Bayesian Models for Dependency Parsing
Using Pitman-Yor Priors

Hanna M. Wallach

University of Massachusetts Amherst
wallach@cs.umass.edu

July 8, 2009

Joint work with Charles Sutton and Andrew McCallum
Dependency Parsing

Dependency Parsing

Dependency trees encode syntactic relationships between words. Each node is a part-of-speech-tagged, cased word. An edge from word $w_m$ to $w_m'$ means $w_m'$ is a dependent of $w_m$. 

Bayesian Models for Dependency Parsing Using Pitman-Yor Priors

Hanna M. Wallach
Dependency Parsing

- Dependency trees encode syntactic relationships between words
- Each node is a part-of-speech-tagged, cased\textsuperscript{1} word
- An edge from word $w_m$ to $w_{m'}$ means $w_{m'}$ is a dependent of $w_m$

\textsuperscript{1}Cases: upper, lower, mixed, first capitalized word
talk outline

▶ Four hierarchical Bayesian dependency models:
  ▶ Bayesian reinterpretation of a classic dependency model
  ▶ Extension of this model using hierarchical Pitman-Yor priors
  ▶ Bayesian dependency model with “syntactic” states
  ▶ Bayesian dependency model with “semantic” states
Eisner’s Generative Dependency Model (Eisner '96)

- Conditioned on their parent, left and right children each form a first-order Markov chain
- Final child in each direction is a special stop symbol
- *Stop* symbols enable simultaneous generation of words and trees
Eisner’s Generative Dependency Model (Eisner '96)

- Conditioned on their parent, left and right children each form a first-order Markov chain
- Final child in each direction is a special stop symbol
- Stop symbols enable simultaneous generation of words and trees

root
Eisner’s Generative Dependency Model (Eisner ’96)

- Conditioned on their parent, left and right children each form a first-order Markov chain
- Final child in each direction is a special stop symbol
- Stop symbols enable simultaneous generation of words and trees
Eisner’s Generative Dependency Model (Eisner ’96)

- Conditioned on their parent, left and right children each form a first-order Markov chain.
- Final child in each direction is a special stop symbol.
- Stop symbols enable simultaneous generation of words and trees.

Diagram:
- root
- [v] hit
- [n] girl
Eisner’s Generative Dependency Model (Eisner '96)

- Conditioned on their parent, left and right children each form a first-order Markov chain
- Final child in each direction is a special stop symbol
- Stop symbols enable simultaneous generation of words and trees
Eisner’s Generative Dependency Model (Eisner ’96)

- Conditioned on their parent, left and right children each form a first-order Markov chain
- Final child in each direction is a special stop symbol
- Stop symbols enable simultaneous generation of words and trees
Eisner’s Generative Dependency Model (Eisner ’96)

- Conditioned on their parent, left and right children each form a first-order Markov chain
- Final child in each direction is a special *stop* symbol
- *Stop* symbols enable simultaneous generation of words and trees
Eisner’s Generative Dependency Model (Eisner ’96)

- Conditioned on their parent, left and right children each form a first-order Markov chain.
- Final child in each direction is a special stop symbol.
- *Stop* symbols enable simultaneous generation of words and trees.
Eisner’s Generative Dependency Model (Eisner '96)

- Conditioned on their parent, left and right children each form a first-order Markov chain
- Final child in each direction is a special *[stop]* symbol
- *[Stop]* symbols enable simultaneous generation of words and trees

---

Bayesian Models for Dependency Parsing Using Pitman-Yor Priors

Hanna M. Wallach
Eisner’s Generative Dependency Model (Eisner '96)

- Conditioned on their parent, left and right children each form a first-order Markov chain
- Final child in each direction is a special stop symbol
- Stop symbols enable simultaneous generation of words and trees
Eisner’s Generative Dependency Model (Eisner ’96)

- Conditioned on their parent, left and right children each form a first-order Markov chain.
- Final child in each direction is a special stop symbol.
- Stop symbols enable simultaneous generation of words and trees.
Generating a Tagged, Cased Word

