Joint Learning of Syntactic and Semantic Dependencies

Xavier Lluís and Lluís Màrquez

Barcelona, 10 de setembre de 2008
The CoNLL-2008 shared task

- Syntactic dependency parsing
- Predicate identification and disambiguation
- Semantic role labeling
1. Introduction
2. Difficulties
3. Joint model
4. Results and discussion
Talk Overview

1. Introduction
   - Syntactic and semantic parsing
   - Design a joint model

2. Difficulties

3. Joint model

4. Results and discussion
Talk Overview

1. Introduction
   - Syntactic and semantic parsing
   - Design a joint model

2. Difficulties

3. Joint model

4. Results and discussion
syntactic and semantic parsing

A sample sentence

-Root- it completed a previously announced acquisition
syntactic and semantic parsing

Syntactic dependencies

- Root
- it
- completed
- a
- previously
- announced
- acquisition

SBJ

OBJ

OBJ

AMOD

NMOD
syntactic and semantic parsing: semantics

Predicate completed
Syntactic and semantic parsing: semantics

Semantic dependencies for completed
Syntactic and semantic parsing

Predicate acquisition
syntactic and semantic parsing: semantics

Semantic dependencies for acquisition
syntactic and semantic parsing: semantics
syntactic and semantic parsing: semantics

Semantic dependencies for announced
Syntactic and semantic parsing: semantics

Semantic dependencies for all predicates
Mainstream approach

1. Syntactic parsing
   - A parser (Eisner, Shift-reduce)

2. Semantic role labeling
   - A simpler (non-structured) classifier
The pipeline approach

1. Propagation or amplification of errors
2. Assumes an order of increasing difficulty
3. Dependencies between layers are hard to be captured
Design a joint model

- One single step for syntax + semantics
- Musillo and Merlo (2006) used extended tags (e.g., “NMODA0”)
Design a joint model

1. Can a joint system overcome the pipeline approach?
2. Is it feasible to build a joint parsing system?
Talk Overview

1. Introduction
   - Syntactic and semantic parsing
   - Design a joint model

2. Difficulties

3. Joint model

4. Results and discussion
A joint approach

Extend a syntactic parsing model to **jointly** parse semantics

1. **Syntactic parsing**
   - A parser (**Eisner**, Shift-reduce)

2. **Semantic role labeling**
   - A simpler (non-structured) classifier
A joint approach

Extend the **Eisner** algorithm *jointly* parse semantics

- $O(n^3)$ algorithm
- Based on CKY algorithm
- Bottom-up parser
The Eisner algorithm

--Root-- it completed a previously announced acquisition

Bottom-up dependency parsing
The Eisner algorithm

- Root- it completed a previously announced acquisition

Bottom-up dependency parsing
The Eisner algorithm

- Root-
  - SBJ
  - it completed
  - NMOD
  - a previously announced acquisition

Bottom-up dependency parsing
The Eisner algorithm

Bottom-up dependency parsing
The Eisner algorithm

Design a joint model

Bottom-up dependency parsing
The Eisner algorithm

Bottom-up dependency parsing
Design a joint model

The Eisner algorithm

Bottom-up dependency parsing
Score of a dependency

A dependency $d = \langle h, m, l \rangle$ of a sentence $x$ is scored by:

$$\text{score}(d, x) = \phi(\langle h, m, l \rangle, x) \cdot w$$

where $\phi$ is a feature extraction function, $w$ is a weight vector
Design a joint model

The Eisner algorithm

Score of a tree

A syntactic tree $y$ for a sentence $x$ is scored by:

$$\text{score}_{\text{tree}}(y, x) = \sum_{\langle h, m, l \rangle \in y} \text{score}(\langle h, m, l \rangle, y)$$

Arc-factorization

The sum of independent scores for each dependency of the tree is called the first order factorization.
The Eisner algorithm

