The Infinite PCFG using Hierarchical Dirichlet Processes

EMNLP 2007 Prague, Czech Republic
June 29, 2007

Percy Liang Slav Petrov
Michael I. Jordan Dan Klein
How do we choose the grammar complexity?

Grammar induction:
How many grammar symbols (NP, VP, etc.)?

She heard the noise
How do we choose the grammar complexity?

Grammar induction:
How many grammar symbols (NP, VP, etc.)?

Grammar refinement:
How many grammar subsymbols (NP-loc, NP-subj, etc.)?

```
S-?
  NP-?
    PRP-?  VBD-?  NP-?
    She    heard    
       DT-?  NN-?
           the  noise
```
How do we choose the grammar complexity?

Grammar induction:
How many grammar symbols (NP, VP, etc.)?

Grammar refinement:
How many grammar subsymbols (NP-loc, NP-subj, etc.)?

Our solution: the HDP-PCFG allows number of (sub)symbols to adapt to data
A motivating example

True grammar:

$S \rightarrow A\ A \mid B\ B \mid C\ C \mid D\ D$

$A \rightarrow a_1 \mid a_2 \mid a_3$

$B \rightarrow b_1 \mid b_2 \mid b_3$

$C \rightarrow c_1 \mid c_2 \mid c_3$

$D \rightarrow d_1 \mid d_2 \mid d_3$
A motivating example

True grammar:

- **S** → **A A | B B | C C | D D**
- **A** → **a_1 | a_2 | a_3**
- **B** → **b_1 | b_2 | b_3**
- **C** → **c_1 | c_2 | c_3**
- **D** → **d_1 | d_2 | d_3**

Generate examples:

- **S**
  - **A A**
    - **a_2 a_3**
  - **S**
    - **B B**
      - **b_1 b_3**
  - **S**
    - **A A**
      - **a_1 a_1**
  - **S**
    - **C C**
      - **c_1 c_1**
A motivating example

True grammar:

\[
S \rightarrow A \, A \mid B \, B \mid C \, C \mid D \, D \\
A \rightarrow a_1 \mid a_2 \mid a_3 \\
B \rightarrow b_1 \mid b_2 \mid b_3 \\
C \rightarrow c_1 \mid c_2 \mid c_3 \\
D \rightarrow d_1 \mid d_2 \mid d_3
\]

Collapse A,B,C,D ⇒ X:

\[
\begin{align*}
S & \quad X \quad X \\
A & \quad a_2 \quad a_3 \\
B & \quad b_1 \quad b_3 \\
C & \quad a_1 \quad a_1 \\
D & \quad c_1 \quad c_1
\end{align*}
\]
A motivating example

True grammar:
S → A A | B B | C C | D D
A → a_1 | a_2 | a_3
B → b_1 | b_2 | b_3
C → c_1 | c_2 | c_3
D → d_1 | d_2 | d_3

Collapse A, B, C, D ⇒ X:

Results:

standard PCFG
A motivating example

True grammar:
\[ S \rightarrow A\ A \mid B\ B \mid C\ C \mid D\ D \]
\[ A \rightarrow a_1 \mid a_2 \mid a_3 \]
\[ B \rightarrow b_1 \mid b_2 \mid b_3 \]
\[ C \rightarrow c_1 \mid c_2 \mid c_3 \]
\[ D \rightarrow d_1 \mid d_2 \mid d_3 \]

Collapse \( A,B,C,D \Rightarrow X \):
\[ S \rightarrow S \mid S \mid S \mid S \]
\[ X \rightarrow X \mid X \mid X \mid X \]
\[ a_2 \mid a_3 \mid b_1 \mid b_3 \mid a_1 \mid a_1 \mid c_1 \mid c_1 \]

Results:

HDP-PCFG

subsymbol
0.25posterior
The meeting of two fields

Grammar learning

Lexicalized
[Charniak, 1996]
[Collins, 1999]

Manual refinement
[Johnson, 1998]
[Klein, Manning, 2003]

Automatic refinement
[Matsuzaki, et al., 2005]
[Petrov, et al., 2006]
The meeting of two fields

Grammar learning
Lexicalized
[Charniak, 1996]
[Collins, 1999]

