Syntax and Parsing I

Constituency Parsing

Slav Petrov – Google

Thanks to:
Dan Klein, Ryan McDonald, Alexander Rush, Joakim Nivre, Greg Durrett, David Weiss, Luheng He, Timothy Dozat

NYU Fall 2018
On January 13, 2018, a false ballistic missile alert was issued via the Emergency Alert System and Commercial Mobile Alert System over television, radio, and cellphones in the U.S. state of Hawaii. The alert stated that there was an incoming ballistic missile threat to Hawaii, advised residents to seek shelter, and concluded "This is not a drill". The message was sent at 8:07 a.m. local time.
They solved the problem with statistics.
They solved the problem with statistics
Constituency and Dependency

They solved the problem with statistics
They solved the problem with statistics
Influential members of the House Ways and Means Committee introduced legislation that would restrict how the new savings-and-loan bailout agency can raise capital, creating another potential obstacle to the government's sale of sick thrifts.
Dependency Parsing

- Directed edges between pairs of word (head, dependent)
- Can handle free word-order languages
- Very efficient decoding algorithms exist
- Second part of today’s lecture
Attachments

• I cleaned the dishes from dinner
• I cleaned the dishes with detergent
• I cleaned the dishes in my pajamas
• I cleaned the dishes in the sink
Classical NLP: Parsing

- Write symbolic or logical rules:

  \[
  \begin{align*}
  \text{VBD} & \quad \text{VB} \\
  \text{VBN} & \quad \text{VBZ} \quad \text{VBP} \quad \text{VBZ} \\
  \text{NNP} & \quad \text{NNS} \quad \text{NN} \quad \text{NNS} \quad \text{CD} \quad \text{NN}
  \end{align*}
  \]

  Fed raises interest rates 0.5 percent

- Use deduction systems to prove parses from words
  - Minimal grammar on “Fed raises” sentence: 36 parses
  - Real-size grammar: many millions of parses

- This scaled very badly, didn’t yield broad-coverage tools
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: \(S \rightarrow NP \ VP, VP \rightarrow VP \ CC \ VP\)
  - Also called rewrites, productions, or local trees

A PCFG adds:
- A top-down production probability per rule \(P(Y_1 Y_2 \ldots Y_k \mid X)\)
Treebank Grammars

- Need a PCFG for broad coverage parsing.
- Can take a grammar right off the trees (doesn’t work well):

  - S → NP VP.
  - NP → PRP 0.5
  - NP → DT NN 0.5
  - VP → VBD NP 1.0
  - PRP → She 1.0
  ...

- Better results by enriching the grammar (e.g., lexicalization).
- Can also get reasonable parsers without lexicalization.
Treebank Grammar Scale

- Treebank grammars can be enormous
  - As FSAs, the raw grammar has ~10K states, excluding the lexicon
  - Better parsers usually make the grammars larger, not smaller
Chomsky Normal Form

- **Chomsky normal form:**
  - All rules of the form \( X \rightarrow Y Z \) or \( X \rightarrow w \)
  - In principle, this is no limitation on the space of (P)CFGs
    - N-ary rules introduce new non-terminals
  - Unaries / empties are “promoted”
  - In practice it’s kind of a pain:
    - Reconstructing n-aries is easy
    - Reconstructing unaries is trickier
    - The straightforward transformations don’t preserve tree scores
  - Makes parsing algorithms simpler!
A Recursive Parser

bestScore(X,i,j,s)
  if (j = i+1)
    return tagScore(X,s[i])
  else
    return max score(X→YZ) *
      bestScore(Y,i,k) *
      bestScore(Z,k,j)
A Memoized Parser

- One small change:

```java
bestScore(X, i, j, s)
    if (scores[X][i][j] == null)
        if (j = i+1)
            score = tagScore(X, s[i])
        else
            score = max score(X->YZ) * bestScore(Y, i, k) * bestScore(Z, k, j)
    scores[X][i][j] = score
    return scores[X][i][j]
```
A Bottom-Up Parser (CKY)

• Can also organize things bottom-up

```plaintext
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]
```
Time: Theory

• How much time will it take to parse?