**Best tree**

We are interested in the best scoring tree among all trees $\mathcal{Y}(x)$:

$$\text{best_tree}(x) = \arg\max_{y \in \mathcal{Y}(x)} \text{score_tree}(y, x)$$

**Eisner algorithm**

The Eisner algorithm is an **exact** search algorithm that computes the best first-order factorized tree.
The Eisner algorithm

A non-projective dependency graph
Joint parsing point of view

Joint parsing implies the prediction of the semantic label \textit{simultaneously} to the syntactic label.
Talk Overview

1. Introduction

2. Difficulties
   - Syntactic and semantic overlap
   - Unavailable features

3. Joint model

4. Results and discussion
Talk Overview

1. Introduction

2. Difficulties
   - Syntactic and semantic overlap
   - Unavailable features

3. Joint model

4. Results and discussion
Joint Approach Difficulties: an example

Syntactic and semantic overlap
Joint Approach Difficulties: an example

Syntactic and semantic overlap
Joint Approach Difficulties: an example
Joint Approach Difficulties: an example

Syntactic and semantic overlap

Joint model

Difficulties

Results and discussion

Future work
1. Are syntax and semantics overlapping?

- **36.4%** of argument-predicate relations do **not** exactly overlap with modifier-head syntactic relations.

**Proposed solution**

Bound the semantic label to the syntactic dependency
Difficulties: non-overlapping semantics

Solution to overlapping dependencies
Difficulties: non-overlapping semantics

Solution to overlapping dependencies
Difficulties: non-overlapping semantics

Solution to overlapping dependencies
Difficulties: non-overlapping semantics

Solution to overlapping dependencies
Difficulties: non-overlapping semantics

Solution to overlapping dependencies
Difficulties: non-overlapping semantics

Solution to overlapping dependencies
Difficulties: non-overlapping semantics

Solution

An extended dependency is:

\[ d = \langle h, m, l_{syn}, l_{sem\ p_1}, \ldots, l_{sem\ p_q} \rangle \]

- \( h \) is the head
- \( m \) the modifier
- \( l_{syn} \) the syntactic label
- \( l_{sem\ p_i} \) one semantic label for each sentence predicate \( p_i \)
A dependency has its syntactic and semantic labels
**Talk Overview**

1. **Introduction**

2. **Difficulties**
   - Syntactic and semantic overlap
   - Unavailable features

3. **Joint model**

4. **Results and discussion**
Unavailable features

Proposed solution

Unavailable features for distant arguments
Proposed solution

Unavailable features for distant arguments
Unavailable features for distant arguments
Proposed solution

Unavailable features for distant arguments
Proposed solution

Unavailable features

Unavailable features for distant arguments
Unavailable features

Proposed solution

Unavailable features for distant arguments
Unavailable features

Proposed solution

Unavailable features for distant arguments
Proposed solution

Unavailable features for distant arguments
2. Problems not appearing in pipeline systems

- State-of-the-art SRL systems strongly rely on syntactic path features.
- There is only a partial visibility of the syntax restricted to the current sentence span.
- A distant argument-predicate relation can occur.

Proposed solution

Pre-parse and extract predicate-modifier syntactic paths.
## Talk Overview

1. **Introduction**
2. **Difficulties**
3. **Joint model**
   - Formalization
   - Learning
4. **Results and discussion**
Talk Overview

1. Introduction

2. Difficulties

3. Joint model
   - Formalization
   - Learning

4. Results and discussion
Joint Model

The **Joint Model** extends and it is based on the first order syntactic model.

