Structural Correspondence Learning for Parse Disambiguation

Barbara Plank
b.plank@rug.nl

University of Groningen (RUG), The Netherlands
EACL 2009 - Student Research Workshop

April 2, 2009
The Problem: Domain dependence

A very common problem/situation in NLP:

- Train a model on data you have; test it, works pretty good
- However, whenever **test** and **training data** differ, the performance of such a supervised system **degrades** considerably (Gildea, 2001)
The Problem: Domain dependence

A very common problem/situation in NLP:

- Train a model on data you have; test it, works pretty good
- However, whenever **test** and **training data** differ, the performance of such a supervised system **degrades** considerably (Gildea, 2001)

Possible solutions:

1. Build a model for every domain we encounter → Expensive!
2. Adapt a model from a *source* domain to a *target* domain → **Domain Adaptation**
Approaches to Domain Adaptation

Recently gained attention - Approaches:
Introduction and Motivation

Approaches to Domain Adaptation

Recently gained attention - Approaches:

a. Supervised Domain Adaptation

- Limited annotated resources in new domain
  (Gildea, 2001; Chelba and Acero, 2004; Hara, 2005; Daume III, 2007)

b. Semi-supervised Domain Adaptation

- No annotated resources in new domain (more difficult, but also more realistic)
  (McClosky et al., 2006): Self-training
  (Blitzer et al., 2006): Structural Correspondence Learning

→ This talk: semi-supervised scenario and parse disambiguation
Introduction and Motivation

Motivation

Structural Correspondence Learning (SCL) for Parse Disambiguation

1. Effectiveness of SCL rather unexplored for Parsing
   - SCL shown to be effective for PoS tagging and Sentiment analysis (Blitzer et al., 2006; Blitzer et al., 2007)
   - Attempt by Shimizu and Nakagawa (2007) in CoNLL 2007; inconclusive

2. Adaptation of Disambiguation Models - less studied area
   - Most previous work on parser adaptation for data-driven systems (i.e. systems employing treebank grammars)
   - Few studies on adapting disambiguation models (Hara, 2005; Plank and van Noord, 2008) focused exclusively on the supervised case
Background: Alpino Parser

- Wide-coverage dependency parser for Dutch

HPSG-style grammar rules, large hand-crafted lexicon

Maximum Entropy Disambiguation Model:
  - Feature functions $f_j$ / weights $w_j$
  - Estimation based on *Informative samples* (Osborne, 2000)

$$p_\theta(\omega|s; w) = \frac{1}{Z_\theta} q_0 \exp\left(\sum_{j=1}^{m} w_j f_j(\omega)\right)$$

Output: Dependency Structure
Structural Correspondence Learning (SCL) - Idea

- Domain adaptation algorithm for feature based classifiers, proposed by Blitzer et al. (2006)
- Use data **from both source and target domain** to induce correspondences among features from different domains
- Incorporate correspondences as new features in the labeled data of the source domain

Learn $w, v$ on source data:

$$x \cdot w + (x \cdot \theta) \cdot v$$
Structural Correspondence Learning (SCL) - Idea

Hypothesis:
If we find good correspondences, then labeled data from source domain will help us building a good classifier for the target domain.

Find correspondences through pivot features:

\[ \text{feat}_X \leftrightarrow \text{pivot feature} \leftrightarrow \text{feat}_Y \]

\[ \text{domain A} \quad \text{("linking" feature)} \quad \text{domain B} \]

Pivot features:
- Common features that occur frequently in both domains
- There should be sufficient features
- Should align well with the task at hand
Step 1: Choose $m$ pivot features

Our instantiation:

- First parse the unlabeled data (Blitzer uses only word-level features); possibly noisy but more abstract representation of the data
- Features are properties of parses ($r_1$: grammar rules, $s_1$: syntactic features, apposition, dependency relations, $p_1$: coordination, etc.)
- **Selection of pivot features**: features (of type $r_1,p_1,s_1$) whose count is $> t$, with $t = 5000$ (on average $m = 360$ pivots)
SCL algorithm - Step 2/4

Step 2: Train pivot predictors

- Train $m$ binary classifiers, one for each pivot feature: “Does pivot feature $l$ occur in this instance?”
- Mask pivot feature and try to predict it using other non-pivot features
- In this way estimate weight vector $w_l$ for pivot feature $l$:
  - Positive weight entries in $w_l$ mean a non-pivot feature is highly correlated with the corresponding pivot
  - Each pivot predictor implicitly aligns non-pivot features from source & target domains
Step 3: Dimensionality reduction