Manual refinement
[Johnson, 1998]
[Klein, Manning, 2003]

Automatic refinement
[Matsuzaki, et al., 2005]
[Petrov, et al., 2006]

Bayesian nonparametrics

Basic theory
[Ferguson, 1973]
[Antoniak, 1974]
[Sethuraman, 1994]
[Escobar, West, 1995]
[Neal, 2000]

More complex models
[Teh, et al., 2006]
[Beal, et al., 2002]
[Goldwater, et al., 2006]
[Sohn, Xing, 2007]
The meeting of two fields

Grammar learning
- Lexicalized
  - Charniak, 1996
  - Collins, 1999
- Manual refinement
  - Johnson, 1998
  - Klein, Manning, 2003
- Automatic refinement
  - Matsuzaki, et al., 2005
  - Petrov, et al., 2006

Bayesian nonparametrics
- Basic theory
  - Ferguson, 1973
  - Antoniak, 1974
  - Sethuraman, 1994
  - Escobar, West, 1995
  - Neal, 2000
- More complex models
  - Teh, et al., 2006
  - Beal, et al., 2002
  - Goldwater, et al., 2006
  - Sohn, Xing, 2007

Nonparametric grammars
- Johnson, et al., 2006
- Finkel, et al., 2007
- Liang, et al., 2007
The meeting of two fields

Grammar learning
Lexicalized
[Charniak, 1996]
[Collins, 1999]

Bayesian nonparametrics
Basic theory
[Ferguson, 1973]
[Antoniak, 1974]

Our contribution
• Definition of the HDP-PCFG
• Simple and efficient variational inference algorithm
• Empirical comparison with finite models on a full-scale parsing task

Nonparametric grammars
[Johnson, et al., 2006]
[Finkel, et al., 2007]
[Liang, et al., 2007]
Bayesian paradigm

Generative model:

\[ \alpha \rightarrow \text{HDP} \rightarrow \theta \rightarrow \text{PCFG} \rightarrow z: \text{parse tree} \]
\[ x: \text{sentence} \]

hyperparameters \hspace{5em} grammar
Bayesian paradigm

Generative model:

\[ \alpha \rightarrow \text{HDP} \rightarrow \theta \rightarrow \text{PCFG} \rightarrow z: \text{parse tree} \]

\[ x: \text{sentence} \]

hyperparameters \hspace{2cm} grammar

Bayesian posterior inference:

Observe \( x \).
What's \( \theta \) and \( z \)?
HDP probabilistic context-free grammars

HDP-PCFG

β ∼ GEM(α) [generate distribution over symbols]

For each symbol \( z \in \{1, 2, \ldots \} \):

\( \phi^E_z ∼ \text{Dirichlet}(α^E) \) [generate emission probs]

\( \phi^B_z ∼ \text{DP}(α^B, ββ^T) \) [binary production probs]

For each nonterminal node \( i \):

\((z_{L(i)}, z_{R(i)}) ∼ \text{Multinomial}(φ^B_{z_i}) \) [child symbols]

For each preterminal node \( i \):

\( x_i ∼ \text{Multinomial}(φ^E_{z_i}) \) [terminal symbol]
HDP probabilistic context-free grammars

\[ \beta \sim \text{GEM}(\alpha) \] [generate distribution over symbols]
For each symbol \( z \in \{1, 2, \ldots \} \):
\[ \phi^E_z \sim \text{Dirichlet}(\alpha^E) \] [generate emission probs]
\[ \phi^B_z \sim \text{DP}(\alpha^B, \beta \beta^T) \] [binary production probs]
For each nonterminal node \( i \):
\[ (z_L(i), z_R(i)) \sim \text{Multinomial}(\phi^B_{z_i}) \] [child symbols]
For each preterminal node \( i \):
\[ x_i \sim \text{Multinomial}(\phi^E_{z_i}) \] [terminal symbol]
HDP-PCFG: prior over symbols

\[ \beta \sim \text{GEM}(\alpha) \]

\[ \alpha = 1 \rightarrow \text{GEM} \rightarrow \ldots \]
HDP-PCFG: prior over symbols