**Best joint tree**

$$\text{best_tree}(x, w, y') = \arg\max_{y \in \mathcal{Y}(x)} \text{score_tree}(y, x, w, y')$$

argmax computed using the Eisner algorithm

- $x$ is the input sentence
- $y$ is the syntactic-semantic tree
- $y'$ pre-parsed syntactic tree
- $w$ is the weight vector
Joint Model

**First order factorization**

\[
\text{score}_\text{tree}(y, x, w, y') = \sum_{\langle h, m, l_{\text{syn}}, l \rangle \in y} \text{score}(\langle h, m, l_{\text{syn}}, l \rangle, x, w, y')
\]

- \(x\) is the input sentence
- \(y\) is the syntactic-semantic tree
- \(y'\) pre-parsed syntactic tree
- \(w\) is the weight vector
- \(l = l_{\text{sem} p_1}, \ldots, l_{\text{sem} p_q}\) are the semantic labels for predicates \(p_i\)
Scoring

\[
\text{score}\left(\langle h, m, l_{\text{syn}}, l \rangle, x, w, y'\right) = \\
\text{syntactic\_score}\left(h, m, l_{\text{syn}}, x, w\right) + \\
\text{semantic\_score}\left(h, m, l_{\text{sem} \, p_1}, \ldots, l_{\text{sem} \, p_q}, x, w, y'\right)
\]

The score of a dependency is the \textit{syntactic score} (as usual) + the \textit{semantic score} of the assigned semantic label (if any) \textit{for each predicate}

\[l = l_{\text{sem} \, p_1}, \ldots, l_{\text{sem} \, p_q}\]
Semantic Scoring

**Semantic scoring function**

\[
\text{semantic\_score} \left( h, m, l_{sem\ p_1}, \ldots, l_{sem\ p_q}, x, w, y' \right) = \\
\sum_{l_{sem\ p_i}} \phi_{sem} \left( \langle h, m, l_{sem\ p_i} \rangle, p_i, x, y' \right) \cdot w^{(l_{sem\ p_i})}
\]

- \( y' \) is the precomputed syntax tree for feature extraction
- \( l_{sem\ p_i} \) is the semantic label of \( m \) for predicate \( p_i \)

**Example**

- Root
- SBj, A0, ... A0
- it completed a previously announced acquisition
- complete.01 announce.01 acquisition.01
Talk Overview

1. Introduction
2. Difficulties
3. Joint model
   - Formalization
   - Learning
4. Results and discussion
The structured perceptron algorithm

\[ w \leftarrow 0 \]

\[ \text{for } t \leftarrow 1 \text{ to } \text{numEpochs} \text{ do} \]

\[ \text{for all } (x, y) \in \text{training set do} \]

\[ \hat{y} = \text{best\_tree}(x, w) \]

\[ \text{for all factor } \in y \setminus \hat{y} \text{ do} \]

\[ w^{(l)} \leftarrow w^{(l)} + \phi(\text{factor}, x, \hat{y}) \]

\[ \text{end for} \]

\[ \text{for all factor } \in \hat{y} \setminus y \text{ do} \]

\[ w^{(l)} \leftarrow w^{(l)} - \phi(\text{factor}, x, \hat{y}) \]

\[ \text{end for} \]

\[ \text{end for} \]

Collins, 2002
Core

Averaged perceptron learning + Eisner algorithm inference

Collins, 2002
Eisner, 1996

and based on Carreras et al. 2006

Joint inference

Extension of the Eisner algorithm with semantic labels
System overview

System architecture

1. Preprocess
   - Feature extraction

2. Syntactic parsing
   - To allow syntactic feature extraction

3. Predicate identification
   - Heuristic rules + SVM

4. Joint syntactic-semantic parsing
   - The core of the system

5. Postprocess
   - Most frequent predicate sense and ILP
System architecture

1. **Preprocess**
   - Feature extraction

2. **Syntactic parsing**
   - To allow syntactic feature extraction

3. **Predicate identification**
   - Heuristic rules + SVM

4. **Joint syntactic-semantic parsing**
   - The core of the system

5. **Postprocess**
   - Most frequent predicate sense and ILP
System architecture

1. **Preprocess**
   - Feature extraction

2. **Syntactic parsing**
   - To allow syntactic feature extraction

3. **Predicate identification**
   - Heuristic rules + SVM

4. **Joint syntactic-semantic parsing**
   - The core of the system

5. **Postprocess**
   - Most frequent predicate sense and ILP
## System architecture

