Hybrid Learning of Dependency Structures from Heterogeneous Linguistic Resources

Yi Zhang†‡, Rui Wang‡, Hans Uszkoreit†‡

†Language Technology Lab
German Research Center for Artificial Intelligence, Germany

‡Department of Computational Linguistics
Saarland University, Germany

CoNLL 2008
Manchester, UK
Outline

1 Motivation & Background
2 Syntactic Dependency Parsing
3 Semantic Role Labeling
4 Results
Outline

1. Motivation & Background
2. Syntactic Dependency Parsing
3. Semantic Role Labeling
4. Results
Hybrid Processing with Heterogeneous Language Resources

- Benefit from different parsing models
  - Graph-based approach
  - Transition-based approach
- Utilize different language resources
  - Data-driven statistical parser
  - Symbolic Grammar-based parser
Deep Linguistic Processing

- Rich Formalism
- Linguistically motivated analysis
- Semantically informed outputs
- State-of-the-art Deep parsers (HPSG, LFG, TAG, CCG, ...)
  - Accurate
  - Efficient
  - Robust
Architecture

- Transition–based DepParser (MaltParser)
- Graph–based DepParser (MST Parser)
- Parse Selector
- Predicate Identification
- Argument Identification
- Argument Classification
- Predicated Classification
- Semantic Role Labeling
- Syn.Dep.

Zhang, Wang, Uszkoreit

Hybrid Learning Dependency Structures
Outline

1. Motivation & Background
2. Syntactic Dependency Parsing
3. Semantic Role Labeling
4. Results
Graph-based vs. Transition-based Syntactic Dependency Parsing

- Transition-based approach: MaltParser [Nivre et al., 2007]
  - Projective, Arc-Eager, SVM with polynomial kernel (d=2)
- Graph-based approach: MSTParser [McDonald et al., 2005]
  - Projective (Eisner), with 2nd order features

Outputs of different parsing models are complementary [McDonald and Nivre, 2007, Nivre and McDonald, 2008]

Combine two parser outputs with a ME-based voting model
Outline

1. Motivation & Background
2. Syntactic Dependency Parsing
3. Semantic Role Labeling
4. Results
ME-based Pipeline SRL model

- Predicate Identification
  - Binary classification for each word
- Argument Identification
  - Given predicate, binary classification for each argument candidate
- Argument Classification
  - Multi-class classification for each argument
- Predicate Classification
  - Ranking role-set for known ambiguous predicates (or xx.01 otherwise)
Basic Features

- **P lemma:** lost
- **P POS:** VBN
- **P rel:** VC
- **P-parent POS:** VBZ
- **A rel:** SBJ
- **P-children POSes:** PRP RB
- **P-children rels:** OBJ MNR
- **P-A path:** [ VC | SBJ ]
- **A-children POSes:** -
- **A-children rels:** -
- **P > A ?:** false
- **A’s position:** 1/3
- **P-siblings POSes:** PRP
- **P-siblings rels:** SBJ
Minimal Recursion Semantics

- A semantic representation with underspecifiability
- Well suited as syntactic-semantic interface for compositional semantics in grammar development
- Has predicate-argument backbone
MRS Features

trouble, has just as quickly

she, lost

it, as

P MRS ep-name: _lose_v_1_rel
P MRS-args labels: ARG1 ARG2
P MRS-args POSes: PRP PRP
A MRS ep-name: pron_rel
A MRS-preds labels: ARG1
A MRS-preds POSes: VBZ
## Features: Overview

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>PI</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
</tr>
<tr>
<td>AI</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
</tr>
<tr>
<td>AC</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
</tr>
<tr>
<td>PC</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
</tr>
</tbody>
</table>

Zhang, Wang, Uszkoreit

Hybrid Learning Dependency Structures
Argument Candidate Selection with P-A Path

- Given a predicate, not every word in the sentence can be candidate argument
- [Hacioglu, 2004] suggests to use “family” dependency nodes of the predicate node as candidates
- Use predicate-argument path patterns (as chain of dep. rels) to select candidates
Argument Candidate Selection with P-A Path: Example

