A Comparison of Structural Correspondence Learning and Self-training for Discriminative Parse Selection

Barbara Plank
University of Groningen, The Netherlands
b.plank@rug.nl

Abstract

This paper evaluates two semi-supervised techniques for the adaptation of a parse selection model to Wikipedia domains. The techniques examined are *Structural Correspondence Learning* (SCL) (Blitzer et al., 2006) and *Self-training* (Abney, 2007; McClosky et al., 2006). A preliminary evaluation favors the use of SCL over the simpler self-training techniques.

1 Introduction and Motivation

Parse selection constitutes an important part of many parsing systems (Hara et al., 2005; van Noord and Malouf, 2005; McClosky et al., 2006). Yet, there is little to no work focusing on the *adaptation* of parse selection models to novel domains. This is most probably due to the fact that potential gains for this task are inherently bounded by the underlying grammar. The few studies on adapting parse disambiguation models, like Hara et al. (2005), have focused exclusively on *supervised domain adaptation*, i.e. one has access to a comparably small, but labeled amount of target data. In contrast, in *semi-supervised domain adaptation* one has only *unlabeled* target data. It is a more realistic situation, but at the same time also considerably more difficult.

In this paper we evaluate two semi-supervised approaches to domain adaptation of a discriminative parse selection model. We examine *Structural Correspondence Learning* (SCL) (Blitzer et al., 2006) for this task, and compare it to several variants of *Self-training* (Abney, 2007; McClosky et al., 2006). For empirical evaluation (section 4) we use the Alpino parsing system for Dutch (van Noord and Malouf, 2005). As target domain, we exploit Wikipedia as primary test and training collection.

2 Previous Work

So far, Structural Correspondence Learning has been applied successfully to PoS tagging and Sentiment Analysis (Blitzer et al., 2006; Blitzer et al., 2007). An attempt was made in the CoNLL 2007 shared task to apply SCL to non-projective dependency parsing (Shimizu and Nakagawa, 2007). However, the system just ended up at rank 7 out of 8 teams. Based on annotation differences in the datasets (Dredze et al., 2007) and a bug in their system (Shimizu and Nakagawa, 2007), their results are inconclusive. A recent attempt (Plank, 2009) shows promising results on applying SCL to parse disambiguation. In this paper, we extend that line of work and compare SCL to bootstrapping approaches such as self-training.

Studies on self-training have focused mainly on generative, constituent based parsing (Steedman et al., 2003; McClosky et al., 2006; Reichart and Rappoport, 2007). Steedman et al. (2003) as well as Reichart and Rappoport (2007) examine self-training for PCFG parsing in the small seed case ($< 1k$ labeled data), with different results. In contrast, McClosky et al. (2006) focus on large seeds and exploit a reranking-parser. Improvements are obtained (McClosky et al., 2006; McClosky and Charniak, 2008), showing that a reranker is necessary for successful self-training in such a high-resource scenario. While they self-trained a generative model, we examine self-training and SCL for semi-supervised adaptation of a discriminative parse selection system.

Proceedings of the NAACL HLT Workshop on Semi-supervised Learning for Natural Language Processing, pages 37–42,
Boulder, Colorado, June 2009.

3 Semi-supervised Domain Adaptation

3.1 Structural Correspondence Learning

Structural Correspondence Learning (Blitzer et al., 2006) exploits unlabeled data from both source and target domain to find correspondences among features from different domains. These correspondences are then integrated as new features in the labeled data of the source domain. The outline of SCL is given in Algorithm 1.

The key to SCL is to exploit *pivot features* to automatically identify feature correspondences. Pivots are features occurring frequently and behaving similarly in both domains (Blitzer et al., 2006). They correspond to auxiliary problems in Ando and Zhang (2005). For every such pivot feature, a binary classifier is trained (step 2 of Algorithm 1) by masking the pivot feature in the data and trying to predict it with the remaining non-pivot features. Non-pivots that correlate with many of the same pivots are assumed to correspond. These pivot predictor weight vectors thus implicitly align non-pivot features from source and target domain. Intuitively, if we are able to find good correspondences through 'linking' pivots, then the augmented source data should transfer better to a target domain (Blitzer et al., 2006).

Algorithm 1 SCL (Blitzer et al., 2006)

1: Select m pivot features.
2: Train m binary classifiers (pivot predictors). Create matrix $W_{n\times m}$ of binary predictor weight vectors $W = [w_1, .., w_m]$, with n number of nonpivots.
3: Dimensionality Reduction. Apply SVD to W: $W_{n\times m} = U_{n\times n}D_{n\times m}V^T_{m\times m}$ and select $\theta = U^T_{[1:h,:]}$ (the h top left singular vectors of W).
4: Train a new model on the original and new features obtained by applying the projection $x \cdot \theta$.

