Structured Prediction Models via the Matrix-Tree Theorem

Terry Koo maestro@csail.mit.edu
Amir Globerson gamir@csail.mit.edu
Xavier Carreras carreras@csail.mit.edu
Michael Collins mcollins@csail.mit.edu

MIT Computer Science and Artificial Intelligence Laboratory
Dependency parsing

- Syntactic structure represented by head-modifier dependencies
Projective vs. non-projective structures

- Non-projective structures allow *crossing dependencies*

- Frequent in languages like Czech, Dutch, etc.

- Non-projective parsing is max-spanning-tree (McDonald et al., 2005)
Contributions of this work

- Fundamental inference algorithms that sum over possible structures:

<table>
<thead>
<tr>
<th>Model type</th>
<th>Inference Algorithm</th>
</tr>
</thead>
<tbody>
<tr>
<td>HMM</td>
<td>Forward-Backward</td>
</tr>
<tr>
<td>Graphical Model</td>
<td>Belief Propagation</td>
</tr>
<tr>
<td>PCFG</td>
<td>Inside-Outside</td>
</tr>
<tr>
<td>Projective Dep. Trees</td>
<td>Inside-Outside</td>
</tr>
<tr>
<td>Non-projective Dep. Trees</td>
<td>??</td>
</tr>
</tbody>
</table>

- This talk:
  - *Inside-outside-style* algorithms for non-projective dependency structures
  - An application: training log-linear and max-margin parsers
  - Independently-developed work: Smith and Smith (2007), McDonald and Satta (2007)
Overview

- Background
- Matrix-Tree-based inference
- Experiments
Edge-factored structured prediction

A dependency tree $y$ is a set of head-modifier dependencies (McDonald et al., 2005; Eisner, 1996)

- $(h, m)$ is a dependency with feature vector $f(x, h, m)$
- $\mathcal{Y}(x)$ is the set of all possible trees for sentence $x$

$$y^* = \arg\max_{y \in \mathcal{Y}(x)} \sum_{(h, m) \in y} w \cdot f(x, h, m)$$
Training log-linear dependency parsers

- Given a training set \( \{(x_i, y_i)\}_{i=1}^{N} \), minimize

\[
L(w) = \frac{C}{2} \|w\|^2 - \sum_{i=1}^{N} \log P(y_i | x_i; w)
\]
Training log-linear dependency parsers

- Given a training set \( \{(x_i, y_i)\}_{i=1}^{N} \), minimize

\[
L(w) = \frac{C}{2} \|w\|^2 - \sum_{i=1}^{N} \log P(y_i \mid x_i; w)
\]

- Log-linear distribution over trees

\[
P(y \mid x; w) = \frac{1}{Z(x; w)} \exp \left\{ \sum_{(h,m) \in y} w \cdot f(x, h, m) \right\}
\]

\[
Z(x; w) = \sum_{y \in \mathcal{Y}(x)} \exp \left\{ \sum_{(h,m) \in y} w \cdot f(x, h, m) \right\}
\]
Training log-linear dependency parsers

- Gradient-based optimizers evaluate $L(w)$ and $\frac{\partial L}{\partial w}$

\[
L(w) = \frac{C}{2} \|w\|^2 - \sum_{i=1}^{N} \sum_{(h,m) \in y_i} w \cdot f(x_i, h, m)
+ \sum_{i=1}^{N} \log Z(x_i; w)
\]

- Main difficulty: computation of the \textit{partition functions}
Training log-linear dependency parsers

- Gradient-based optimizers evaluate $L(w)$ and $\frac{\partial L}{\partial w}$

$$
\frac{\partial L}{\partial w} = Cw - \sum_{i=1}^{N} \sum_{(h,m) \in y_i} f(x_i, h, m)
+ \sum_{i=1}^{N} \sum_{h', m'} P(h' \rightarrow m' | x; w)f(x_i, h', m')
$$

- The *marginals* are edge-appearance probabilities

$$
P(h \rightarrow m | x; w) = \sum_{y \in Y(x) : (h,m) \in y} P(y | x; w)
$$
Generalized log-linear inference

- Vector $\theta$ with parameter $\theta_{h,m}$ for each dependency

$$P(y \mid x; \theta) = \frac{1}{Z(x; \theta)} \exp \left\{ \sum_{(h,m) \in y} \theta_{h,m} \right\}$$

$$Z(x; \theta) = \sum_{y \in \mathcal{Y}(x)} \exp \left\{ \sum_{(h,m) \in y} \theta_{h,m} \right\}$$

$$P(h \rightarrow m \mid x; \theta) = \frac{1}{Z(x; \theta)} \sum_{y \in \mathcal{Y}(x) : (h,m) \in y} \exp \left\{ \sum_{(h,m) \in y} \theta_{h,m} \right\}$$

