Features of Statistical Parsers

Mark Johnson
Brown University
April, 2005

Joint work with Eugene Charniak, Michael Collins and students in BLLIP
Supported by NSF grants LIS 9720368 and IIS0095940
Talk outline

• Who cares about parsing?

• PCFGs, maximum likelihood estimation and (lack of) independence

• Dependence, arbitrary features and exponential parsing models

• Joint vs. conditional maximum likelihood estimation of exponential models

• Features of an exponential parsing model
Why parsing?

• Applications
  – Recovering syntactic structure
    * Information extraction
    * Question answering
    * Machine translation
  – Language modeling (distinguishing likely from unlikely sentences)
    * Speech recognition
    * Speech error detection and correction

• Science
  – First step on the path to meaning
Short history of parsing

- (1960s–1970s) basic dynamic programming algorithms, ATNs
  - inspired by linguistic theory (transformational grammar)
- (1980s–1990s) “unification” grammar (no deep structure)
  - parsers provably implement a specific linguistic grammar
  - grammar ~ axioms, UG ~ logic, parsing ~ deduction
  - emphasis on grammar formalisms
- (1980s–??) direct statistical models (e.g., n-gram models)
  - revolutionized speech recognition, practical applications
- (1990s–??) statistical models of linguistic structure
  - “curse of dimensionality” limits direct statistical models
  - emphasis on frequent constructions
N-gram models of strings

- Model dependencies between \( n \) adjacent words (\( n \)-gram models)
  - predict a word based on the \( n - 1 \) preceding words

\[
\$ \rightarrow \text{the} \rightarrow \text{man} \rightarrow \text{in} \rightarrow \text{the} \rightarrow \text{hat} \rightarrow \text{drinks} \rightarrow \text{red} \rightarrow \text{wine} \rightarrow \$
\]

\( n = 1 : P(\text{hat}); \ n = 2 : P(\text{hat}|\text{the}); \ n = 3 : P(\text{hat}|\text{in}, \text{the}) \)

- N-gram models are \textit{universal approximators} as \( n \rightarrow \infty \)

- Probabilities estimated from real \textit{corpora}

- Probability distinguishes “good” from “bad” sentences

- These simple models work surprisingly well because \textit{they are lexicalized} and \textit{most dependencies are local}
Limitations of $n$-gram models

- *Curse of dimensionality:* the number of possible $n$-grams (sequences of $n$ words) grows exponentially with $n$
  - $n = 1 \Rightarrow V^n \approx 1,000$
  - $n = 2 \Rightarrow V^n \approx 1,000,000$
  - $n = 3 \Rightarrow V^n \approx 1,000,000,000$

- *Bias-variance dilemma:* As $n$ gets larger ...
  - the model becomes more accurate (lower bias)
  - more data is needed to learn the model (higher variance)

- *Structured models* collect statistics on linguistic structures
  - only model linguistic dependencies (strong bias)
  - ignore other dependencies (lower variance)
Probabilistic Context Free Grammars

\[
P(t) = P(S \rightarrow NP \ VP) \times \\
P(NP \rightarrow D \ N \ PP) \times \\
P(D \rightarrow \text{the}) \times \\
P(N \rightarrow \text{man}) \times \\
\ldots
\]

- Rules are associated with \textit{probabilities}
- Tree probability is the \textit{product of rule probabilities}
- \textit{Most probable tree} is “best guess” at correct syntactic structure
The Penn treebank contains hand-annotated parse trees for \( \sim 50,000 \) sentences

Treebanks also exist for the Brown corpus, the Switchboard corpus (spontaneous telephone conversations) and Chinese and Arabic corpora
Estimating a grammar from a treebank

• *Maximum likelihood principle*: Choose the grammar and rule probabilities that make the trees in the corpus *as likely as possible*
  
  – read the rules off the trees
  
  – for PCFGs, set rule probabilities to the *relative frequency* of each rule in the treebank

  \[
  P(\text{VP} \rightarrow \text{V NP}) = \frac{\text{Number of times } \text{VP} \rightarrow \text{V NP occurs}}{\text{Number of times } \text{VP occurs}}
  \]

