Data point selection for self-training

Ines Rehbein
SFB 632: Information structure
University of Potsdam
irehbein@uni-potsdam.de

Abstract

Problems for parsing morphologically rich languages are, amongst others, caused by the higher variability in structure due to less rigid word order constraints and by the higher number of different lexical forms. Both properties can result in sparse data problems for statistical parsing. We present a simple approach for addressing these issues. Our approach makes use of self-training on instances selected with regard to their similarity to the annotated data. Our similarity measure is based on the perplexity of part-of-speech trigrams of new instances measured against the annotated training data. Preliminary results show that our method outperforms a self-training setting where instances are simply selected by order of occurrence in the corpus and argue that self-training is a cheap and effective method for improving parsing accuracy for morphologically rich languages.

1 Introduction

Up to now, most work on statistical parsing has been focussed on English, a language with a configurational word order and little morphology. The inherent properties of morphologically rich languages include a higher variability in structure due to less rigid word order constraints, thus leading to greater attachment ambiguities, and a higher number of different word forms, leading to coverage problems caused by sparse data. These issues pose a great challenge to statistical parsing.

One sensible way to treat these issues is the development of more sophisticated parsing models adapted to the language-specific properties of morphologically rich languages. Another, simpler approach, tries to overcome the problems outlined above by expanding the training data. Possible approaches for expansion include self-training and active learning.

For self-training a parser is trained on a seed dataset of gold trees and applied to new text, either coming from the same domain or, in the context of domain adaptation, from a domain different from the seed data. The parser output trees are then added to the seed data and the parser is re-trained on its own output. For the in-domain setting it is quite unintuitive why this approach should work, as we only add more of what the parser already knows, and we also include a considerable amount of errors in the training set.

Active learning, on the other hand, tries to expand the training set by selecting those instances which provide the parser with a high amount of new information.[1] The underlying idea is that those instances have yet to be learned by the parser and thus will support the learning process. These instances have to be labelled by a human coder (often called the oracle) and then added to the seed data. The parser is re-trained and new instances can be selected, based on the new model. The intuition why this approach should work is more straightforward than for the self-training setting: we do provide the model with new, unseen information and, assuming that our oracle is right, the amount of noise is kept to a minimum. The great advantage of self-training, however,

[1] Common measures for data point selection are based on the uncertainty of the model with regard to its own predictions.

Proceedings of the 2nd Workshop on Statistical Parsing of Morphologically-Rich Languages (SPMRL 2011), pages 62–67,
Dublin, Ireland, October 6, 2011.

is that it is unsupervised, thus obviating the need for human annotation.

In this study we test the potential of self-training for parsing morphologically rich languages. We present experiments for German, a language with rich morphology (relative to English) and semi-free word order, and show that self-training can improve parsing accuracy when only a small amount of labelled training data is available. Furthermore, we show that selecting sentences for self-training on the basis of similarity to the training data is a good strategy which can further improve results while avoiding the downside of expensive human annotation.

The paper is structured as follows. Section 2 reports on related work. Section 3 describes the setup of our experiments and reports preliminary results. In Section 4 we conclude and outline future work.

2 Related work

The question whether or not self-training can be employed to improve parsing accuracy and to overcome sparse data problems has gained a lot of attention in recent years. While training a generative parsing model on its own output (Charniak, 1997; Steedman et al., 2003) does not seem to work well, McClosky et al. (2006a; 2006b) showed promising results when combining the self-training approach with a two-stage reranking parser model (Charniak and Johnson, 2005). This triggered a number of follow-up studies especially in the area of domain adaptation (Bacchiani et al., 2006; Foster et al., 2007; McClosky et al., 2010), where self-training is used to adapt the parser to a target domain for which no (or only a small amount of) annotated training data is available.

(Reichart and Rappoport, 2007) are the first to report successful self-training using a generative parsing model only. They claim that the crucial difference to earlier studies is the size of the seed data and the number of parser output trees added to the training data. In their experiments they train a reimplementation of Collins' parsing model 2 on a small seed set of trees (100-2000 trees) from the WSJ and add automatically parsed analyses for WSJ sections 2-21. Then they test their models on section 23 of the WSJ and report a substantial improvement for the in-domain self-training setting.