$\beta \sim \text{GEM}(\alpha)$

$\alpha = 1 \rightarrow \text{GEM} \rightarrow \ldots$
HDP-PCFG: prior over symbols

\[ \beta \sim \text{GEM}(\alpha) \]

\[ \alpha = 1 \quad \rightarrow \quad \text{GEM} \quad \rightarrow \quad \ldots \]
HDP-PCFG: prior over symbols

\[ \beta \sim \text{GEM}(\alpha) \]

\[ \alpha = 1 \quad \rightarrow \quad \text{GEM} \quad \rightarrow \quad \ldots \]
HDP-PCFG: prior over symbols

$$\beta \sim \text{GEM}(\alpha)$$

$$\alpha = 0.5 \quad \rightarrow \quad \text{GEM} \quad \rightarrow \quad \ldots$$

$$\alpha = 1 \quad \rightarrow \quad \text{GEM} \quad \rightarrow \quad \ldots$$
HDP-PCFG: prior over symbols

\[ \beta \sim \text{GEM}(\alpha) \]

\( \alpha = 0.2 \rightarrow \text{GEM} \rightarrow \)

\( \alpha = 0.5 \rightarrow \text{GEM} \rightarrow \)

\( \alpha = 1 \rightarrow \text{GEM} \rightarrow \)
HDP-PCFG: prior over symbols

\[ \beta \sim \text{GEM}(\alpha) \]

<table>
<thead>
<tr>
<th>( \alpha )</th>
<th>( \text{GEM} )</th>
<th>distribution</th>
</tr>
</thead>
</table>
| 0.2 |  | \[
\begin{array}{c}
0.2 \\
0.5 \\
1 \\
5 \\
10
\end{array}
\]
| 0.5 |  |  
| 1 |  |  
| 5 |  |  
| 10 |  |  

HDP probabilistic context-free grammars

\[ \beta \sim \text{GEM}(\alpha) \] [generate distribution over symbols]
For each symbol \( z \in \{1, 2, \ldots \} \):
\[ \phi_z^E \sim \text{Dirichlet}(\alpha^E) \] [generate emission probs]
\[ \phi_z^B \sim \text{DP}(\alpha^B, \beta\beta^T) \] [binary production probs]
For each nonterminal node \( i \):
\( (z_L(i), z_R(i)) \sim \text{Multinomial}(\phi_{z_i}^B) \) [child symbols]
For each preterminal node \( i \):
\( x_i \sim \text{Multinomial}(\phi_{z_i}^E) \) [terminal symbol]
HDP-PCFG: binary productions

Distribution over symbols (top-level):

\[ \beta \sim \text{GEM}(\alpha) \]

![Bar chart showing distribution of \( \beta \) with values at 1, 2, 3, 4, 5, 6, and a few more]
HDP-PCFG: binary productions

Distribution over symbols (top-level):

\[ \beta \sim \text{GEM}(\alpha) \]

Mean distribution over child symbols:

\[ \beta \beta^T \]
HDP-PCFG: binary productions

Distribution over symbols (top-level):

\[ \beta \sim \text{GEM}(\alpha) \]

\[ 1 \quad 2 \quad 3 \quad 4 \quad 5 \quad 6 \]

\[ \ldots \]

Mean distribution over child symbols:

\[ \beta \beta^T \]

Distribution over child symbols (per-state):

\[ \phi_z^B \sim \text{DP}(\alpha^B, \beta \beta^T) \]
HDP-PCFG: binary productions

Distribution over symbols (top-level):

\[ \beta \sim \text{GEM}(\alpha) \]

Mean distribution over child symbols:

\[ \beta \beta^T \]

Distribution over child symbols (per-state):

\[ \phi_z^B \sim \text{DP}(\alpha^B, \beta \beta^T) \]
HDP-PCFG: binary productions

Distribution over symbols (top-level):

\[ \beta \sim \text{GEM}(\alpha) \]

Mean distribution over child symbols:

\[ \beta\beta^T \]

Distribution over child symbols (per-state):

\[ \phi^B_z \sim \text{DP}(\alpha^B, \beta\beta^T) \]
HDP-PCFG: binary productions

Distribution over symbols (top-level):

\[ \mathbf{\beta} \sim \text{GEM}(\alpha) \]

\[ \begin{array}{cccccc}
1 & 2 & 3 & 4 & 5 & 6 \\
\end{array} \]

\[ \cdots \]