• *If the language is generated by a PCFG and the treebank trees are its derivation trees, the estimated grammar converges to the true grammar.*
Estimating PCFGs from visible data

S
   / \  \
NP   VP
   \   \  
    rice grows

S
   / \  \
NP   VP
   \   \  
    rice grows

S
   / \  \
NP   VP
   \   \  
    corn grows

<table>
<thead>
<tr>
<th>Rule</th>
<th>Count</th>
<th>Rel Freq</th>
</tr>
</thead>
<tbody>
<tr>
<td>S → NP VP</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>NP → rice</td>
<td>2</td>
<td>2/3</td>
</tr>
<tr>
<td>NP → corn</td>
<td>1</td>
<td>1/3</td>
</tr>
<tr>
<td>VP → grows</td>
<td>3</td>
<td>1</td>
</tr>
</tbody>
</table>

$P\left(\begin{array}{c}
S \\
NP \\
\text{rice} \\
\text{grows}
\end{array}\right) = \frac{2}{3}$

$P\left(\begin{array}{c}
S \\
NP \\
\text{corn} \\
\text{grows}
\end{array}\right) = \frac{1}{3}$
Non-local constraints and PCFG MLE

\[
\begin{align*}
S & \rightarrow \text{NP VP} \\
\text{NP} & \rightarrow \text{rice} \\
\text{VP} & \rightarrow \text{grows} \\
S & \rightarrow \text{NP VP} \\
\text{NP} & \rightarrow \text{rice} \\
\text{VP} & \rightarrow \text{grows} \\
S & \rightarrow \text{NP VP} \\
\text{NP} & \rightarrow \text{bananas} \\
\text{VP} & \rightarrow \text{grow}
\end{align*}
\]

| Rule               | Count | Rel Freq | \[ P\left( \begin{array}{c} S \\
\text{NP} \\
\text{VP} \\
\text{rice} \\
\text{grows} \end{array} \right) \] | \[ P\left( \begin{array}{c} S \\
\text{NP} \\
\text{VP} \\
\text{bananas} \\
\text{grow} \end{array} \right) \] |
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>( S \rightarrow \text{NP VP} )</td>
<td>3</td>
<td>1</td>
<td>( \frac{4}{9} )</td>
<td>( \frac{1}{9} )</td>
</tr>
<tr>
<td>( \text{NP} \rightarrow \text{rice} )</td>
<td>2</td>
<td>( \frac{2}{3} )</td>
<td></td>
<td></td>
</tr>
<tr>
<td>( \text{NP} \rightarrow \text{bananas} )</td>
<td>1</td>
<td>( \frac{1}{3} )</td>
<td></td>
<td></td>
</tr>
<tr>
<td>( \text{VP} \rightarrow \text{grows} )</td>
<td>2</td>
<td>( \frac{2}{3} )</td>
<td></td>
<td></td>
</tr>
<tr>
<td>( \text{VP} \rightarrow \text{grow} )</td>
<td>1</td>
<td>( \frac{1}{3} )</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

partition function \( Z = \frac{5}{9} \)
Dividing by partition function $Z$

\[
P \left( \frac{S \quad \text{NP} \quad \text{VP}}{\text{rice} \quad \text{grows}} \right) = \frac{4}{9} \quad \frac{4}{5}
\]

\[
P \left( \frac{S \quad \text{NP} \quad \text{VP}}{\text{bananas} \quad \text{grow}} \right) = \frac{1}{9} \quad \frac{1}{5}
\]

\[
Z = \frac{5}{9}
\]

**Rule** | **Count** | **Rel Freq**
---|---|---
$S \rightarrow \text{NP} \ \text{VP}$ | 3 | 1
$\text{NP} \rightarrow \text{rice}$ | 2 | $\frac{2}{3}$
$\text{NP} \rightarrow \text{bananas}$ | 1 | $\frac{1}{3}$
$\text{VP} \rightarrow \text{grows}$ | 2 | $\frac{2}{3}$
$\text{VP} \rightarrow \text{grow}$ | 1 | $\frac{1}{3}$
Other values do do better!