- E.g., $\theta_{h,m} = w \cdot f(x, h, m)$
Applications of log-linear inference

- Generalized inference engine that takes $\theta$ as input
  - Different definitions of $\theta$ can be used for log-linear or max-margin training

\[
\mathbf{w}_{LL}^* = \arg\min_{\mathbf{w}} \left[ \frac{C}{2} \| \mathbf{w} \|^2 - \sum_{i=1}^{N} \log P(y_i | x_i; \mathbf{w}) \right]
\]

\[
\mathbf{w}_{MM}^* = \arg\min_{\mathbf{w}} \left[ \frac{C}{2} \| \mathbf{w} \|^2 + \sum_{i=1}^{N} \max_y \left( E_{i,y} - m_{i,y}(\mathbf{w}) \right) \right]
\]

- Exponentiated-gradient updates for max-margin models
  - Globerson, Koo, Carreras and Collins (2007)
Overview

- Background
  - Matrix-Tree-based inference
- Experiments
Single-root vs. multi-root structures

- Multi-root structures allow *multiple* edges from *. 

- Single-root structures have *exactly one* edge from *. 

- Independent adaptations of the Matrix-Tree Theorem: Smith and Smith (2007), McDonald and Satta (2007)
Matrix-Tree Theorem (Tutte, 1984)

Given:

1. Directed graph $G$
2. Edge weights $\theta$
3. A node $r$ in $G$

A matrix $L^{(r)}$ can be constructed whose determinant is the sum of weighted spanning trees of $G$ rooted at $r$.
Matrix-Tree Theorem (Tutte, 1984)

- Given:
  1. Directed graph $G$
  2. Edge weights $\theta$
  3. A node $r$ in $G$

A matrix $L^{(r)}$ can be constructed whose determinant is the sum of weighted spanning trees of $G$ rooted at $r$

$$\sum = \exp\{2 + 4\}$$
Matrix-Tree Theorem (Tutte, 1984)

- Given:
  1. Directed graph $G$
  2. Edge weights $\theta$
  3. A node $r$ in $G$

A matrix $L^{(r)}$ can be constructed whose determinant is the sum of weighted spanning trees of $G$ rooted at $r$

$$\sum = \exp \{2 + 4\} + \exp \{1 + 3\}$$
Matrix-Tree Theorem (Tutte, 1984)

- Given:
  1. Directed graph \( G \)
  2. Edge weights \( \theta \)
  3. A node \( r \) in \( G \)

A matrix \( L^{(r)} \) can be constructed whose determinant is the sum of weighted spanning trees of \( G \) **rooted at** \( r \)

\[
\sum = \exp \{2 + 4\} + \exp \{1 + 3\} = \det(L^{(1)})
\]
Multi-root partition function

- Edge weights $\theta$, root $r = 0$

- $\det(L^{(0)}) = \text{non-projective } \textit{multi-root} \text{ partition function}$
Construction of $L^{(0)}$

- $L^{(0)}$ has a simple construction
  
  **off-diagonal:**
  \[ L^{(0)}_{h,m} = - \exp \{ \theta_{h,m} \} \]

  **on-diagonal:**
  \[ L^{(0)}_{m,m} = \sum_{h'=0}^{n} \exp \{ \theta_{h,m} \} \]

- E.g., $L^{(0)}_{3,3}$

- The determinant of $L^{(0)}$ can be evaluated in $O(n^3)$ time
Single-root vs. multi-root structures

- Multi-root structures allow *multiple* edges from *
- Single-root structures have *exactly one* edge from *
- Independent adaptations of the Matrix-Tree Theorem: Smith and Smith (2007), McDonald and Satta (2007)
Single-root partition function

- Naïve method for computing the single-root non-projective partition function
Single-root partition function

- Naïve method for computing the single-root non-projective partition function

- Exclude all root edges except $(0, 1)$

- Computing $n$ determinants requires $O(n^4)$ time
Single-root partition function

- Naïve method for computing the single-root non-projective partition function

- Exclude all root edges except (0, 2)

- Computing $n$ determinants requires $O(n^4)$ time
Single-root partition function

- Naïve method for computing the single-root non-projective partition function

- Exclude all root edges except \((0, 3)\)

- Computing \(n\) determinants requires \(O(n^4)\) time
An alternate matrix $\hat{L}$ can be constructed such that $\det(\hat{L})$ is the single-root partition function

