When is Self-training Effective for Parsing?

David McClosky
dmcc@cs.brown.edu

Brown Laboratory for Linguistic Information Processing (BLLIP)

Joint work with Eugene Charniak and Mark Johnson
Outline

- What is self-training?
- Previous work
- Experimental setup
- Four hypotheses
- Conclusions
Outline

• What is self-training?
• Previous work
• Experimental setup
• Four hypotheses
• Conclusions
Self-training Requirements

- Labeled data
- Large amount of unlabeled data
- Statistical model:
  - model = train(labeled data)
  - labels = label(model, unlabeled data)
Self-training Pseudocode

1. base = train(labeled)
2. autolabeled = label(base, unlabeled)
3. combined = labeled + autolabeled
4. selftrained = train(combined)

(Not pictured: data selection, iteration, weighting, etc.)
Outline

- What is self-training?
- Previous work
- Experimental setup
- Four hypotheses
- Conclusions
## Previous Work

<table>
<thead>
<tr>
<th>Parser type</th>
<th>Seed size</th>
<th>Iterations</th>
<th>Improved?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Charniak (1997)</td>
<td>Generative</td>
<td>Large</td>
<td>Single</td>
</tr>
<tr>
<td>Steedman et al. (2003)</td>
<td>Generative</td>
<td>Small</td>
<td>Multiple</td>
</tr>
<tr>
<td>Reichart and Rappoport (2007)</td>
<td>Generative</td>
<td>Small</td>
<td>Single</td>
</tr>
</tbody>
</table>

(large = ~40k sentences, small = <1k sentences)

Summary of self-training for parsing experiments
# Previous Work

<table>
<thead>
<tr>
<th></th>
<th>Parser type</th>
<th>Seed size</th>
<th>Iterations</th>
<th>Improved?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Charniak (1997)</td>
<td>Generative</td>
<td>Large</td>
<td>Single</td>
<td>No</td>
</tr>
<tr>
<td>Steedman et al. (2003)</td>
<td>Generative</td>
<td>Small</td>
<td>Multiple</td>
<td>No</td>
</tr>
<tr>
<td>Reichart and Rappoport (2007)</td>
<td>Generative</td>
<td>Small</td>
<td>Single</td>
<td>Yes</td>
</tr>
</tbody>
</table>

(large = ~40k sentences, small = <1k sentences)

**Summary of self-training for parsing experiments**

- In large seed case, generative + discriminative parser is necessary.
## Previous Work

<table>
<thead>
<tr>
<th>Parser type</th>
<th>Seed size</th>
<th>Iterations</th>
<th>Improved?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Charniak (1997)</td>
<td>Generative</td>
<td>Large</td>
<td>Single</td>
</tr>
<tr>
<td>Steedman et al. (2003)</td>
<td>Generative</td>
<td>Small</td>
<td>Multiple</td>
</tr>
<tr>
<td>Reichart and Rappoport (2007)</td>
<td>Generative</td>
<td>Small</td>
<td>Single</td>
</tr>
</tbody>
</table>

(large = ~40k sentences, small = <1k sentences)

### Summary of self-training for parsing experiments

- In large seed case, generative + discriminative parser is necessary.
- Performing only one iteration is better than multiple iterations.
## Previous Analysis

<table>
<thead>
<tr>
<th>Seed size</th>
<th>Predictor</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Length</td>
</tr>
<tr>
<td>McClosky et al. (2006)</td>
<td>Large</td>
</tr>
<tr>
<td>Reichart and Rappoport (2007)</td>
<td>Small</td>
</tr>
</tbody>
</table>
## Previous Analysis

<table>
<thead>
<tr>
<th>Seed Size</th>
<th>Predictor Length</th>
<th>Predictor # Unknown Words</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>McClosky et al. (2006)</strong></td>
<td>Large</td>
<td>+</td>
</tr>
<tr>
<td><strong>Reichart and Rappoport (2007)</strong></td>
<td>Small</td>
<td>+</td>
</tr>
</tbody>
</table>

- Unknown words are a good predictor of self-training's success **only** in the small seed case.
Outline

● What is self-training?
● Previous work
● **Experimental setup**
● Four hypotheses
● Conclusions
Experimental Setup

- Labeled data: WSJ (Marcus et al., 1993)
- Unlabeled data: NANC (Graff, 1995)
- Parser: Charniak and Johnson (2005) reranking parser
Experimental Setup

- Labeled data: WSJ (Marcus et al., 1993)
- Unlabeled data: NANC (Graff, 1995)
- Parser: Charniak and Johnson (2005) reranking parser
Experimental Setup

- Labeled data: WSJ (Marcus et al., 1993)
- Unlabeled data: NANC (Graff, 1995)
- Parser: Charniak and Johnson (2005) reranking parser
Outline

• What is self-training?
• Previous work
• Experimental setup
• **Four hypotheses**
• Conclusions
Hypotheses for Self-training