\[
P\left(\begin{array}{c}
S \\
| NP \\
| VP \\
| rice \\
| grows 
\end{array}\right) = 2/6 \quad 2/3
\]

\[
P\left(\begin{array}{c}
S \\
| NP \\
| VP \\
| bananas \\
| grow 
\end{array}\right) = 1/6 \quad 1/3
\]

\[
Z = 3/6
\]

<table>
<thead>
<tr>
<th>Rule</th>
<th>Count</th>
<th>Rel Freq</th>
</tr>
</thead>
<tbody>
<tr>
<td>S \rightarrow NP VP</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>NP \rightarrow rice</td>
<td>2</td>
<td>2/3</td>
</tr>
<tr>
<td>NP \rightarrow bananas</td>
<td>1</td>
<td>1/3</td>
</tr>
<tr>
<td>VP \rightarrow grows</td>
<td>2</td>
<td>1/2</td>
</tr>
<tr>
<td>VP \rightarrow grow</td>
<td>1</td>
<td>1/2</td>
</tr>
</tbody>
</table>

(Abney 1997)
### Make dependencies local – GPSG-style

<table>
<thead>
<tr>
<th>rule</th>
<th>count</th>
<th>rel freq</th>
</tr>
</thead>
<tbody>
<tr>
<td>$S \rightarrow NP \ VP$ +singular +singular</td>
<td>2</td>
<td>2/3</td>
</tr>
<tr>
<td>$S \rightarrow NP \ VP$ +plural +plural</td>
<td>1</td>
<td>1/3</td>
</tr>
<tr>
<td>$NP \rightarrow$ rice +singular</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>$NP \rightarrow$ bananas +plural</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>$VP \rightarrow$ grows +singular</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>$VP \rightarrow$ grow +plural</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

\[
P \left( \begin{array}{c}
S \\
NP \quad VP \\
| \\
rice \quad grows
\end{array} \right) = \frac{2}{3}
\]

\[
P \left( \begin{array}{c}
S \\
NP \quad VP \\
| \\
bananas \quad grow
\end{array} \right) = \frac{1}{3}
\]
“Head to head” dependencies

Rules:

- S \( \rightarrow \) NP VP
  - drinks \( \rightarrow \) man drinks

- VP \( \rightarrow \) V NP
  - drinks \( \rightarrow \) drinks wine

- NP \( \rightarrow \) AP N
  - wine \( \rightarrow \) red wine

- ...  

- **Lexicalization** captures syntactic and semantic dependencies

- Lexicalized structural preferences may be most important
Summary so far

- Maximum likelihood is a good way of estimating a grammar
- Maximum likelihood estimation of a PCFG from a treebank is easy if the trees are accurate
- But real language has many more dependencies than treebank grammar describes

→ relative frequency estimator not MLE
  - Make non-local dependencies local by splitting categories

→ Astronomical number of possible categories

- Or find some way of dealing with non-local dependencies ...
Exponential models

- Rules are not independent $\Rightarrow$ in general $Z \neq 1$
- Models with dependencies between features are called \textit{exponential models}
  - Universe $\mathcal{T}$ (set of all possible parse trees)
  - Features $f = (f_1, \ldots, f_m)$ ($f_j(t) =$ value of $j$ feature on $t \in \mathcal{T}$)
  - Feature weights $\mathbf{w} = (w_1, \ldots, w_m)$

\[
P(t) = \frac{1}{Z} \exp \mathbf{w} \cdot f(t)
\]

\[
Z = \sum_{t' \in \mathcal{T}} \exp \mathbf{w} \cdot f(t')
\]

Hint: Think of $\exp \mathbf{w} \cdot f(t)$ as unnormalized probability of $t$
PCFGs are exponential models

\[ \mathcal{T} = \text{set of all trees generated by PCFG } G \]

\[ f_j(t) = \text{number of times the } j\text{th rule is used in } t \in \mathcal{T} \]

\[ p(r_j) = \text{probability of } j\text{th rule in } G \]