• For each diff (<= n)
  • For each i (<= n)
    • For each rule X → Y Z
      • For each split point k
        Do constant work

• Total time: |rules|*n^3

• Something like 5 sec for an unoptimized parse of a 20-word sentences, or 0.2sec for an optimized parser
Unary Rules

• Unary rules?

```python
bestScore(X,i,j,s)
    if (j = i+1)
        return tagScore(X,s[i])
    else
        return max
            max score(X→YZ) *
            bestScore(Y,i,k) *
            bestScore(Z,k,j)
            max score(X→Y) *
            bestScore(Y,i,j)
```
Treebank Grammars

- Need a PCFG for broad coverage parsing.
- Can take a grammar right off the trees (doesn’t work well):

  \[
  S \rightarrow NP \ VP \ . \ 1.0
  \]

  \[
  NP \rightarrow PRP \ 0.5 \quad NP \rightarrow DT \ NN \ 0.5
  \]

  \[
  VP \rightarrow VBD \ NP \ 1.0 \quad PRP \rightarrow She \ 1.0
  \]

  ...  

- Better results by enriching the grammar (e.g., lexicalization).
- Can also get reasonable parsers

<table>
<thead>
<tr>
<th>Model</th>
<th>F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Charniak ’96</td>
<td>72.0</td>
</tr>
</tbody>
</table>
Conditional Independence?

- Not every NP expansion can fill every NP slot

```
S
  NP
  PRP  VBD  NP
  She  heard  DT  NN
       the   noise
```

- A grammar with symbols like “NP” won’t be context-free
- Statistically, conditional independence too strong
Non-Independence

- Independence assumptions are often too strong.

- Example: the expansion of an NP is highly dependent on the parent of the NP (i.e., subjects vs. objects).

- Also: the subject and object expansions are correlated!
The Game of Designing a Grammar

- Structure Annotation [Johnson ‘98, Klein & Manning ’03]
- Lexicalization [Collins ‘99, Charniak ‘00]
- Latent Variables [Matsuzaki et al. 05, Petrov et al. ‘06]
- (Neural) CRF Parsing [Hall et al. ’14, Durrett & Klein ‘15]
A Fully Annotated (Unlexicalized) Tree

[Klein & Manning ’03]

<table>
<thead>
<tr>
<th>Model</th>
<th>F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Charniak ’96</td>
<td>72.0</td>
</tr>
<tr>
<td>Klein&amp;Manning ’03</td>
<td>86.3</td>
</tr>
</tbody>
</table>
The Game of Designing a Grammar

- Annotation refines base treebank symbols to improve statistical fit of the grammar
  - Head lexicalization [Collins ’99, Charniak ’00]
Problems with PCFGs

If we do no annotation, these trees differ only in one rule:

- VP $\rightarrow$ VP PP
- NP $\rightarrow$ NP PP

Parse will go one way or the other, regardless of words.
We addressed this in one way with unlexicalized grammars (how?)
Lexicalization allows us to be sensitive to specific words
Problems with PCFGs

- What’s different between basic PCFG scores here?
- What (lexical) correlations need to be scored?
Lexicalized Trees [Charniak ’97, Collins ’97]

- Add “headwords” to each phrasal node
  - Syntactic vs. semantic heads
  - Headship not in (most) treebanks
  - Usually use head rules, e.g.:
    - NP:
      - Take leftmost NP
      - Take rightmost N*
      - Take rightmost JJ
      - Take right child
    - VP:
      - Take leftmost VB*
      - Take leftmost VP
      - Take left child
Lexicalized Grammars

Challenges:

- Many parameters to estimate: requires sophisticated smoothing techniques
- Exact inference is too slow: requires pruning heuristics
- Difficult to adapt to new languages: At least head rules need to be specified, typically more changes needed

<table>
<thead>
<tr>
<th>Model</th>
<th>F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Klein&amp;Manning ’03</td>
<td>86.3</td>
</tr>
<tr>
<td>Charniak ’00</td>
<td>89.7</td>
</tr>
</tbody>
</table>
The Game of Designing a Grammar

- Annotation refines base treebank symbols to improve statistical fit of the grammar
  - Automatic clustering
Latent Variable Grammars

[Matsuzaki et al. ’05, Petrov et al. ’06]

Parse Tree

Sentence $T$

Derivations $t : T$

Parameters $\theta$

Grammar $G$

<table>
<thead>
<tr>
<th>Rule</th>
<th>Parameter</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>$S_1 \rightarrow NP_1\ VP_1$</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>Lexicon</td>
<td></td>
</tr>
<tr>
<td>$PRP_0 \rightarrow She$</td>
<td>?</td>
</tr>
<tr>
<td>$PRP_1 \rightarrow She$</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>
</tbody>
</table>
Learning Latent Annotations

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

Just like Forward-Backward for HMMs.

