next week for semantic parsing)

- 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]

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)

TIG:

\[ \phi \]

\[ \psi \]

\[ \phi' \]

\[ \psi \]

\[ S \]

\[ NP \downarrow \]

\[ VP \]

\[ V \]

\[ NP \downarrow \]

\[ saw \]

\[ D \downarrow \]

\[ N \]

\[ man \]

\[ S \]

\[ NP \]

\[ VP \]

\[ D \downarrow \]

\[ N \]

\[ man \]

\[ V \]

\[ NP \downarrow \]

\[ saw \]
Neural Models for Parsing
Word Embeddings for Parsing

- Discrete or continuous, trained on large amounts of context

- BROWN (Brown et al., 1992):

  ![Brown Word Cluster Hierarchy]

    ```plaintext
    apple  pear  Apple  IBM  bought  run  of  in
    000    001   010   011   100   101   110   111
    apple  →  000
    pear   →  001
    Apple  →  010
    ``

- SKIPGRAM (Mikolov et al., 2013):

  ![Skipgram Diagram]

  ```plaintext
  INPUT    PROJECTION    OUTPUT
  w(t)     w(t-2)       [0.65 0.15 -0.21 0.15 0.70 -0.90]
  w(t-1)   [0.51 0.05 -0.32 0.20 0.80 -0.95]
  w(t+1)   [0.11 0.33 0.51 -0.05 -0.41 0.50]
  w(t+2)
  ```

[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:

\[
\begin{array}{cccccc}
\text{dep label} & \text{grandparent} & \text{parent} & \text{child} & \text{dep label} \\
\text{PMOD}_{<L>} & \text{regulation}_{<G>} & \text{of} & \text{safety} & \text{PMOD}_{<L>}
\end{array}
\]

- [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, ...]

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

[Bansal et al., 2014]
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.

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 →

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

https://www.aclweb.org/anthology/P13-1045.pdf
Other Recent Neural Parsers

- Recurrent Neural Network Grammars. [https://www.aclweb.org/anthology/N16-1024.pdf](https://www.aclweb.org/anthology/N16-1024.pdf)
- Many Languages, One Parser. [https://www.aclweb.org/anthology/Q16-1031.pdf](https://www.aclweb.org/anthology/Q16-1031.pdf)
- Cross-lingual Dependency Parsing Based on Distributed Representations. [https://www.aclweb.org/anthology/P15-1119.pdf](https://www.aclweb.org/anthology/P15-1119.pdf)
- Transition-Based Dependency Parsing with Stack Long Short-Term Memory. [https://www.aclweb.org/anthology/P15-1033.pdf](https://www.aclweb.org/anthology/P15-1033.pdf)
- Constituency Parsing with a Self-Attentive Encoder. [https://www.aclweb.org/anthology/P18-1249.pdf](https://www.aclweb.org/anthology/P18-1249.pdf)