Set weight \( w_j = \log p(r_j) \)

\[
\begin{align*}
    f \left( \begin{array}{c}
        S \\
        \left( \begin{array}{c}
            \text{NP} \\
            \text{rice}
        \end{array} \right) \quad \left( \begin{array}{c}
            \text{VP} \\
            \text{grows}
        \end{array} \right)
    \end{array} \right) &= \left[ \begin{array}{c}
        1 \\
        1 \\
        0 \\
        1 \\
        0
    \end{array} \right] \\
    &\text{S} \rightarrow \text{NP} \quad \text{VP} \quad \text{NP} \rightarrow \text{rice} \quad \text{NP} \rightarrow \text{bananas} \quad \text{VP} \rightarrow \text{grows} \quad \text{VP} \rightarrow \text{grow}
\end{align*}
\]

\[ P_w(t) = \prod_{j=1}^{m} p(r_j)^{f_j(t)} = \prod_{j=1}^{m} (\exp w_j)^{f_j(t)} = \exp(w \cdot f(t)) \]

So a PCFG is just a special kind of exponential model with \( Z = 1 \).
Advantages of exponential models

- Exponential models are very flexible . . .

- Features $f$ can be \textit{any function of parses} . . .
  - whether a particular structure occurs in a parse
  - conjunctions of prosodic and syntactic structure

- Parses $t$ need not be trees, but \textit{can be anything at all}
  - Feature structures (LFG, HPSG)
  - Minimalist derivations

- Exponential models are related to other popular models
  - Harmony theory and optimality theory
  - Maxent models
Modeling dependencies

• It’s usually difficult to design a PCFG model that captures a particular set of dependencies
  – probability of the tree must be broken down into a product of conditional probability distributions
  – non-local dependencies must be expressed in terms of GPSG-style feature passing

• It’s easy to make exponential models sensitive to new dependencies
  – add a new feature functions to existing feature functions
  – figuring out what the right dependencies are is hard, but incorporating them into an exponential model is easy
MLE of exponential models is hard

• An exponential model associates features
  \[ f(t) = (f_1(t), \ldots, f_m(t)) \]
  with weights \( w = (w_1, \ldots, w_m) \)

  \[
  P(t) = \frac{1}{Z} \exp w \cdot f(t)
  \]

  \[
  Z = \sum_{t' \in \mathcal{T}} \exp w \cdot f(t')
  \]

• Given treebank \( (t_1, \ldots, t_n) \), MLE chooses \( w \) to maximize
  \[
  P(t_1) \times \ldots \times P(t_n), \text{ i.e., make the treebank as likely as possible}
  \]

• Computing \( P(t) \) requires the partition function \( Z \)

• Computing \( Z \) requires a sum over all parses \( \mathcal{T} \) for all sentences

\[ \Rightarrow \text{computing MLE of an exponential parsing model seems very hard} \]
Conditional ML estimation

- Conditional ML estimation chooses feature weights to maximize $P(t_1|s_1) \times \ldots \times P(t_n|s_n)$, where $s_i$ is string for $t_i$
  - choose feature weights to make $t_i$ most likely relative to parses $\mathcal{T}(s_i)$ for $s_i$

$\Rightarrow$ CMLE doesn’t involve other sentences

$$P(t|s) = \frac{1}{Z(s)} \exp \mathbf{w} \cdot f(t)$$

$$Z(s) = \sum_{t' \in \mathcal{T}(s)} \exp \mathbf{w} \cdot f(t')$$

- CMLE “only” involves repeatedly parsing training data
- With “wrong” models, CMLE often produces a more accurate parser than joint MLE
# Conditional vs joint MLE

## Rule count and relative frequency

<table>
<thead>
<tr>
<th>Rule</th>
<th>count</th>
<th>rel freq</th>
<th>rel freq</th>
</tr>
</thead>
<tbody>
<tr>
<td>VP → V</td>
<td>100</td>
<td>100/105</td>
<td>4/7</td>
</tr>
<tr>
<td>VP → V NP</td>
<td>3</td>
<td>3/105</td>
<td>1/7</td>
</tr>
<tr>
<td>VP → VP PP</td>
<td>2</td>
<td>2/105</td>
<td>2/7</td>
</tr>
<tr>
<td>NP → N</td>
<td>6</td>
<td>6/7</td>
<td>6/7</td>
</tr>
<tr>
<td>NP → NP PP</td>
<td>1</td>
<td>1/7</td>
<td>1/7</td>
</tr>
</tbody>
</table>
## Conditional ML estimation