1. **Preprocess**
   - Feature extraction

2. **Syntactic parsing**
   - To allow syntactic feature extraction

3. **Predicate identification**
   - Heuristic rules + SVM

4. **Joint syntactic-semantic parsing**
   - The core of the system

5. **Postprocess**
   - Most frequent predicate sense and ILP
System overview

System architecture

1. Preprocess
   - Feature extraction
2. Syntactic parsing
   - To allow syntactic feature extraction
3. Predicate identification
   - Heuristic rules + SVM
4. Joint syntactic-semantic parsing
   - The core of the system
5. Postprocess
   - Most frequent predicate sense and ILP
Talk Overview

1. Introduction
2. Difficulties
3. Joint model
4. Results and discussion
   - Experimentation
   - CoNLL-2008 results
Talk Overview

1. Introduction
2. Difficulties
3. Joint model
4. Results and discussion
   - Experimentation
   - CoNLL-2008 results
The CoNLL-2008 shared task

Datasets
- Syntactic parsing
  - WSJ corpus with ~1M tokens corpus.
- Semantic role labeling
  - Nominal and verbal predicates

Evaluation
- Syntax  Labeled attachment score (LAS)
- Semantics  Precision, Recall, $F_1$
- Global  Macro averaged $F_1$
Discussion

1. *Can a joint system overcome the pipeline approach?*

2. *Is it feasible to build a joint parsing system?*
Learning curve (development)

- **Pipe**: ~-0.1
- **Syntax**: ~-0.1
- **Semantics**: ~+5
Answers to the questions

Discussion

1. **Can a joint system overcome the pipeline approach?**
   - **Syntactic parsing** similar results
   - **Semantic parsing** improved by $\sim5$ points
   - Semantic results are better and improving (and hurting syntax?)

2. **Is it feasible to build a joint parsing system?**
   - **Syntactic pipeline parser** $\sim0.2$ s/sentence
   - **Joint parser** $\sim0.3$ s/sentence
   - **Joint parser memory** $<1.5$ GB

   65% of time computing the averaged perceptron.
   *1 epoch in less than 4h. on an amd64 x2 5000+*
Syntactic-Semantic Overlap

Learning curve (development)

- Pipe
- Joint
- Syntax: ~-0.1
- Joint syn+sem
- Semantics: ~+5

epochs

score (semantic F1, F1, LAS)
About syntactic-semantic overlapping

- **A semantic relation highly dependent on the correct syntactic head:**
  - will increase the correct dependency score
  - will benefit the final syntax tree

- **A semantic relation not so dependent on the correct syntactic head:**
  - will increase some dependencies scores
  - could hurt the final syntax tree
syntactic and semantic parsing: semantics

Semantic scoring
syntactic and semantic parsing: semantics

Semantic scoring
syntactic and semantic parsing: semantics
syntactic and semantic parsing: semantics

Semantic scoring
syntactic and semantic parsing: semantics

Semantic scoring
syntactic and semantic parsing: semantics

Semantic scoring
syntactic and semantic parsing: semantics

Semantic scoring
syntactic and semantic parsing: semantics

Semantic scoring
Discussion

Semantic scoring function

\[
\text{semantic\_score} \left( h, m, l_{sem\ p_i}, x, w, y' \right) = \\
\phi_{sem} \left( h, m, p_i, x, y' \right) \cdot w^{(l_{sem\ p_i})}
\]

Features are extracted by \( \phi_{sem} \) from:

- \( h \) head
- \( m \) modifier
- \( p_i \) predicate
- \( h, m \) modifier-head
- \( m, p_i \) modifier-predicate
- \( h, p_i \) head-predicate
## Posteval results