1. Phase Transition
2. Search Errors
3. Non-generative Reranker Features
4. Bilexical Dependencies
Phase Transition

Self-training works after a phase transition

self-training doesn't help

threshold

self-training helps

accuracy (f-score)
Phase Transition

- Parser: 89.9%
- Reranking Parser: 91.5%

accuracy (f-score)
sections 1, 22, 24

self-training doesn't help
self-training helps
Phase Transition

There is no phase transition for self-training.
Phase Transition

Parser
85.8%
10% WSJ

Reranking Parser
87.0%
10% WSJ

Parser
89.9%
100% WSJ

Reranking Parser
91.5%
100% WSJ

accuracy (f-score)
sections 1, 22, 24

See also: Reichart and Rappoport (2007)

➔ There is no phase transition for self-training.
Self-trained models have fewer search errors

(Daniel Marcu, p.c.)
Self-trained models have fewer search errors

(Daniel Marcu, p.c.)

model prefers worse parse

model  search

errors
Self-trained models have fewer search errors (Daniel Marcu, p.c.)

- Model prefers worse parse
- Best parse for model not found

Search Errors
Some notation

\[ \text{orig} \leftarrow \text{parses from original parser} \]
\[ \text{st} \leftarrow \text{parses from self-trained parser} \]
\[ \text{top}_m(P) \leftarrow \text{best parse in } P \text{ for reranker model } m \]
\[ (m \text{ in } \{\text{orig, st}\}) \]
Comparing $n$-best lists

<table>
<thead>
<tr>
<th>Overlap of $\text{st}$ and $\text{orig}$</th>
<th>66.0%</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\text{top}<em>{\text{st}}(\text{st}) = \text{top}</em>{\text{orig}}(\text{orig})$</td>
<td>42.4%</td>
</tr>
<tr>
<td>$\text{top}_{\text{st}}(\text{st})$ in $\text{orig}$</td>
<td>60.3%</td>
</tr>
<tr>
<td>Search errors</td>
<td>2.5%</td>
</tr>
</tbody>
</table>

(statistics on 5,039 sentences in sections 1, 22, 24)

- Search errors =
  $\text{top}_{\text{st}}(\text{st})$ not in $\text{orig}$ and
  $\text{top}_{\text{orig}}(\text{st}\cup\text{orig}) = \text{top}_{\text{st}}(\text{st})$
Decreasing Search Errors

- Add parses from self-trained n-best list to original parser's n-best list, rescoring by original parser.

  ![Diagram showing the process of adding parses from self-trained parser to the original parser's n-best list.]
Evaluation

- Original reranking parser: 91.5%
- Self-trained reranking parser: 92.0%
- Augmented original reranking parser: 91.7%

(reranking parser f-score on sections 1, 22, 24)
Evaluation

The original parser makes both **model** and **search** errors relative to the self-trained model.

- Original reranking parser: 91.5%
- Self-trained reranking parser: 92.0%
- Augmented original reranking parser: 91.7%

(reranking parser f-score on sections 1, 22, 24)
Reranker features

Non-generative reranker features help self-training more

reranker features
(1.3 million total)
Non-generative reranker features help self-training more

- **Generative** (448k)
- **Non-generative** (885k)

Reranker features (1.3 million total)
Non-generative reranker features help self-training more.

- Bilexical dependencies
- CFG rules
- ... (other features)
- Coordination
- Edge
- N-gram tree

Reranker features:
- Generative (448k)
- Non-generative (885k)

Total reranker features: 1.3 million
Non-generative reranker features help self-training more

(reranking parser f-score on section 24)

reranker features

90.4% 91.1%

91.3%

(1.3 million total)

90.4% 91.1%

generative (448k) non-generative (885k)
Non-generative reranker features help self-training more

8.18.2008

reranking parser without self-training

reranker features (1.3 million total)

90.5%

generative (448k)

90.4%

non-generative (885k)

91.1%

reranking parser f-score on section 24

91.3%
Non-generative reranker features are essential for self-training with a reranker.

(reranking parser f-score on section 24)
Bilexical Dependencies

Self-training teaches the parser about bilexical dependencies

(Mitch Marcus, p.c.)

bilexical dependency
or “bihead”

VP[give]

VBP
give

DT
NN

NP[example]
an
dexample
Bilexical Dependencies

Self-training teaches the parser about bilexical dependencies

(Mitch Marcus, p.c.)

Two ways to test:

- Factor analysis (as in previous work)
- Transfer biheads distribution from self-trained model to original model

![Diagram showing bilexical dependency or “bihead”](image-url)
Factor analysis: Words

Number of unknown words in tree

Total number of incorrect nodes

Reduction in incorrect nodes

8.18.2008   COLING 2008
Factor analysis: Words

- Biggest improvement when no unknown words!
Factor Analysis: Biheads

Number of unknown biheads in tree

Total number of incorrect nodes

Reduction in incorrect nodes
Factor Analysis: Biheads

seeing more unknown biheads helps
Just biheads?