<table>
<thead>
<tr>
<th>$s$</th>
<th>$f(\bar{t})$</th>
<th>${f(t) : t \in T(s), t \neq \bar{t}(s)}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>sentence 1</td>
<td>(1, 3, 2)</td>
<td>(2, 2, 3), (3, 1, 5), (2, 6, 3)</td>
</tr>
<tr>
<td>sentence 2</td>
<td>(7, 2, 1)</td>
<td>(2, 5, 5)</td>
</tr>
<tr>
<td>sentence 3</td>
<td>(2, 4, 2)</td>
<td>(1, 1, 7), (7, 2, 1)</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

- Parser designer specifies *feature functions* $f = (f_1, \ldots, f_m)$
- A parser produces trees $T(s)$ for each sentence $s$
- Treebank tells us correct tree $\bar{t}(s) \in T(s)$ for sentence $s$
- Feature functions $f$ apply to each tree $t \in T(s)$, producing feature values $f(t) = (f_1(t), \ldots, f_m(t))$
- MCLE estimates feature weights $\mathbf{w} = (\mathbf{w}_1, \ldots, \mathbf{w}_m)$
Coarse-to-fine parsing

- $Z(s)$ is still hard to compute $\Rightarrow$ make $T(s)$ even smaller!
- Restrict attention to 50-best parses produced by Charniak parser (a good PCFG-based parser)
- Exponential model is trained using CMLE to pick out best parse from Charniak’s 50-best parses

![Diagram of parse trees]

- Sentence
- Charniak parser
- Trees
- Features
- Parse scores
Parser evaluation

- A node’s *edge* is its label and beginning and ending *string positions*
- $E(t)$ is the set of edges associated with a tree $t$
- If $t$ is a parse tree and $\tilde{t}$ is the correct tree, then
  
  $\textit{precision} \quad P_\tilde{t}(t) = \frac{|E(t)|}{|E(t) \cap E(\tilde{t})|}$
  
  $\textit{recall} \quad R_\tilde{t}(t) = \frac{|E(\tilde{t})|}{|E(t) \cap E(\tilde{t})|}$
  
  $\textit{f-score} \quad F_\tilde{t}(t) = 2/(P_\tilde{t}(t)^{-1} + R_\tilde{t}(t)^{-1})$ (geometric mean of $P$ and $R$)

```
Elges
(0  NP  2)
(2  VP  3)
(0  S   3)
```

```
ROOT
  S
  NP   VP
    DT   N   VB
      the  dog  barks
```

26
Performance of Charniak parser

- F-score of Charniak’s most probable parse $= 0.896$
  (cross-validated on PTB sections 2-19)
- Oracle f-score of Charniak’s 50-best parses $= 0.965$ (66% redn)
Expt 1: Only “old” features

- Features: 1 \textit{log Charniak probability}, 10,124 Rule features
- Charniak’s parser already conditions on local trees!
- Feature selection: features must vary on 5 or more sentences
- Results: \( f\)-score = 0.894; baseline = 0.890; \(\approx 4\%\) error reduction

\(\Rightarrow\) \textit{discriminative training alone can improve accuracy}

\[
\begin{array}{c}
\text{ROOT} \\
\downarrow \\
S \\
\downarrow \\
NP \\
\downarrow \\
\text{WDT} \\
\downarrow \\
\text{That} \\
\downarrow \\
\text{VBD} \\
\downarrow \\
\text{PP} \\
\downarrow \\
\text{IN} \\
\downarrow \\
\text{over} \\
\downarrow \\
\text{NP} \\
\downarrow \\
\text{DT} \\
\downarrow \\
\text{the} \\
\downarrow \\
\text{permissible} \\
\downarrow \\
\text{line} \\
\downarrow \\
\text{IN} \\
\downarrow \\
\text{NP} \\
\downarrow \\
\text{PP} \\
\downarrow \\
\text{CC} \\
\downarrow \\
\text{and} \\
\downarrow \\
\text{JJ} \\
\downarrow \\
\text{fuzzy} \\
\downarrow \\
\text{NNS} \\
\downarrow \\
\text{feelings} \\
\end{array}
\]
Lexicalized and parent-annotated rules