- Arrange the weight vectors in matrix $W$.
- $W^T \cdot x$ would give $m$ features (too many)
- Compute Singular value decomposition (SVD) on $W$:

$$W = \begin{bmatrix} \theta & U & D \end{bmatrix} = \begin{bmatrix} \theta_{n \times h} & U_{n \times n} & D_{n \times m} \end{bmatrix}$$

$\theta = \text{top } h \text{ of } U^T$

- Use top left singular vectors $\theta = U^T_{1:h,:}$ (parametrized by $h$)
Step 4: Train a new model on augmented data

- Add new features to source data by applying: $\theta \cdot x$

$$\begin{bmatrix} \theta \\ n \times 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ n \times 1 \end{bmatrix} = \text{new Features}$$

- Train classifier (estimate $w, v$) on augmented source data:

$$w \cdot x + v \cdot (\theta \cdot x)$$
Experimental design

Data

- General, out-of-domain: Alpino (newspaper text; 145k tokens)
- Domain-specific: Wikipedia articles

Construction of target data from Wikipedia (WikiXML)

- Exploit Wikipedia’s category system (XQuery, Xpath): extract pages related to \( p \) (through sharing a direct, sub- or super category)

Overview of collected unlabeled target data:

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Size</th>
<th>Relationship</th>
</tr>
</thead>
<tbody>
<tr>
<td>Prince</td>
<td>290 articles, 145k tokens</td>
<td>filtered super</td>
</tr>
<tr>
<td>Pope Johannes Paulus II</td>
<td>445 articles, 134k tokens</td>
<td>all</td>
</tr>
<tr>
<td>De Morgan</td>
<td>394 articles, 133k tokens</td>
<td>all</td>
</tr>
</tbody>
</table>

Evaluation metric: Concept Accuracy (labeled dependency accuracy)
Experiments & Results

<table>
<thead>
<tr>
<th></th>
<th>Accuracy</th>
<th>Error red.</th>
</tr>
</thead>
<tbody>
<tr>
<td>baseline Prince</td>
<td>85.03</td>
<td>-</td>
</tr>
<tr>
<td>SCL, $h = 25$</td>
<td>85.12</td>
<td>2.64</td>
</tr>
<tr>
<td>SCL, $h = 50$</td>
<td>85.29</td>
<td>7.29</td>
</tr>
<tr>
<td>SCL, $h = 100$</td>
<td>85.19</td>
<td>4.47</td>
</tr>
<tr>
<td>baseline DeMorgan</td>
<td>80.09</td>
<td>-</td>
</tr>
<tr>
<td>SCL, $h = 25$</td>
<td>80.15</td>
<td>1.88</td>
</tr>
<tr>
<td>baseline Paus</td>
<td>85.72</td>
<td>-</td>
</tr>
<tr>
<td>SCL, $h = 25$</td>
<td>85.87</td>
<td>4.52</td>
</tr>
</tbody>
</table>

- Parser normally operates on an accuracy level of 88-89% (newspaper text)
- SCL: small but consistent increase in accuracy
- $h$ parameter little effect
- Work in progress

Table: Result of our instantiation of SCL
Results obtained without additional operation on feature level (as in Blitzer (2006)):

- Normalization & rescaling
- Feature-specific regularization
- Block SVDs
Additional Empirical Result

Block SVD

- Apply Dimensionality Reduction by feature type
- Standard setting of Blitzer et al. (2006) (based on Ando & Zhang (2005))

Idea: Apply Dimensionality Reduction by feature type


W:

- Pivot predictors
  - \( f_{rl} \)
  - \( f_{apapos} \)
- Nonpivots
  - Feature type submatrix
    - \( f_{dep} \)
    - ...

Graph: Comparison of baseline, SCL, and SCL block with SCL - no split and SCL - block SVD (9 feature types)
Conclusions

- Novel application of SCL for parse disambiguation
- Our first instantiation of SCL gives promising initial results
- SCL slightly but constantly outperformed the baseline
- Applying SCL involves many design choices and practical issues
- Examined self-training (not in paper): SCL outperforms self-training

Future work

a. Further explore/refine SCL (other testsets, varying amount of target domain data, pivot selection, etc.)

b. Other ways to exploit unlabeled data (e.g. more 'direct' mapping between features?)
Conclusions and Future Work

Thank you for your attention.