Mean distribution over child symbols:

\[ \mathbf{\beta} \mathbf{\beta}^T \]

Distribution over child symbols (per-state):

\[ \phi_{z}^B \sim \text{DP}(\alpha^B, \mathbf{\beta} \mathbf{\beta}^T) \]
HDP probabilistic context-free grammars

\[ \beta \sim \text{GEM}(\alpha) \] [generate distribution over symbols]

For each symbol \( z \in \{1, 2, \ldots \} \):
\[ \phi^E_z \sim \text{Dirichlet}(\alpha^E) \] [generate emission probs]
\[ \phi^B_z \sim \text{DP}(\alpha^B, \beta^T) \] [binary production probs]

For each nonterminal node \( i \):
\[ (z_{L(i)}, z_{R(i)}) \sim \text{Multinomial}(\phi^B_{zi}) \] [child symbols]

For each preterminal node \( i \):
\[ x_i \sim \text{Multinomial}(\phi^E_{zi}) \] [terminal symbol]
HDP probabilistic context-free grammars

HDP-PCFG

\( \beta \sim \text{GEM}(\alpha) \) [generate distribution over symbols]
For each symbol \( z \in \{1, 2, \ldots \} \):
\( \phi_z^E \sim \text{Dirichlet}(\alpha^E) \) [generate emission probs]
\( \phi_z^B \sim \text{DP}(\alpha^B, \beta\beta^T) \) [binary production probs]
For each nonterminal node \( i \):
\( (z_{L(i)}, z_{R(i)}) \sim \text{Multinomial}(\phi_{z_i}^B) \) [child symbols]
For each preterminal node \( i \):
\( x_i \sim \text{Multinomial}(\phi_{z_i}^E) \) [terminal symbol]
Variational Bayesian inference

\[
\alpha \xrightarrow{\text{HDP}} \theta \xrightarrow{\text{PCFG}} z: \text{parse tree}
\]
\[
\begin{aligned}
\text{hyperparameters} & & \text{grammar} \\
x: \text{sentence}
\end{aligned}
\]

Goal: compute posterior \( p(\theta, z \mid x) \)
Variational Bayesian inference

\[ \alpha \xrightarrow{\text{HDP}} \theta \xrightarrow{\text{PCFG}} z : \text{parse tree} \]
\[ \text{x: sentence} \]

Goal: compute posterior \( p(\theta, z | x) \)

Variational inference:
approximate posterior with best from a set of tractable distributions \( Q \):
\[
q^* = \arg\min_{q \in Q} KL(q || p)
\]

\[ Q \]
\[ q^* \]
\[ \cdots \]
\[ p \]
Variational Bayesian inference

**Goal:** compute posterior $p(\theta, z \mid x)$

**Variational inference:** approximate posterior with best from a set of tractable distributions $Q$:

$$q^* = \arg\min_{q \in Q} KL(q \parallel p)$$

**Mean-field approximation:**

$$Q = \left\{ q : q = q(z)q(\beta)q(\phi) \right\}$$

Diagram:

- $\alpha$ → HDP → $\theta$ → PCFG → $z$: parse tree
- $x$: sentence
- hyperparameters → grammar
- $Q$
- $q^*$
- $p$
Coordinate-wise descent algorithm

Goal: \( \text{argmin} \ KL(q \| p) \quad q \in Q \)

\[
q = q(z)q(\phi)q(\beta)
\]

\( z = \text{parse tree} \)
\( \phi = \text{rule probabilities} \)
\( \beta = \text{inventory of symbols} \)
Coordinate-wise descent algorithm

Goal: \( \arg\min_{q \in Q} KL(q \| p) \)
\[
q = q(z)q(\phi)q(\beta)
\]

\( z = \) parse tree
\( \phi = \) rule probabilities
\( \beta = \) inventory of symbols

Iterate:

- Optimize \( q(z) \) (E-step):
- Optimize \( q(\phi) \) (M-step):
- Optimize \( q(\beta) \) (no equivalent in EM):
Coordinate-wise descent algorithm

**Goal:** \( \text{argmin} \ KL(q || p) \)

\[ q = q(z)q(\phi)q(\beta) \]