- Generate a tag given the tagged, cased parent word and the sibling tag
- Generate an uncased word given the tagged, cased parent word and the just-generated tag
- Generate a case value given the just-generated tag and uncased word

Bayesian Models for Dependency Parsing Using Pitman-Yor Priors

Hanna M. Wallach
Generating a Tagged, Cased Word

- Generate a **tag** given the tagged, cased parent word and the sibling tag

Bayesian Models for Dependency Parsing Using Pitman-Yor Priors
Hanna M. Wallach
Generating a Tagged, Cased Word

- Generate a **tag** given the tagged, cased parent word and the sibling tag
- Generate an **uncased word** given the tagged, cased parent word and the just-generated tag
Generating a Tagged, Cased Word

- Generate a **tag** given the tagged, cased parent word and the sibling tag
- Generate an **uncased word** given the tagged, cased parent word and the just-generated tag
- Generate a case value given the just-generated tag and uncased word
Estimating Probabilities from Data

- Use a corpus $\mathcal{D} = \{s, w, c, t\}$ of tagged, cased sentences and trees
- Count relevant occurrences in $\mathcal{D}$, e.g.,
  \[ N_{c|sw} = \# \text{ times uncased word } w \text{ with tag } s \text{ has case value } c \]
- Use counts to form “estimators”, e.g., $N_{c|sw} / N_{.|sw}$
- The more specific the context, the sparser the counts
- “Smooth” more specific estimators with less specific ones
- Same approach as interpolated $n$-gram language modeling
Estimating Probabilities from Data

- Contexts are obvious in language modeling ($w_{n-2} w_{n-1} \rightarrow w_{n-1}$).
- Choice of contexts is much less obvious in parsing, e.g.,

  \[ \text{tag word} \rightarrow \text{tag} \quad \text{or} \quad \text{tag word} \rightarrow \text{word} \]

- Eisner estimates e.g., the case value probability as follows:

  \[
P(\text{case} = c \mid \text{tag} = s, \text{word} = w, \mathcal{D}) = \frac{N_{c|sw} + 3 \frac{N_{c|s} + 0.5 \frac{1}{C}}{N_{.|s} + 0.5}}{N_{.|sw} + 3}
\]
Contexts are obvious in language modeling ($w_{n-2} w_{n-1} \rightarrow w_{n-1}$)

Choice of contexts is much less obvious in parsing, e.g.,

\[ \text{tag word} \rightarrow \text{tag} \quad \text{or} \quad \text{tag word} \rightarrow \text{word} \]

Eisner estimates e.g., the case value probability as follows:

\[
P(\text{case} = c \mid \text{tag} = s, \text{word} = w, D) = \frac{N_{c|sw} + 3 \left( \frac{N_{c|s} + 0.5}{N_{.s} + 0.5} \right)}{N_{.|sw} + 3}
\]
Estimating Probabilities from Data

- Contexts are obvious in language modeling ($w_{n-2} w_{n-1} \rightarrow w_{n-1}$)
- Choice of contexts is much less obvious in parsing, e.g.,

  \[\text{tag word} \rightarrow \text{tag} \quad \text{or} \quad \text{tag word} \rightarrow \text{word}\]

- Eisner estimates e.g., the case value probability as follows:

  \[
P(\text{case} = c \mid \text{tag} = s, \text{word} = w, \mathcal{D}) = \frac{N_{c|sw} + 3 \left( \frac{N_{c|s}}{N_{.s}} + 0.5 \frac{1}{C} \right)}{N_{.|sw} + 3}
  \]
Contexts are obvious in language modeling \( (w_{n-2} w_{n-1} \rightarrow w_{n-1}) \)

Choice of contexts is much less obvious in parsing, e.g.,

\[
\text{tag word} \rightarrow \text{tag} \quad \text{or} \quad \text{tag word} \rightarrow \text{word}
\]