Discussion has focussed on the question of which factors are responsible for the success (or failure) of self-training. Reichart and Rappoport (2007) show that the number of unknown words is a good indicator of the usefulness of self-training when applied to small seed data sets. McClosky et al. (2008) have provided a thorough analysis and conclude that an important source of improvement comes from seeing words already known to the parser in new contexts. A question which, until now, has not gained much attention is the impact of language-specific features on the effect of self-training.

Another strand of research related to our work is that of cross-language adaptation of parsers, where there exists labelled data for one language but none (or only little) for the other. Zeman and Resnik (2008) present cross-language adaptation of a constituency parser by mapping the part-of-speech tags from the source and target languages into a universal tagset, claiming that the similarities between two closely related languages allow for abstraction from the level of word forms. They apply their method to Danish and Swedish, two closely related languages, and present an f-score of 66.4% for constituency trees for Swedish after having trained their parser on data from the Danish treebank.

Søgaard (2011) pushes this line of research further and applies it to languages as different as Arabic, Bulgarian, Danish and Portuguese. The basic approach is similar to (Zeman and Resnik, 2008). Søgaard (2011) delexicalises the treebanks and maps the part-of-speech tags into one common tagset. Crucial for the success of his approach is the filtering of the training data. Søgaard only trains on the 90% of the source trees which are most similar to the target language. As a similarity measure he uses perplexity on the basis of POS ngrams. The results are quite impressive. Despite the very different properties of the languages Søgaard achieves f-scores in the range of 50-75% on full-length sentences.

We take up the idea of data point selection based on similarity and apply it to our self-training scenario. Is is not straightforward whether this strategy will work or not, as it may seem to be diametrically opposed to the idea of active learning, where the system is provided with instances with a high information content. Here, on the contrary, we select instances which are similar to the training data, which might mean that they do not contribute new, useful information for the parser. Nevertheless, we hope that, since they are similar to what the parser already knows, it might handle these instances reasonably well and therefore the amount of noise added to the training set will be small. At the same time we assume with McClosky et al. (2008) that one important factor in self-training is providing the parser with additional context for already known words, and therefore presume that selecting similar sentences will support the learning process.

3 Self-training experiments

3.1 Data

In our experiments we use data from two German treebanks. We take syntactically annotated trees from the TiGer treebank (Brants et al., 2002) and raw text from the TüBa-D/Z treebank (Telljohann et al., 2005). The TüBa-D/Z (Release 6) consists of 55 814 sentences, TiGer (Release 2) includes 50 474 sentences. Sentence length in the two treebanks is comparable, with around 17 words per sentence. TiGer is annotated with phrase structure trees, dependency (grammatical relation) information and POS tags, according to the Stuttgart Tübingen Tag Set (STTS) (Schiller et al., 1995). The tree structure is flat and does not contain unary nodes as non-local dependencies are encoded by the use of crossing branches.

Both treebanks include German newspaper text, coming from two different newspapers (Frankfurter Rundschau and *taz*). Rehbein and van Genabith (2007) showed that there are considerable domain differences between the two treebanks and that the texts can easily be separated on the basis of the distribution of part-of-speech tags in the two corpora.

3.2 Preprocessing

We use the TiGer trees as our training data and the sentences in the TüBa-D/Z for expanding the corpus. Our setup is as follows.

First we normalised different forms of apostrophes in the text.[2] Then we divided the 50474 trees in TiGer into training and test set, following the proposal described in Dubey (2004). We split the data into 20 buckets by placing the first tree of the treebank into bucket 1, the second tree into bucket 2, and so on. We then combined the content of buckets 1 to 19 into the training set (47951 trees), and used bucket 20 as our test set (2523 trees).

From the randomly ordered training set we created 8 new training subsets of increasing size, putting the first 5000 trees in the training set in subset 1, the first 10000 trees in subset 2, and so on, up to 40000 trees (subset 8). We resolved the crossing branches in the TiGer trees by attaching the non-head child nodes higher up in the tree, following (Kübler, 2005).

