Subdomain Sensitive Statistical Parsing using Raw Corpora

Barbara Plank\textsuperscript{1} and Khalil Sima’an\textsuperscript{2}

\textsuperscript{1} Alfa Informatica, Faculty of Arts
University of Groningen, The Netherlands
b.plank@rug.nl

\textsuperscript{2} Language and Computation, Faculty of Science
University of Amsterdam, The Netherlands
simaan@science.uva.nl

LREC 2008
Marrakech, Morocco
Outline

1. Introduction and Motivation
2. Subdomain Sensitive Statistical Parsing using Raw Corpora
   - Subdomain Sensitive Parsers
   - Parser Combination Techniques
3. Experiments and Results
4. Conclusions and Future Work
Problem: Ambiguity of natural language sentences

Common approach: Train a parser/model on a treebank. Apply to new input.

Variations: phrase/dependency structure, formal grammar, statistical model and estimator.
Motivation

Is there more in a treebank that we might exploit?

- We view a treebank as a mixture of subdomains, each addressing certain concepts more than others. "politics, stock market, financial news etc. can be found in the WSJ" (Kneser and Peters, 1997)
- The parsing statistics gathered from the treebank are averages over different subdomains,
- Averages smooth out the differences between subdomains and weaken the biases

1. Do subdomains matter?
2. How to incorporate subdomain sensitivity into an existing state-of-the-art parser?
Motivation - Our Approach

Subdomains \( \{c_i\} \) as hidden features

\[
P(s, t) = \sum_i P(s, c_i)P(t|s, c_i)
\]  

This work: approximate it by creating an ensemble of parsers

Assumptions:

- We know a set of subdomains \( \{c_i, \ldots, c_k\} \)
- Approximate \( \sum_i \) by combining predictions of subdomains parsers
Overview and Problem Statement

Sub-domain driven parsing - Schema

(1) How to create domain-dependent parsers?

(2) How to combine them?
Creating subdomain-specific parsers

Weight the trees in treebank $TB$ with subdomain statistics

- Use domain-dependent raw corpus $C$ (flat sentences)
- Induce statistical Language Model (LM) $\theta$ from $C$
- Assign a count $f$ to every tree $\pi_i \in TB$ such that:
  \[ f = \text{average per-word “count” of yield } y_{[\pi_i]} \text{ under LM } \theta \]

Retrain parser on subdomain-weighted $TB_\theta$. 
Overview of our approach - Details

Introduction and Motivation

Subdomain Sensitive Statistical Parsing

Subdomain Sensitive Parsers

Parser Combination Techniques

Experiments and Results

Conclusions and Future Work

Overview of our approach - Details

Tree weighting (exploit raw corpora)

Parser Baseline

Parsed sentences (Parser Baseline)

New, unlabeled test sentences

Parsed sentences (combined)

Parsed sentences (Parser Sports)

Parsed sentences (Parser Financial)

Parsed sentences (Parser Politics)

Train parsers

Apply parsers

Combine output

Corpus Sports

Corpus Financial

Corpus Politics

Labeled examples
Penn Treebank WSJ

6x
2x
8x
3x

10x
1x
8x
5x

3x
9x
4x
Parser Combination Techniques

How to combine them?

Parser pre-selection

Test sentence \( s \)

Parser \( 1 \)

Parser \( 2 \)

Parser ... 

Parser \( k \)

Parser post-selection

Parse tree \( \pi(s) \)
Parser Combination Techniques

How to combine them?

Parser Pre-selection: selecting a parser up-front (given: $s$)

Parser Post-selection: selecting a parser after parsing (given: $s, t$)
Pre-selection: Divergence Model (DVM)

We measure for every word how well it discriminates between the subdomains using the notion of divergence. The \textit{divergence} of a word \( w \) in a subdomain \( i \in [1 \ldots k] \), from all other \( (k-1) \) subdomains \( (j \in [1 \ldots k], j \neq i) \):

\[
divergence_i(w) = 1 + \frac{\sum_{j \neq i} |\log \frac{p_{\theta_i}(w)}{p_{\theta_j}(w)}|}{(k-1)} \quad (2)
\]

\[
divergence_{\text{sent}}_i(w^n) = \frac{\sum_{x=1}^{n} divergence_i(w_x)}{n} \quad (3)
\]

Boundary issues:

- if \( p_{\theta_i}(w) = 0 \) then \( divergence_i(w) = 1 \), and
- if \( p_{\theta_j}(w) = 0 \), then \( p_{\theta_j}(w) = 10^{-15} \) (constant).
Pre-selection: Divergence Model (DVM) - Example

For example, 'multi-million-dollar' (score FINANCIAL domain: 5.5), 'equal' (score all domains from 1.6 to 1.9)
Post-Selection: Node Weighting + DVM (NW-DVM)

For parse tree $\pi_i$ with $1 \leq i \leq k$ and sentence $w_1^n$:

$$\text{score}(c) = \left[ \frac{1}{k} \sum_{i=1}^{k} \delta[c, \pi_i] \right]$$

$$(4)$$

$$\text{score}(\pi_i) = (1-\lambda) \left[ \frac{1}{|\pi_i|} \sum_{c \in \pi_i} \text{score}(c) \right] + \lambda \text{divergence}_{\text{sent};i}(w_1^n)$$

$$(5)$$

where $|\pi_i|$ is the size of the constituent set, and $0 < \lambda < 1$ an interpolation factor.