- If self-trained model learns more about biheads, can we transfer that knowledge to original model?

dangerously oversimplified Charniak parsing model!
Just biheads?

- If self-trained model learns more about biheads, can we transfer that knowledge to original model?

<table>
<thead>
<tr>
<th></th>
<th>Expansions</th>
<th>Biheads</th>
<th>Rest</th>
<th>Original Model</th>
<th>89.8%</th>
</tr>
</thead>
<tbody>
<tr>
<td>Self-trained Model</td>
<td>Expansions</td>
<td>Biheads</td>
<td>Rest</td>
<td>90.8%</td>
<td></td>
</tr>
</tbody>
</table>

*(generative parser f-score on sections 1, 22, 24)*
Just biheads?

• If self-trained model learns more about biheads, can we transfer that knowledge to original model?

original model

<table>
<thead>
<tr>
<th>expansions</th>
<th>biheads</th>
<th>rest</th>
<th>89.8%</th>
</tr>
</thead>
<tbody>
<tr>
<td>expansions</td>
<td>biheads</td>
<td>rest</td>
<td>90.1%</td>
</tr>
</tbody>
</table>

self-trained model

| expansions | biheads | rest | 90.8% |

(generative parser f-score on sections 1, 22, 24)
Just biheads?

- If self-trained model learns more about biheads, can we transfer that knowledge to original model?

<table>
<thead>
<tr>
<th></th>
<th>expansions</th>
<th>biheads</th>
<th>rest</th>
<th>f-score</th>
</tr>
</thead>
<tbody>
<tr>
<td>original model</td>
<td></td>
<td></td>
<td></td>
<td>89.8%</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>90.1%</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>90.4%</td>
</tr>
<tr>
<td>self-trained model</td>
<td>expansions</td>
<td>biheads</td>
<td>rest</td>
<td>90.8%</td>
</tr>
</tbody>
</table>

(Generative parser f-score on sections 1, 22, 24)
Just biheads?

- If self-trained model learns more about biheads, can we transfer that knowledge to original model?

<table>
<thead>
<tr>
<th></th>
<th>Expansions</th>
<th>Biheads</th>
<th>Rest</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>Original model</td>
<td></td>
<td></td>
<td></td>
<td>89.8%</td>
</tr>
<tr>
<td>Self-trained</td>
<td></td>
<td></td>
<td></td>
<td>90.1%</td>
</tr>
<tr>
<td>self-trained</td>
<td></td>
<td></td>
<td></td>
<td>90.4%</td>
</tr>
<tr>
<td>model</td>
<td></td>
<td></td>
<td></td>
<td>90.7%</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>90.8%</td>
</tr>
</tbody>
</table>

(generative parser f-score on sections 1, 22, 24)
Just biheads?

➔ Self-training improves biheads distribution, but also expansions distribution.

<table>
<thead>
<tr>
<th></th>
<th>Expansions</th>
<th>Biheads</th>
<th>Rest</th>
<th>Original Model</th>
<th>Self-Trained Model</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>89.8%</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>90.1%</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>90.4%</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>90.7%</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>90.8%</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

(Generative parser f-score on sections 1, 22, 24)
Outline

- What is self-training?
- Previous work
- Experimental setup
- Four hypotheses
- Conclusions
Hypothesis Results

1. Phase Transition
   Disproved.

2. Search Errors
   Reducing search errors helps but model errors remain.

3. Non-generative Reranker Features
   Reranker features must be different from generative parser.

4. Bilexical Dependencies
   Biheads correlated with self-training improvements.
   Self-training helps all parts of the parser, not just biheads.
# Two different cases?

<table>
<thead>
<tr>
<th></th>
<th>Predictor</th>
<th>Need reranker?</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Length</td>
<td>unk. words</td>
</tr>
<tr>
<td>Large seed</td>
<td>+</td>
<td>-</td>
</tr>
<tr>
<td>Small seed</td>
<td>+</td>
<td>+</td>
</tr>
</tbody>
</table>
A Possible Connection

- **Small seed case**: Vocabulary is sparse so unknown *words* may be resolved in unlabeled text.
- **Large seed case**: Learn new *head information* in unlabeled text. Non-generative features in the reranker needed to handle more complex constructions.
Future Work

• “Bilexical dependencies” experiments with small seed size.

• Different parsers (Collins, LTAG, discriminative, ...) with and without rerankers.

• More hypotheses...
  (Audience participation?)
Acknowledgments

This work was supported by DARPA GALE contract HR0011-06-2-0001.

We would like to thank Roi Reichart, Matt Lease and the rest of the BLLIP team, and our anonymous reviewers for their comments.

Questions?

http://bllip.cs.brown.edu/selftraining/
Extra Slides
Bigrams and Biheads

Number of unknown bigrams in tree

Number of unknown biheads in tree
Bigrams and Biheads

seeing more unknown bigrams/biheads helps