SCL for Discriminative Parse Selection So far, pivot features on the word level were used (Blitzer et al., 2006; Blitzer et al., 2007). However, for parse disambiguation based on a conditional model they are irrelevant. Hence, we follow Plank (2009) and actually first parse the unlabeled data. This allows a possibly noisy, but more abstract representation of the underlying data. Features thus correspond to properties of parses: application of grammar rules (*r1*,*r2* features), dependency relations (*dep*), PoS tags (*f1*,*f2*), syntactic features (*s1*), precedence (*mf*), bilexical preferences (*z*), apposition (*appos*) and further features for unknown words, temporal phrases, coordination (*h*,*in_year* and *p1*, respectively). These features are further described in van Noord and Malouf (2005).

Selection of pivot features As pivot features should be common across domains, here we restrict our pivots to be of the type *r1*,*p1*,*s1* (the most frequently occurring feature types). In more detail, *r1* indicates which grammar rule applied, *p1* whether coordination conjuncts are parallel, and *s1* whether local/non-local extraction occurred. We count how often each feature appears in the parsed source and target domain data, and select those *r1*,*p1*,*s1* features as *pivot features*, whose count is $> t$, where t is a specified threshold. In all our experiments, we set $t = 5000$. In this way we obtained on average 360 pivot features, on the datasets described in Section 4.

3.2 Self-training

Self-training (Algorithm 2) is a simple single-view bootstrapping algorithm. In self-training, the newly labeled instances are taken at face value and added to the training data.

There are many possible ways to instantiate self-training (Abney, 2007). One variant, introduced in Abney (2007) is the notion of '(in)delibility': in the *delible* case the classifier relabels all of the unlabeled data from scratch in every iteration. The classifier may become unconfident about previously selected instances and they may drop out (Steven Abney, personal communication). In contrast, in the *indelible* case, labels once assigned do not change again (Abney, 2007).

In this paper we look at the following variants of self-training:

- single versus multiple iterations,
- selection versus no selection (taking all self-labeled data or selecting presumably higher quality instances); different scoring functions for selection,
- delibility versus indelibility for multiple iterations.

Algorithm 2 Self-training (indelible) (Abney, 2007).

1: L_0 is labeled [seed] data, U is unlabeled data
2: $c \leftarrow train(L_0)$
3: **repeat**
4: $\quad L \leftarrow L + select(label(U - L, c))$
5: $\quad c \leftarrow train(L)$
6: **until** stopping criterion is met

Scoring methods We examine three simple scoring functions for instance selection: i) *Entropy* ($-\sum_{y \in Y(s)} p(\omega|s,\theta) \log p(\omega|s,\theta)$). ii) *Number of parses* ($|Y(s)|$); and iii) *Sentence Length* ($|s|$).

4 Experiments and Results

Experimental Design The system used in this study is Alpino, a two-stage dependency parser for Dutch (van Noord and Malouf, 2005). The first stage consists of a HPSG-like grammar that constitutes the parse generation component. The second stage is a Maximum Entropy (MaxEnt) parse selection model. To train the MaxEnt model, parameters are estimated based on informative samples (Osborne, 2000). A parse is added to the training data with a score indicating its "goodness" (van Noord and Malouf, 2005). The score is obtained by comparing it with the gold standard (if available; otherwise the score is approximated through parse probability).

The source domain is the Alpino Treebank (van Noord and Malouf, 2005) (newspaper text; approx. 7,000 sentences; 145k tokens). We use Wikipedia both as testset and as unlabeled target data source. We assume that in order to parse data from a very specific domain, say about the artist Prince, then data related to that domain, like information about the New Power Generation, the Purple rain movie, or other American singers and artists, should be of help. Thus, we exploit Wikipedia's category system to gather domain-specific target data. In our empirical setup, we follow Blitzer et al. (2006) and balance the size of source and target data. Thus, depending on the size of the resulting target domain dataset, and the "broadness" of the categories involved in creating it, we might wish to filter out certain pages. We implemented a filter mechanism that excludes pages of a certain category (e.g. a supercategory that is hypothesized to be "too broad"). Further details about the dataset construction are given in (Plank, 2009). Table 1 provides information on the target domain datasets constructed from Wikipedia.