\( z = \) parse tree
\( \phi = \) rule probabilities
\( \beta = \) inventory of symbols

**Iterate:**

- Optimize \( q(z) \) (E-step):
  - Inside-outside with rule weights \( W(r) \)
  - Gather expected rule counts \( C(r) \)

- Optimize \( q(\phi) \) (M-step):
- Optimize \( q(\beta) \) (no equivalent in EM):
Coordinate-wise descent algorithm

Goal: \( \arg\min_{q \in Q} KL(q \| p) \)

\( q = q(z)q(\phi)q(\beta) \)

\( z = \) parse tree
\( \phi = \) rule probabilities
\( \beta = \) inventory of symbols

Iterate:
- **Optimize** \( q(z) \) (E-step):
  - Inside-outside with rule weights \( W(r) \)
  - Gather expected rule counts \( C(r) \)
- **Optimize** \( q(\phi) \) (M-step):
  - Update Dirichlet posteriors (expected rule counts + pseudocounts)
  - Compute rule weights \( W(r) \)
- **Optimize** \( q(\beta) \) (no equivalent in EM):
Coordinate-wise descent algorithm

Goal: \[ \arg\min_{q \in Q} KL(q \| p) \]

\[ q = q(z)q(\phi)q(\beta) \]

\( z = \) parse tree
\( \phi = \) rule probabilities
\( \beta = \) inventory of symbols

Iterate:
- \textbf{Optimize} \( q(z) \) (E-step):
  - Inside-outside with rule weights \( W(r) \)
  - Gather expected rule counts \( C(r) \)
- \textbf{Optimize} \( q(\phi) \) (M-step):
  - Update Dirichlet posteriors
    (expected rule counts + pseudocounts)
  - Compute rule weights \( W(r) \)
- \textbf{Optimize} \( q(\beta) \) (no equivalent in EM):
  - Truncate at level \( K \)
    (set the maximum number of symbols)
  - Use projected gradient to adapt number of symbols
Rule weights

- Weight $W(r)$ of rule $r$ similar to probability $p(r)$
- $W(r)$ unnormalized $\Rightarrow$ extra degree of freedom
Rule weights

- Weight $W(r)$ of rule $r$ similar to probability $p(r)$
- $W(r)$ unnormalized $\Rightarrow$ extra degree of freedom

**EM (maximum likelihood):**

$$W(r) = \frac{C(r)}{\sum_{r'} C(r')}$$
Rule weights

• Weight $W(r)$ of rule $r$ similar to probability $p(r)$
• $W(r)$ unnormalized $\Rightarrow$ extra degree of freedom

EM (maximum likelihood):

$$W(r) = \frac{C(r)}{\sum_{r'} C(r')}$$

EM (maximum a posteriori):

$$W(r) = \frac{\text{prior}(r) - 1 + C(r)}{\sum_{r'} \text{prior}(r') - 1 + C(r')}$$
Rule weights

• Weight $W(r)$ of rule $r$ similar to probability $p(r)$
• $W(r)$ unnormalized $\Rightarrow$ extra degree of freedom

EM (maximum likelihood):

$$W(r) = \frac{C(r)}{\sum_{r'} C(r')}$$

EM (maximum a posteriori):

$$W(r) = \frac{\text{prior}(r)-1+C(r)}{\sum_{r'} \text{prior}(r')-1+C(r')}$$

Mean-field (with DP prior):

$$W(r) = \frac{\exp \Psi(\text{prior}(r)+C(r))}{\exp \Psi(\sum_{r'} \text{prior}(r')+C(r'))}$$
Rule weights

- Weight $W(r)$ of rule $r$ similar to probability $p(r)$
- $W(r)$ unnormalized $\Rightarrow$ extra degree of freedom

**EM (maximum likelihood):**

$$W(r) = \frac{C(r)}{\sum_{r'} C(r')}$$

**EM (maximum a posteriori):**

$$W(r) = \frac{\text{prior}(r) - 1 + C(r)}{\sum_{r'} \text{prior}(r') - 1 + C(r')}$$

**Mean-field (with DP prior):**

$$W(r) = \frac{\exp \Psi(\text{prior}(r) + C(r))}{\exp \Psi(\sum_{r'} \text{prior}(r') + C(r'))}$$
Rule weights