Eisner estimates e.g., the case value probability as follows:

\[
P(\text{case} = c \mid \text{tag} = s, \text{word} = w, \mathcal{D}) = \frac{N_{c|sw} + 3 \frac{N_{c|s} + 0.5 \frac{1}{C}}{N_{.|s} + 0.5}}{N_{.|sw} + 3}
\]
## Contexts for Tags and Uncased Words

\[
P(\text{tag} \mid \text{parent tagged cased word, sibling tag, dir})
\]

<table>
<thead>
<tr>
<th>parent tag</th>
<th>parent word</th>
<th>parent case</th>
<th>sibling tag</th>
<th>dir</th>
</tr>
</thead>
<tbody>
<tr>
<td>parent tag</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>parent tag</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>parent tag</td>
<td></td>
<td></td>
<td></td>
<td>dir</td>
</tr>
</tbody>
</table>

\[
P(\text{word} \mid \text{parent tagged cased word, dir})
\]

<table>
<thead>
<tr>
<th>tag</th>
<th>parent tag</th>
<th>parent word</th>
<th>parent case</th>
<th>dir</th>
</tr>
</thead>
<tbody>
<tr>
<td>tag</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>tag</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>tag</td>
<td></td>
<td>parent tag</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Bayesian Models for Dependency Parsing Using Pitman-Yor Priors

Hanna M. Wallach
We can redefine Eisner’s model from a Bayesian perspective. Treat each probability vector as a random variable, e.g.,

\[ \psi_{sw} = \text{distribution over case values given context } s \text{ and } w \]

Draw each probability vector from a Dirichlet prior, e.g.,

\[ \psi_{sw} \sim \text{Dir}(\psi_{sw}; \alpha_1, m_s) \]

\( m_s \) is a tag-specific base measure (distribution over case values).
Base Measures

- Base measures of Dirichlet priors, e.g., \( \{m_s\}_{s=1}^S \), are also unknown.
- Can also draw each \( m_s \) from a Dirichlet prior.
- Eisner: tag word \( \rightarrow \) tag \( \rightarrow \) uniform, so
  \[
  m_s \sim \text{Dir}(m_s; \alpha_0, u)
  \]
- This induces a \textit{hierarchical} Dirichlet prior over \( \psi_{sw} \).
- Can integrate out \( m_s \) and \( \psi_{sw} \) to obtain the \textit{predictive distribution}.
Predictive probability of case value $c$ is

$$P(\text{case} = c \mid \text{tag} = s, \text{word} = w, D, \alpha_1, \alpha_0) =$$

$$\frac{N_{c|sw} + \alpha_1 \frac{\hat{N}_{c|s} + \alpha_0 \frac{1}{C}}{\hat{N}_{.|s} + \alpha_0}}{N_{.|sw} + \alpha_1}$$

- Bottom-level counts $N_{c|sw}$ and $N_{.|sw}$ are raw observation counts
- Higher-level counts are not necessarily raw observation counts
Relationship to Eisner’s Model

- Compare Eisner’s probabilities with predictive distributions, e.g.,

**Eisner**

\[
P(c | s, w, D) = \frac{N_{c|sw} + 3 \frac{N_{c|s} + 0.5 \frac{1}{C}}{N_{.|s} + 0.5}}{N_{.|sw} + 3}
\]

**Bayesian**

\[
P(c | s, w, D, \alpha_1, \alpha_0) = \frac{N_{c|sw} + \alpha_1 \frac{\hat{N}_{c|s} + \alpha_0 \frac{1}{C}}{\hat{N}_{.|s} + \alpha_0}}{N_{.|sw} + \alpha_1}
\]

- Only differences: concentration parameters, higher-level counts
There are (at least) three ways of varying the Bayesian model:

1. Concentration parameters (e.g., $\alpha_1$, $\alpha_0$) can be sampled
   - Need not be arbitrarily chosen or set using cross validation
2. Counts need not correspond to observation counts
3. Can use priors other than the hierarchical Dirichlet distribution
   - e.g., the hierarchical Pitman-Yor process