- How well does the parse tree $\pi_i$ fit the domain?
- How well does $w_1^n$ fit the domain?
First Experiment: Variance among Parsers

- Are subdomain parsers complementary?
- Optimal decision procedure - an oracle:

\[ \pi_{best\_oracle} = \arg\max_i f_{\text{F-score}}(\pi_i) \]
First Experiment: Variance among Parsers

- Are subdomain parsers complementary?
- Optimal decision procedure - an oracle:

\[
\pi_{best\_oracle} = \arg\max_i f_{F\text{-}score}(\pi_i)
\]  \hspace{1cm} (6)

<table>
<thead>
<tr>
<th>Parser</th>
<th>LR</th>
<th>LP</th>
<th>F-score</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section 00 (development set)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Baseline</td>
<td>89.44</td>
<td>89.63</td>
<td>89.53</td>
</tr>
<tr>
<td>Sports</td>
<td>88.95</td>
<td>88.83</td>
<td>88.89</td>
</tr>
<tr>
<td>Financial</td>
<td>89.01</td>
<td>88.84</td>
<td>88.92</td>
</tr>
<tr>
<td>Politics</td>
<td>88.86</td>
<td>88.70</td>
<td>88.78</td>
</tr>
<tr>
<td>Oracle combination</td>
<td>90.59</td>
<td>90.66</td>
<td>90.62</td>
</tr>
<tr>
<td>Improvement over baseline</td>
<td>+1.15</td>
<td>+1.03</td>
<td>+1.09</td>
</tr>
<tr>
<td>Section 23 (test set)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Baseline</td>
<td>88.77</td>
<td>88.87</td>
<td>88.82</td>
</tr>
<tr>
<td>Oracle combination</td>
<td>90.11</td>
<td>90.11</td>
<td>90.11</td>
</tr>
<tr>
<td>Improvement over baseline</td>
<td>+1.34</td>
<td>+1.24</td>
<td>+1.29</td>
</tr>
</tbody>
</table>
Effect Using Domain-awareness - Example

Sent#90: *South Korea registered a trade deficit of $101 million in October, reflecting the country’s economic sluggishness, according to government figures released Wednesday.*

Parser_{BASELINE} F-score: 87.80%; incorrect PP-attachment

Oracle prediction F-score: 100%

(Parser_{FINANCIAL} or Parser_{POLITICS})
Short Recap

- The example illustrates that a domain-specifically trained parser may find a correct or better result than the baseline parser.
- Our first experiment shows that our subdomain sensitive parsing instantiation in general has potential.
- We presented parser combination techniques that aim at achieving this potential.
Results of Parser Combination Techniques

<table>
<thead>
<tr>
<th>Parser</th>
<th>Section 00 (development set)</th>
<th>( \leq 40 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline</td>
<td>89.44</td>
<td>89.63</td>
</tr>
<tr>
<td><strong>Parser Pre-selection:</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Divergence Model (DVM)</td>
<td>89.50</td>
<td>89.68</td>
</tr>
<tr>
<td><strong>Parser Post-selection:</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Node Weighting incl. DVM, ( \lambda = 0.6 )</td>
<td>89.53</td>
<td>89.71</td>
</tr>
</tbody>
</table>

Parser Post-selection NW-DVM highest F-score: 89.62\%, i.e. \( +0.09\% \) over baseline.
Results of Parser Combination Techniques

Result of Node Weighting incl. DVM (NW-DVM)

Node Weighting including DVM on the Sentence Level

- WSJ-40 (SentLevel)
- WSJ-100 (SentLevel)
- Baseline WSJ-40
- Baseline WSJ-100

F-score vs. Lambda

- F-score values:
  - 88.5
  - 89
  - 89.5
  - 90
  - 90.5

- Lambda values:
  - 0
  - 0.2
  - 0.4
  - 0.6
  - 0.8
  - 1
Results of Parser Combination Techniques

Summary

- Post-selection that considers both the parse tree and sentence performs best
- Nevertheless, it is closely followed by Parser Pre-selection based on the sentence only
- Results are confirmed on the test set (section 23):
  1. Node Weighting incl. DVM with $\lambda = 0.6$ (+0.08% F-score)
  2. Divergence Model (+0.03%)
Conclusions and Future Work

- Our first instantiation of subdomain sensitive parsing has indeed demonstrated to have potential.
- However, combining the parsers to obtain a substantially better result is not an easy task.
- Our approach leaves space open to extend, refine or improve various parts:
  - Other ways of instantiating domain-dependent parsers (e.g. self-training)
  - More sophisticated notion of domain
  - Further explore parser combination techniques
  - Explore to what extent $n$-best parsing might benefit from subdomain information
Thank you for your attention.
Treebank Weighting

Weight the trees in treebank $TB$ with subdomain statistics and retrain parser.

- Use domain-dependent raw corpus $C$ (flat sentences)
  - $C \in \{\text{sports, financial, politics}\}$
- Induce statistical Language Model (LM) $\theta$ from $C$
- Assign a count$^a$ $f$ to every tree $\pi_i \in TB$:
  \[
  f_\theta(\pi_i) = f_\theta(y_{[\pi_i]}) = -\log P_\theta(y_{[\pi_i]})/n
  \]  
  (7)
- Let $f_\theta^{\text{max}}$ be the maximum count of a tree in $TB$ according to $\theta$. The weight $w_i$ assigned to $\pi_i$ is defined as:
  \[
  w_i = \text{round}\left\{\left(\frac{f_\theta^{\text{max}}}{f_\theta(\pi_i)}\right)^a\right\}
  \]  
  (8)
  where $a \geq 1$ is a scaling constant. In the default setting $a = 1$.

$^a f = \text{average per-word “count” of the yield } y_{[\pi_i]} \text{ under LM } \theta$