Related to	Articles	Sents	Tokens	Relationship
Prince	290	9,772	145,504	filtered super
Paus	445	8,832	134,451	all
DeMorgan	394	8,466	132,948	all

Table 1: Size of related unlabeled data; relationship indicates whether all related pages are used or some are filtered out.

The size of the target domain testsets is given in Table 2. As evaluation measure concept accuracy (CA) (van Noord and Malouf, 2005) is used (similar to labeled dependency accuracy).

The training data for the pivot predictors are the 1-best parses of source and target domain data as selected by the original Alpino model. We report on results of SCL with dimensionality parameter set to $h = 25$, and remaining settings identical to Plank (2009) (i.e., no feature-specific regularization and no feature normalization and rescaling).

Baseline Table 2 shows the baseline accuracies (model trained on labeled out-of-domain data) on the Wikipedia testsets (last column: size in number of sentences). The second and third column indicate lower (first parse) and upper- (oracle) bounds.

Wikipedia article	baseline	first	oracle	sent
Prince (musician)	85.03	71.95	88.70	357
Paus Johannes Paulus II	85.72	74.30	89.09	232
Augustus De Morgan	80.09	70.08	83.52	254

Table 2: Supervised Baseline results.

SCL and Self-training results The results for SCL (Table 3) show a small, but consistent increase in absolute performance on all testsets over the baselines (up to $+0.27$ absolute CA or 7.34% relative error reduction, which is significant at $p < 0.05$ according to sign test).

In contrast, basic self-training (Table 3) achieves roughly only baseline accuracy and lower performance than SCL, with one exception. On the DeMorgan testset, self-training scores slightly higher than SCL. However, the improvements of both SCL and self-training are not significant on this rather

small testset. Indeed, self-training scores better than the baseline on only 5 parses out of 254, while its performance is lower on 2, leaving only 3 parses that account for the difference.

	CA	ϕ	Rel.ER
Prince baseline	85.03	78.06	0.00
SCL	$\star$ 85.30	79.67	7.34
Self-train (all-at-once)	85.08	78.38	1.46
Paus baseline	85.72	77.23	0.00
SCL	85.82	77.87	2.81
Self-train (all-at-once)	85.78	77.62	1.71
DeMorgan baseline	80.09	74.44	0.00
SCL	80.15	74.92	1.88
Self-train (all-at-once)	80.24	75.63	4.65

Table 3: Results of SCL and self-training (single iteration, no selection). Entries marked with $\star$ are statistically significant at $p < 0.05$. The ϕ score incorporates upper- and lower-bounds.

To gauge whether other instantiations of self-training are more effective, we evaluated the self-training variants introduced in section 3.2 on the Prince dataset. In the iterative setting, we follow Steedman et al. (2003) and parse 30 sentences from which 20 are selected in every iteration.

With regard to the comparison of delible versus indelible self-training (whether labels may change), our empirical findings shows that the two cases achieve *very* similar performance; the two curves highly overlap (Figure 1). The accuracies of both curves fluctuate around 85.13, showing no upward or downward trend. In general, however, indelibility is preferred since it takes considerably less time (the classifier does not have to relabel U from scratch in every iteration). In addition, we tested EM (which uses all unlabeled data in each iteration). Its performance is consistently lower, varying around the baseline.

Figure 2 compares several self-training variants with the supervised baseline and SCL. It summarizes the effect of i) selection versus no selection (and various selection techniques) as well as ii) single versus multiple iterations of self-training. For clarity, the figure shows the learning curve of the best selection technique only, but depicts the performance of the various selection techniques in a single iteration (non-solid lines).

In the iterative setting, taking the whole self-labeled data and not selecting certain instances (grey curve in Figure 2) degrades performance. In contrast, selecting shorter sentences slightly improves accuracy, and is the best selection method among the ones tested (shorter sentences, entropy, fewer parses).

For all self-training instantiations, running multiple iterations is on average just the same as running a single iteration (the non-solid lines are roughly the average of the learning curves). Thus there is no real need to run several iterations of self-training.

The main conclusion is that in contrast to SCL, none of the self-training instantiations achieves a significant improvement over the baseline.

5 Conclusions and Future Work

The paper compares Structural Correspondence Learning (Blitzer et al., 2006) with (various instances of) self-training (Abney, 2007; McClosky et al., 2006) for the adaptation of a parse selection model to Wikipedia domains.

The empirical findings show that none of the evaluated self-training variants (delible/indelible, single versus multiple iterations, various selection techniques) achieves a significant improvement over the baseline. The more 'indirect' exploitation of unlabeled data through SCL is more fruitful than pure self-training. Thus, favoring the use of the more complex method, although the findings are not confirmed on all testsets.