All three variations have the potential to improve model quality
Can use Pitman-Yor priors instead of Dirichlet priors, e.g.,

\[ \psi_{sw} \sim \text{PY}(\psi_{sw} \mid \alpha_1, \epsilon_1, m_s) \]

\[ m_s \sim \text{PY}(m_s \mid \alpha_0, \epsilon_0, u) \]

- \( \epsilon_1 \) and \( \epsilon_0 \) are *discount* parameters
- When \( \epsilon_1 \) and \( \epsilon_0 \) are zero, identical to a Dirichlet distribution
- PY priors give distributions that better resemble natural language
  - Better at modeling rare words
Pitman-Yor Predictive Distributions

- Probability of case value $c$ is now given by:

$$P(\text{case} = c \mid \text{tag} = s, \text{word} = w, \mathcal{D}, \alpha_1, \alpha_0, \epsilon_1, \epsilon_0) =$$

$$\frac{M_{c|sw} + (\alpha_1 + \epsilon_1 L_{.|sw}) \hat{M}_{c|s} + (\alpha_0 + \epsilon_0 L_{.|s}) \frac{1}{C}}{\hat{N}_{.|s} + \alpha_0}$$

- Counts are now given by $M_{c|sw} = N_{c|sw} - \epsilon_1 L_{c|sw}$ etc.
Relationship to Bayesian $n$-gram Language Modeling

- PY priors have been used for language modeling (e.g., Teh ’06)
- Kneser-Ney smoothing is equivalent to
  - Setting concentration parameters ($\alpha$s) to zero
  - Using the “minimal path” approximate inference scheme
- Kneser-Ney smoothing is one of the best smoothing methods
- Dependency models are *lexicalised* (unlike, e.g., PCFGs)
- PY priors are particularly appropriate for dependency models
Using the Model

- Can now compute $P(D | \alpha_1, \alpha_0, \epsilon_1, \epsilon_0)$
- If this were language modeling, we'd be done

- For real sentences, only words, tags and case values are known
  - Goal: infer dependency trees for real sentences
- Use training data (sentences + trees) to learn the model
- Determine trees for test sentences
  - Sample trees using Metropolis-Hastings (Johnson et al. '07)
Parsing Experiments

- *Wall Street Journal* sections of Penn Treebank:
  - Training (sections 2–21): 39,832 sentences
  - Testing (section 23): 2,416 sentences

- Parse accuracy: percentage of parents correctly identified
- Maximum probability trees used for comparison purposes
- For efficiency, part-of-speech tags fixed to:
  - Training: “gold standard” tags from Treebank
  - Testing: tags from Ratnaparkhi’s tagger ('96)
Results: Parse Accuracy

PY prior, sampled hyperparameters: 26% error reduction over Eisner
Latent Variable Parsing Models

- Generative framework allows for inclusion of other latent variables
- E.g., “syntactic” and “semantic” topics
  - Specialized syntactic or semantic distributions over words
- Can define a (simpler) model that uses latent variables:
  - Sentences are untagged and uncased
  - Siblings are not taken into account (i.e., first order model)
  - Distribution over children depends on parent and latent state variable
First Order Models

- Computationally more efficient than models that consider siblings
- Children are independent of each other given their parent

\[ P(\text{girl hit . with the bat the ball}) = P(\text{the girl hit the ball with the bat}) \]
“Syntactic” Latent Variables

\[ \text{Generate a state given the parent word} \]
\[ \text{Generate a word given the just-generated state and the parent word} \]

hit [state] ball

Bayesian Models for Dependency Parsing Using Pitman-Yor Priors
Hanna M. Wallach
"Syntactic" Latent Variables

- Generate a state given the parent word

...[state]...hit...[state]...ball...
“Syntactic” Latent Variables

- Generate a **state** given the parent word
- Generate a **word** given the just-generated state and the parent word