Trouble is, she has lost it just as quickly.

```
trouble, she, has
```

```
trouble

is

SBJ, P

```

```
has

PRD, P

```

```
lost

SBJ, VC

```

```
she

```

```
lost

OBJ, MNR

```

```
it

```

```
just

```

```
as

```

```
quickly

```

```
```

Zhang, Wang, Uszkoreit

Hybrid Learning Dependency Structures
Argument Candidate Selection with P-A Path: Example

Trouble is, she has lost it just as quickly.

```
lost(A0:she): [VC|SBJ]
```

Zhang, Wang, Uszkoreit

Hybrid Learning Dependency Structures
Argument Candidate Selection with P-A Path: Example

Trouble is, she has lost it just as quickly.

lost(A1:it): [OBJ]

Zhang, Wang, Uszkoreit

Hybrid Learning Dependency Structures
Argument Candidate Selection with P-A Path: Example

Trouble is, she has lost it just as quickly.

lost(AM-MNR:just): [\text{[MNR]}]
Argument Candidate Selection with P-A Path: Statistics

Avg. #Candidate per P vs. AI Coverage

- P-A Path freq. threshold 40
- 272 patterns
- AI coverage upper bound 95%
- Avg. candidate 5.76 per P
Outline

1. Motivation & Background
2. Syntactic Dependency Parsing
3. Semantic Role Labeling
4. Results
Setup

- MaltParser & MSTParser
- TADM for ME parameter estimation
- PET parser + English Resource Grammar (HPSG)
  - Unknown word handling
  - Partial Parsing
  - Average parsing speed: ~3 seconds per sentence
- The SRL module takes ~1 hour to train, and less than 1 minute runtime on WSJ test
### Results

<table>
<thead>
<tr>
<th></th>
<th>In-Domain</th>
<th></th>
<th>Out-Domain</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Syntactic Dep.</td>
<td>88.14%</td>
<td>90.78%</td>
<td>80.80%</td>
<td>86.12%</td>
</tr>
<tr>
<td>SRL Closed</td>
<td>72.67%</td>
<td>82.68%</td>
<td>60.16%</td>
<td>76.98%</td>
</tr>
<tr>
<td>SRL Open</td>
<td>73.08%</td>
<td>83.04%</td>
<td>62.11%</td>
<td>78.48%</td>
</tr>
</tbody>
</table>

Zhang, Wang, Uszkoreit

Hybrid Learning Dependency Structures
### Results

<table>
<thead>
<tr>
<th></th>
<th>In-Domain</th>
<th></th>
<th>Out-Domain</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Syntactic Dep.</td>
<td>88.14%</td>
<td>90.78%</td>
<td>80.80%</td>
<td>86.12%</td>
</tr>
<tr>
<td>SRL Closed</td>
<td>72.67%</td>
<td>82.68%</td>
<td>60.16%</td>
<td>76.98%</td>
</tr>
<tr>
<td>SRL Open</td>
<td>73.08%</td>
<td>83.04%</td>
<td>62.11%</td>
<td>78.48%</td>
</tr>
</tbody>
</table>

Zhang, Wang, Uszkoreit

Hybrid Learning Dependency Structures
Conclusion

- Combined syntactic dependency parsing model delivers improved results
- SRL benefits significant improvement from deep parsing outputs, especially in out-domain test
For Further Reading I

Semantic role labeling using dependency trees.

Characterizing the errors of data-driven dependency parsing models.
In *Proceedings of the 2007 joint conference on empirical methods in natural language processing and computational natural language learning (EMNLP-CoNLL)*, pages 122–131, Prague, Czech Republic.

Non-Projective Dependency Parsing using Spanning Tree Algorithms.
For Further Reading II

Integrating graph-based and transition-based dependency parsers.
In *Proceedings of ACL-08: HLT*, pages 950–958, Columbus, Ohio.

Maltparser: A language-independent system for data-driven dependency parsing.