Of course, our results are preliminary and, rather than warranting yet many definite conclusions, encourage further investigation of SCL (varying size of target data, pivots selection, bigger testsets as well as other domains etc.) as well as related semi-supervised adaptation techniques.

Acknowledgments

Thanks to Gertjan van Noord and the anonymous reviewers for their comments. The Linux cluster of the High-Performance Computing Center of the University of Groningen was used in this work.

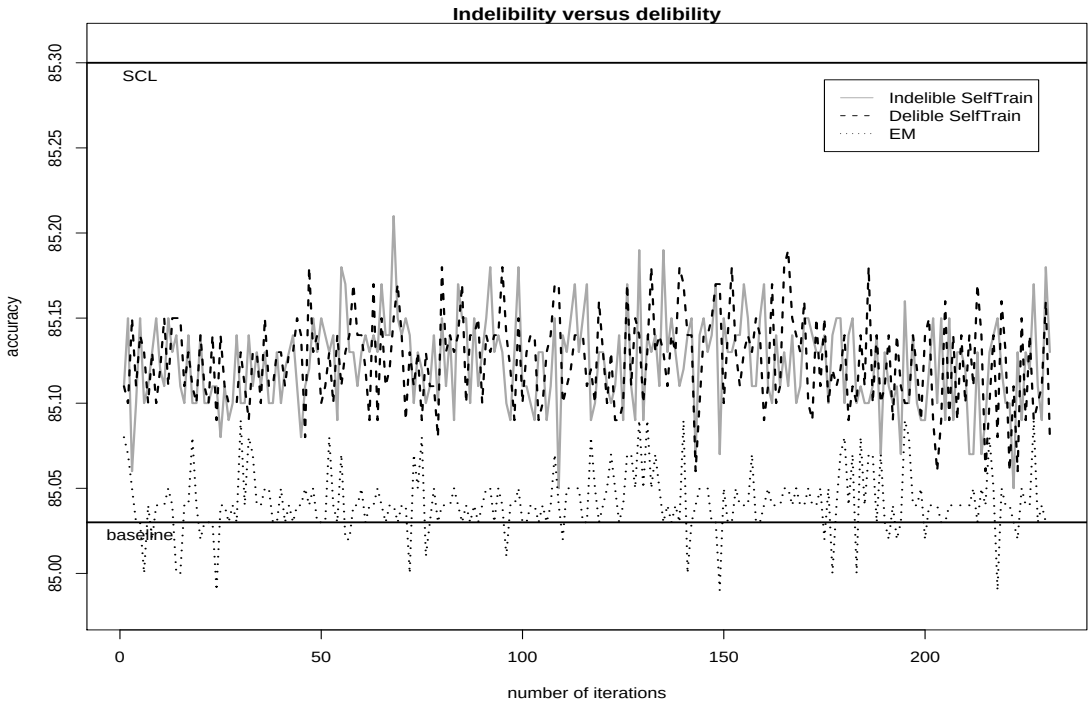

Figure 1: Delible versus Indelible self-training and EM. Delible and indelible self-training achieve *very* similar performance. However, indelibility is preferred over delibility since it is considerably faster.