• Weight $W(r)$ of rule $r$ similar to probability $p(r)$
• $W(r)$ unnormalized $\Rightarrow$ extra degree of freedom

EM (maximum likelihood):

$$W(r) = \frac{C(r)}{\sum_{r'} C(r')}$$

EM (maximum a posteriori):

$$W(r) = \frac{\text{prior}(r) - 1 + C(r)}{\sum_{r'} \text{prior}(r') - 1 + C(r')}$$

Mean-field (with DP prior):

$$W(r) = \frac{\exp \Psi(\text{prior}(r) + C(r))}{\exp \Psi(\sum_{r'} \text{prior}(r') + C(r'))}$$

$$\approx \frac{C(r) - 0.5}{\sum_{r'} C(r') - 0.5}$$
Rule weights

- Weight $W(r)$ of rule $r$ similar to probability $p(r)$
- $W(r)$ unnormalized $\Rightarrow$ extra degree of freedom

EM (maximum likelihood):
$$W(r) = \frac{C(r)}{\sum_{r'} C(r')}$$

EM (maximum a posteriori):
$$W(r) = \frac{\text{prior}(r) - 1 + C(r)}{\sum_{r'} \text{prior}(r') - 1 + C(r')}$$

Mean-field (with DP prior):
$$W(r) = \frac{\exp \Psi(\text{prior}(r) + C(r))}{\exp \Psi(\sum_{r'} \text{prior}(r') + C(r'))}$$
$$\approx \frac{C(r) - 0.5}{\sum_{r'} C(r') - 0.5}$$

Subtract 0.5 $\Rightarrow$ small counts hurt more than large counts $\Rightarrow$ rich gets richer $\Rightarrow$ controls number of symbols
Parsing the WSJ Penn Treebank

Setup: grammar refinement (split symbols into subsymbols)
Training on one section:

<table>
<thead>
<tr>
<th>Maximum number of subsymbols (truncation $K$)</th>
<th>F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>76.0</td>
</tr>
<tr>
<td>9</td>
<td>92.0</td>
</tr>
<tr>
<td>12</td>
<td>84.0</td>
</tr>
<tr>
<td>16</td>
<td>92.0</td>
</tr>
<tr>
<td>20</td>
<td>68.0</td>
</tr>
</tbody>
</table>

Graph showing F1 score against maximum number of subsymbols (truncation $K$): Standard PCFG
Parsing the WSJ Penn Treebank

Setup: grammar refinement (split symbols into subsymbols)
Training on one section:

- F1 scores for maximum number of subsymbols (truncation $K$):
  - HDP-PCFG
  - Standard PCFG
Parsing the WSJ Penn Treebank

Setup: grammar refinement (split symbols into subsymbols)
Training on one section:

<table>
<thead>
<tr>
<th>maximum number of subsymbols (truncation $K$)</th>
<th>F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>68.0</td>
</tr>
<tr>
<td>9</td>
<td>76.0</td>
</tr>
<tr>
<td>12</td>
<td>84.0</td>
</tr>
<tr>
<td>16</td>
<td>92.0</td>
</tr>
<tr>
<td>20</td>
<td>100.0</td>
</tr>
</tbody>
</table>

- HDP-PCFG ($K = 20$)
- Standard PCFG

Graph showing the F1 score for different maximum number of subsymbols.
Parsing the WSJ Penn Treebank

Setup: grammar refinement (split symbols into subsymbols)

Training on one section:

5 9 ...

Training on 20 sections: 

Standard PCFG: 86.23

HDP-PCFG: 87.08

F_1

maximum number of subsymbols (truncation K)

Training on 20 sections: (K = 16)
Parsing the WSJ Penn Treebank

Setup: grammar refinement (split symbols into subsymbols)
Training on one section:

Results:
- HDP-PCFG overfits less than standard PCFG
- If have large amounts of data, HDP-PCFG \( \approx \) standard PCFG
Conclusions

- **What?** HDP-PCFG model allows number of grammar symbols to adapt to data
- **How?** Mean-field algorithm (variational inference) simple, efficient, similar to EM
- **When?** Have small amounts of data overfits less than standard PCFG
- **Why?** Declarative framework Grammar complexity specified declaratively in model rather than in learning procedure