<table>
<thead>
<tr>
<th>Model</th>
<th>F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Charniak ’00</td>
<td>89.7</td>
</tr>
<tr>
<td>Petrov et al. ‘06</td>
<td>90.1</td>
</tr>
</tbody>
</table>
The Game of Designing a Grammar

- Annotation refines base treebank symbols to improve statistical fit of the grammar
  - CRF Parsing (+Neural Network Representations)
Generative vs. Discriminative

### Generative

Maximize joint likelihood of gold tree and sentence

EM-algorithm

**EASY:** expectations over observed trees

[W1 W2 ... Wn]

[Matsuzaki et al. ’05, Petrov et al. ’06]

### Discriminative

Maximize conditional likelihood of gold tree given sentence

Gradient-based algorithm

**HARD:** expectations over all trees

[W1 W2 ... Wn]

[Petrov & Klein ’07, ’08]
He gave a speech

Score of VP over this span

Be a tree

CRF Parsing Sparse Features

\[ P(T|x) \propto \prod_{r \in T} \exp(\text{score}(r)) \]

\[
\text{score}(2\text{NP}_{7} \rightarrow 2\text{NP}_{4} 4\text{PP}_{7}) = w^{\top} f(2\text{NP}_{7} \rightarrow 2\text{NP}_{4} 4\text{PP}_{7})
\]

FirstWord = a & NP $\rightarrow$ NP PP

PrevWord = gave & NP $\rightarrow$ NP PP

AfterSplit = on & NP $\rightarrow$ NP PP

FirstWord = a & NP

...
Neural CRF Model

\[
score(2NP_7 \rightarrow 2NP_4 4PP_7) = \\
W \odot \left( f_s(2X_7 \rightarrow 2X_4 4X_7) f_o^T(NP \rightarrow NP PP) \right)
\]

\[
f_s = g(Hv)
\]

(arbitrary neural network)

He gave a speech on foreign policy.

[Durrett et al. ’15]

<table>
<thead>
<tr>
<th>Model</th>
<th>F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Petrov et al. ‘06</td>
<td>90.1</td>
</tr>
<tr>
<td>Durrett et. al. ‘15</td>
<td>91.1</td>
</tr>
</tbody>
</table>
Parsing with Self-Attention

...(VP(VBD fled)(NP(DT the)(NN market))...)
LSTM Parsing [Vinyals et al. ’15]

- Treat parsing as a sequence-to-sequence prediction problem
- Completely ignores tree structure, uses LSTMs as black boxes
- No global normalization, only local normalization

**Model**

<table>
<thead>
<tr>
<th></th>
<th>F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Durrett et. al. ‘15</td>
<td>91.1</td>
</tr>
<tr>
<td>Vinyals et al. ‘15</td>
<td>88.6*</td>
</tr>
</tbody>
</table>
Detailed English Results (Old)

- Single Parser
  - Charniak '00
  - Petrov et al. '06
  - Carreras et al. '08
  - Hall '12
  - Durrett et al. '15
  - Zhu et al. '13
  - Dyer et al. '16

- Self-Trained
  - Huang & Harper '08
  - [Charniak & Johnson '05]
  - [McClosky et al. '06]

- Reranker
  - [Charniak & Johnson '05]
  - [McClosky et al. '06]

- Product
  - [Charniak & Johnson '05]
  - [McClosky et al. '06]

- Combination
  - [Sagae & Lavie '06]
  - [Fossum & Knight '09]
  - [Zhang et al. '09]
  - [Vinyals et al. '16]
Multi-Lingual Results

Test set F₁ all lengths

- Petrov et al. '06*
- Hall et al. '14
- Durrett et al. '15
- Kitaev & Klein '18

Languages:
- Arabic
- Basque
- French
- German
- Hebrew
- Hungarian
- Korean
- Polish
- Swedish
- Average