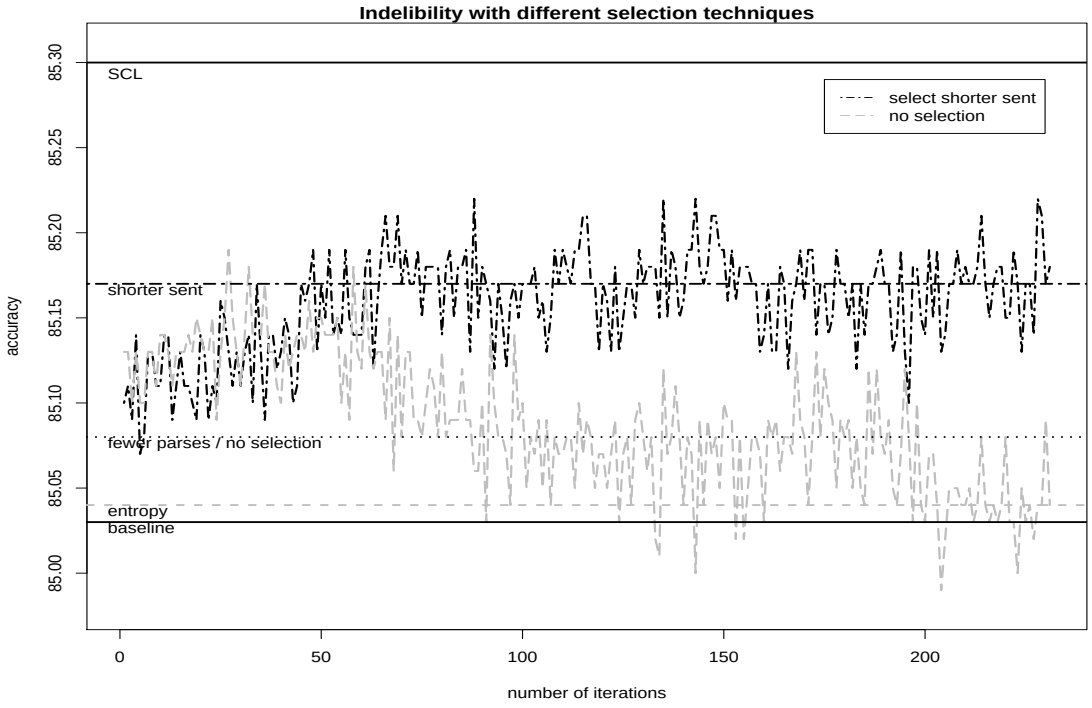

Figure 2: Self-training variants compared to supervised baseline and SCL. The effect of various selection techniques (Sec. 3.2) in a single iteration is depicted (non-solid lines; fewer parses and no selection achieve identical results). For clarity, the figure shows the learning curve for the best selection technique only (shorter sent) versus no selection. On average running multiple iterations is just the same as a single iteration. In all cases SCL still performs best.

References

Steven Abney. 2007. *Semi-supervised Learning for Computational Linguistics*. Chapman & Hall.

Rie Kubota Ando and Tong Zhang. 2005. A framework for learning predictive structures from multiple tasks and unlabeled data. *Journal of Machine Learning Research*, 6:1817–1853.

John Blitzer, Ryan McDonald, and Fernando Pereira. 2006. Domain adaptation with structural correspondence learning. In *Conference on Empirical Methods in Natural Language Processing*, Sydney, Australia.

John Blitzer, Mark Dredze, and Fernando Pereira. 2007. Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification. In *Association for Computational Linguistics*, Prague, Czech Republic.

Mark Dredze, John Blitzer, Pratha Pratim Talukdar, Kuzman Ganchev, Joao Graca, and Fernando Pereira. 2007. Frustratingly hard domain adaptation for parsing. In *Proceedings of the CoNLL Shared Task Session - Conference on Natural Language Learning*, Prague, Czech Republic.

Tadayoshi Hara, Miyao Yusuke, and Jun'ichi Tsujii. 2005. Adapting a probabilistic disambiguation model of an hpsg parser to a new domain. In *Proceedings of the International Joint Conference on Natural Language Processing*.

David McClosky and Eugene Charniak. 2008. Self-training for biomedical parsing. In *Proceedings of ACL-08: HLT, Short Papers*, pages 101–104, Columbus, Ohio, June. Association for Computational Linguistics.

David McClosky, Eugene Charniak, and Mark Johnson. 2006. Effective self-training for parsing. In *Proceedings of the Human Language Technology Conference of the NAACL, Main Conference*, pages 152–159, New York City. Association for Computational Linguistics.

Miles Osborne. 2000. Estimation of stochastic attribute-value grammars using an informative sample. In *Proceedings of the Eighteenth International Conference on Computational Linguistics (COLING 2000)*.

Barbara Plank. 2009. Structural correspondence learning for parse disambiguation. In *Proceedings of the Student Research Workshop at EACL 2009*, Athens, Greece, April.

Roi Reichart and Ari Rappoport. 2007. Self-training for enhancement and domain adaptation of statistical parsers trained on small datasets. In *Proceedings of Association for Computational Linguistics*, Prague.

Nobuyuki Shimizu and Hiroshi Nakagawa. 2007. Structural correspondence learning for dependency parsing. In *Proceedings of the CoNLL Shared Task Session of EMNLP-CoNLL 2007*.

Mark Steedman, Miles Osborne, Anoop Sarkar, Stephen Clark, Rebecca Hwa, Julia Hockenmaier, Paul Ruhlen, Steven Baker, and Jeremiah Crim. 2003. Bootstrapping statistical parsers from small datasets. In *In Proceedings of the EACL*, pages 331–338.

Gertjan van Noord and Robert Malouf. 2005. Wide coverage parsing with stochastic attribute value grammars. Draft. A preliminary version of this paper was published in the Proceedings of the IJCNLP workshop Beyond Shallow Analyses, Hainan China, 2004.