- **first row:** $\hat{L}_{1,m} = \exp \{ \theta_{0,m} \}$
- **other rows, on-diagonal:** $\hat{L}_{m,m} = \sum_{h'=1}^{n} \exp \{ \theta_{h,m} \}$
- **other rows, off-diagonal:** $\hat{L}_{h,m} = - \exp \{ \theta_{h,m} \}$

**Single-root partition function requires** $O(n^3)$ time
Non-projective marginals

- The log-partition generates the marginals

\[ P(h \rightarrow m \mid x; \theta) = \frac{\partial \log Z(x; \theta)}{\partial \theta_{h,m}} = \frac{\partial \log \det(\hat{L})}{\partial \theta_{h,m}} = \sum_{h',m'} \frac{\partial \log \det(\hat{L})}{\partial \hat{L}_{h',m'}} \frac{\partial \hat{L}_{h',m'}}{\partial \theta_{h,m}} \]

- Derivative of log-determinant:

\[ \frac{\partial \log \det(\hat{L})}{\partial \hat{L}} = (\hat{L}^{-1})^T \]

- Complexity dominated by matrix inverse, \( O(n^3) \)
Summary of non-projective inference

- Partition function: matrix determinant, $O(n^3)$
- Marginals: matrix inverse, $O(n^3)$
- Single-root inference: $\hat{L}$
- Multi-root inference: $L^{(0)}$
Overview

- Background
- Matrix-Tree-based inference
  - Experiments
Log-linear and max-margin training

- **Log-linear training**

\[
w_{LL}^* = \arg\min_w \left[ \frac{C}{2} \|w\|^2 - \sum_{i=1}^N \log P(y_i | x_i; w) \right]
\]

- **Max-margin training**

\[
w_{MM}^* = \arg\min_w \left[ \frac{C}{2} \|w\|^2 + \sum_{i=1}^N \max_y (E_{i,y} - m_{i,y}(w)) \right]
\]
Multilingual parsing experiments

- Six languages from CoNLL 2006 shared task
- Training algorithms: averaged perceptron, log-linear models, max-margin models
- Projective models vs. non-projective models
- Single-root models vs. multi-root models
Multilingual parsing experiments

<table>
<thead>
<tr>
<th>Dutch (4.93%cd)</th>
<th>Projective Training</th>
<th>Non-Projective Training</th>
</tr>
</thead>
<tbody>
<tr>
<td>Perceptron</td>
<td>77.17</td>
<td>78.83</td>
</tr>
<tr>
<td>Log-Linear</td>
<td>76.23</td>
<td>79.55</td>
</tr>
<tr>
<td>Max-Margin</td>
<td>76.53</td>
<td>79.69</td>
</tr>
</tbody>
</table>

- Non-projective training helps on non-projective languages
Multilingual parsing experiments

<table>
<thead>
<tr>
<th>Language</th>
<th>Projective Training</th>
<th>Non-Projective Training</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Spanish</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(0.06% cd)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Perceptron</td>
<td>81.19</td>
<td>80.02</td>
</tr>
<tr>
<td>Log-Linear</td>
<td>81.75</td>
<td>81.57</td>
</tr>
<tr>
<td>Max-Margin</td>
<td>81.71</td>
<td>81.93</td>
</tr>
</tbody>
</table>

- Non-projective training doesn’t hurt on projective languages
Multilingual parsing experiments

- Results across all 6 languages (Arabic, Dutch, Japanese, Slovene, Spanish, Turkish)

<table>
<thead>
<tr>
<th>Model</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>Perceptron</td>
<td>79.05</td>
</tr>
<tr>
<td>Log-Linear</td>
<td>79.71</td>
</tr>
<tr>
<td>Max-Margin</td>
<td>79.82</td>
</tr>
</tbody>
</table>

- Log-linear and max-margin parsers show improvement over perceptron-trained parsers
  - Improvements are statistically significant (sign test)
Summary

- Inside-outside-style inference algorithms for non-projective structures
  - Application of the Matrix-Tree Theorem
  - Inference for both multi-root and single-root structures

- Empirical results
  - Non-projective training is good for non-projective languages
  - Log-linear and max-margin parsers outperform perceptron parsers
Thanks!

Thanks for listening!
Thanks!
Challenges for future research

- State-of-the-art performance is obtained by higher-order models (McDonald and Pereira, 2006; Carreras, 2007)

- Higher-order non-projective inference is nontrivial (McDonald and Pereira, 2006; McDonald and Satta, 2007)

- Approximate inference may work well in practice

- Reranking of $k$-best spanning trees (Hall, 2007)