- **Lexicalization** associates each constituent with its head
- **Ancestor annotation** provides a little “vertical context”
- **Context annotation** indicates constructions that only occur in main clause (c.f., Emonds)
There are at least two sensible notions of head (c.f., Grimshaw)

- **Functional heads:** determiners of NPs, auxiliary verbs of VPs, etc.
- **Lexical heads:** rightmost Ns of NPs, main verbs in VPs, etc.

In a Maxent model, it is easy to use both!
The SynSemHeads features collect pairs of functional and lexical heads of phrases.

This captures number agreement in NPs and aspects of other head-to-head dependencies.
n-gram rule features generalize rules

- Collects *adjacent constituents* in a local tree
- Also includes *relationship to head*
- Constituents can be *ancestor-annotated* and *lexicalized*
Head to head dependencies

- Head-to-head dependencies track the function-argument dependencies in a tree
- Co-ordination leads to phrases with multiple heads or functors
- Parameterized by head type and number of governors to include
Subject-Verb Agreement

- The SubjVerbAgr features are the POS of the subject NP’s lexical head and the VP’s functional head

```
ROOT
  S
  NP
    DT NNS VBP
    The rules force NP
    NNS TO VP
    executives to VB NP
    report NNS
    purchases
```
Head trees record all dependencies

- Head trees consist of a (lexical) head, all of its projections and (optionally) all of the siblings of these nodes

- These correspond roughly to TAG elementary trees
Tree n-gram

- A tree n-gram are tree fragments that connect sequences of adjacent n words
**Rightmost branch bias**

- The RightBranch feature’s value is the number of nodes on the right-most branch (ignoring punctuation)
- Reflects the tendency toward right branching in English
- Only 2 different features, but very useful in final model!
Constituent Heavyness and location

- Heavyness measures the constituent’s category, its (binned) size and (binned) closeness to the end of the sentence

> 5 words = 1 punctuation
The CoPar feature indicates the depth to which adjacent conjuncts are parallel.
The CoLenPar feature indicates the difference in length in adjacent conjuncts and whether this pair contains the last conjunct.

CoLenPar feature: (2,true) 6 words
Experimental results with all features

- Features must vary on parses of at least 5 sentences in training data
- In this experiment, 724,550 features
- Gaussian regularization, adjusted via cross-validation on section 23
- $f$-score on section 23 = 0.912 (15% error reduction over Charniak parser)
Conclusion

- It’s possible to build (moderately) accurate, broad-coverage parsers
- Generative parsing models are easy to estimate, but make questionable independence assumptions
- Exponential models don’t assume independence, so it’s easy to add new features, but are difficult to estimate
- Coarse-to-fine conditional MLE for exponential models is a compromise
  - flexibility of exponential models
  - possible to estimate from treebank data
- Gives the currently best-reported parsing accuracy results
Colorless green ideas sleep furiously.

Furiously sleep NNS JJ JJ

ideas green colorless
Sample parser errors

He will not be shaken out by external events, however surprising, alarming or vexing:

44
Soviet leaders said they would support their Kabul clients by all means necessary: -- and did.

Soviet leaders said they would support their Kabul clients by all means necessary: -- and did.
Kia is the most aggressive of the Korean Big Three in offering financing.
Two years ago, the district decided to limit the bikes to fire roads in its 65,000 hilly acres.
The company also pleased analysts by announcing four new store openings planned for fiscal 1990.

The company also pleased analysts by announcing four new store openings planned for fiscal 1990.
But funds generally are better prepared this time around.
The U.S. VP said it would fully support the resistance: -- and did n't.