<table>
<thead>
<tr>
<th>Group</th>
<th>Name</th>
<th>WSJ + Brown</th>
<th>WSJ</th>
<th>Brown</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lund (*)</td>
<td>Johansson (*)</td>
<td>85.49</td>
<td>86.61</td>
<td>76.34</td>
</tr>
<tr>
<td>Yahoo! (*)</td>
<td>Ciaramita (*)</td>
<td>82.69</td>
<td>83.83</td>
<td>73.51</td>
</tr>
<tr>
<td>HIT-IR</td>
<td>Che</td>
<td>82.66</td>
<td>83.78</td>
<td>73.57</td>
</tr>
<tr>
<td>Hong Kong (*)</td>
<td>Zhao (*)</td>
<td>82.24</td>
<td>83.41</td>
<td>72.70</td>
</tr>
<tr>
<td>Geneva (*)</td>
<td>Henderson (*)</td>
<td>80.48</td>
<td>81.53</td>
<td>71.93</td>
</tr>
<tr>
<td>Koc</td>
<td>Yuret</td>
<td>79.84</td>
<td>80.97</td>
<td>70.55</td>
</tr>
<tr>
<td>GSLT ML2</td>
<td>Samuelsson</td>
<td>79.79</td>
<td>80.92</td>
<td>70.49</td>
</tr>
<tr>
<td>DFKI 2</td>
<td>Zhang</td>
<td>79.32</td>
<td>80.41</td>
<td>70.48</td>
</tr>
<tr>
<td>NAIST</td>
<td>Watanabe</td>
<td>79.10</td>
<td>80.3</td>
<td>69.29</td>
</tr>
<tr>
<td>Antwerp</td>
<td>Morante</td>
<td>78.43</td>
<td>79.52</td>
<td>69.55</td>
</tr>
<tr>
<td>HIT-ICR</td>
<td>Li</td>
<td>78.35</td>
<td>79.38</td>
<td>70.01</td>
</tr>
<tr>
<td>UPC (*)</td>
<td>Lluís (*)</td>
<td>78.11</td>
<td>79.16</td>
<td>69.84</td>
</tr>
<tr>
<td>UT Austin</td>
<td>Baldridge</td>
<td>77.49</td>
<td>78.57</td>
<td>68.53</td>
</tr>
<tr>
<td>Koc</td>
<td>Yatbaz</td>
<td>77.45</td>
<td>78.43</td>
<td>69.61</td>
</tr>
<tr>
<td>USTC</td>
<td>Chen</td>
<td>77.00</td>
<td>77.95</td>
<td>69.23</td>
</tr>
<tr>
<td>Korea</td>
<td>Lee</td>
<td>76.90</td>
<td>77.96</td>
<td>68.34</td>
</tr>
<tr>
<td>Peking</td>
<td>Sun</td>
<td>76.28</td>
<td>77.1</td>
<td>69.58</td>
</tr>
<tr>
<td>Colorado</td>
<td>Choi</td>
<td>71.23</td>
<td>72.22</td>
<td>63.44</td>
</tr>
<tr>
<td>UAIC</td>
<td>Trandabat</td>
<td>63.45</td>
<td>64.21</td>
<td>57.41</td>
</tr>
<tr>
<td>DFKI 1</td>
<td>Neumann</td>
<td>19.93</td>
<td>20.13</td>
<td>18.14</td>
</tr>
</tbody>
</table>
Future and Ongoing Work

1. Higher degree of joint processing
   - Joint predicate identification
   - No previous dependency parsing
2. Higher order dependencies
3. Improvement of the semantic classifier component
4. Projectivization techniques
5. Feature engineering and system tuning
6. Alternative joint models
The end

Thank you
For further reading

James Henderson, Paola Merlo, Gabriele Musillo and Ivan Titov
*A Latent Variable Model of Synchronous Parsing for Syntactic and Semantic Dependencies.*

Gabriele Musillo and Paola Merlo
*Robust Parsing for the Proposition Bank.*

Xavier Lluís and Lluís Màrquez
*A Joint Model for Parsing Syntactic and Semantic Dependencies.*