Bayesian Models for Dependency Parsing Using Pitman-Yor Priors
Hanna M. Wallach
“Syntactic” Latent Variables

- Generate a **state** given the parent word
- Generate a **word** given the just-generated state and the parent word

Give all probability vectors Dirichlet priors as before
"Semantic" Latent Variables

- Generate a state given the document, as in LDA.
- Generate a word given the just-generated state and the parent word.

```
hit
ball

<doc>

[state]
```

Bayesian Models for Dependency Parsing Using Pitman-Yor Priors

Hanna M. Wallach
“Semantic” Latent Variables

- Generate a state given the document, as in LDA

Bayesian Models for Dependency Parsing Using Pitman-Yor Priors  
Hanna M. Wallach
“Semantic” Latent Variables

- Generate a state given the document, as in LDA
- Generate a word given the just-generated state and the parent word

Diagram:
```
<doc>
  hit
    ...
    ...
<state>
  ball
```
“Semantic” Latent Variables

- Generate a **state** given the *document*, as in LDA
- Generate a **word** given the just-generated state and the parent word
- Give all probability vectors Dirichlet priors
<table>
<thead>
<tr>
<th>position</th>
<th>name</th>
<th>location</th>
<th>action</th>
<th>state</th>
<th>type</th>
<th>count</th>
</tr>
</thead>
<tbody>
<tr>
<td>president</td>
<td>u.s.</td>
<td>made</td>
<td>is</td>
<td>would</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>director</td>
<td>california</td>
<td>offered</td>
<td>are</td>
<td>will</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>officer</td>
<td>washington</td>
<td>filed</td>
<td>was</td>
<td>could</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>chairman</td>
<td>texas</td>
<td>put</td>
<td>has</td>
<td>should</td>
<td>50</td>
<td></td>
</tr>
<tr>
<td>executive</td>
<td>york</td>
<td>asked</td>
<td>have</td>
<td>can</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>head</td>
<td>london</td>
<td>approved</td>
<td>were</td>
<td>might</td>
<td>15</td>
<td></td>
</tr>
<tr>
<td>attorney</td>
<td>japan</td>
<td>announced</td>
<td>will</td>
<td>had</td>
<td>20</td>
<td></td>
</tr>
<tr>
<td>manager</td>
<td>canada</td>
<td>left</td>
<td>had</td>
<td>may</td>
<td>30</td>
<td></td>
</tr>
<tr>
<td>chief</td>
<td>france</td>
<td>held</td>
<td>’s</td>
<td>must</td>
<td>25</td>
<td></td>
</tr>
<tr>
<td>secretary</td>
<td>britain</td>
<td>bought</td>
<td>would</td>
<td>owns</td>
<td>3</td>
<td></td>
</tr>
</tbody>
</table>
Unsupervised Leave-One-Out Bits-Per-Word

<table>
<thead>
<tr>
<th>Model</th>
<th>Bits-per-Word</th>
</tr>
</thead>
<tbody>
<tr>
<td>LDA</td>
<td>9.08</td>
</tr>
<tr>
<td>Deps. only</td>
<td>8.75</td>
</tr>
<tr>
<td>Deps. &amp; Syntactic Topics</td>
<td>8.68</td>
</tr>
<tr>
<td>Deps. &amp; Semantic Topics</td>
<td>8.25</td>
</tr>
</tbody>
</table>

- The fewer the bits-per-word, the better the model
Conclusions and Future Work

- Reinterpreted a classic dependency parser using Bayesian framework
- Parsing performance is improved by:
  - Using Pitman-Yor priors
  - Sampling hyperparameters
- Can incorporate latent variables into the model:
  - Syntactic topics that cluster parent–child relationships
  - Semantic topics, as in LDA
- Future work: syntactic + semantic topics
Questions?

wallach@cs.umass.edu

http://www.inference.phy.cam.ac.uk/hmw26/