3.3 Data point selection

In the next step we created language models for each of the 8 TiGer training subsets on the basis of the part-of-speech trigrams[3] and computed the perplexity for each sentence in the TüBa-D/Z treebank based on its part-of-speech trigrams. The TüBa-D/Z POS tags used in our experiments have been assigned using the RFTagger (Schmid and Laws, 2008). For TiGer, we used the gold POS tags.

Perplexity (Equation 1) is an information-theoretic measure and can be used to assess the homogeneity of a corpus. It can be unpacked as the inverse of the corpus probability, normalised by corpus size. The perplexity of a sentence from the TüBa-D/Z tells us how similar this sentence is to the TiGer training data.

$$PP(W) = \sqrt[N]{\frac{1}{P(w_1 w_2 ... w_N)}} \qquad (1)$$

For each of the 8 subsets we selected the 25000 sentences from the TüBa-D/Z with the lowest perplexity, thus the TüBa-D/Z sentences most similar in structure to the respective TiGer training subset. Then we parsed the selected sentences and added them to the TiGer training data (subsets 1-8). We re-trained the parser and evaluated against the TiGer test set, comparing the results against the perfor-

[2] TiGer uses “ and ‘ for opening and ” and ’ for closing double quotes, TüBa-D/Z uses ‘ for opening and ’ for closing single quotes but does not distinguish between opening and closing double quotes.

[3] The language models were produced and calculated using the CMU/Cambridge toolkit (`http://mi.eng.cam.ac.uk/prc14/toolkit.html`).

mance of the parser when trained on the original subset from the TiGer treebank.

3.4 Parsing experiments

For our experiments we use the unlexicalised Berkeley parser (Petrov et al., 2006) and the lexicalised form of the Stanford parser (Klein and Manning, 2003). The Berkeley parser is an unlexicalised latent variable PCFG parser which uses a split-and-merge technique to automatically refine the training data. The splits result in more and more fine-grained subcategories, which are merged again if not proven useful. We train a PCFG from each of the 8 training subsets by carrying out six cycles of the split-and-merge process. The model is language-agnostic. The Stanford parser provides a factored probabilistic model combining a PCFG with a dependency model. We use the Stanford parser in its lexicalised, markovised form.[4]

Both parsers were trained on the syntactic nodes of the trees only, stripping off the grammatical function (GF) labels from the trees. We add the GF to the parser output in a postprocessing step, using the method of (Seeker et al., 2010), and include GF in the evaluation. Training the parser on syntactic node labels without GF has the advantage of considerably reducing the number of atomic labels in the grammar. As a result, we obtain smaller grammars which are more efficient for parsing, and we also avoid sparse data problems. We also lose information, but the treebank refinement techniques used by the Berkeley parser easily recover this information and thus yield comparable results for both settings. As an additional benefit we avoid the problem of multiple governable GF assigned to children of the same parent node, an error occasionally made by the Berkeley parser. The method by (Seeker et al., 2010), on the other hand, uses linguistically informed hard constraints to prevent these errors.

While we computed perplexity on the basis of the gold POS tags in TiGer treebank and automatically assigned POS tags to the TüBa-D/Z sentences, for parsing we used raw text as input and let the parsers assign their own POS tags.

3.5 Results

We compare the impact of self-training on parsing accuracy for a lexicalised (Stanford) and an unlexicalised (Berkeley) parsing model. For self-training we test the following settings: a) selecting new training data from TüBa-D/Z based on perplexity, adding the 25 000 parser output trees most similar to the TiGer training subset (PERPLEXITY) and b) adding the first 25 000 sentences from the TüBa-D/Z (FIRST) to each of the TiGer training subsets.

Table 1 shows results for the different settings including GF in the evaluation.[5] In general, the results for the Berkeley parser are much higher (according to the PARSEVAL metric) than the results for the lexicalised version of the Stanford parser. The most striking finding is that for the Stanford parser self-training was not able to improve parsing accuracy over the baseline of training the parser on the (much smaller) TiGer training subsets only, while for the Berkeley parser we get a significant improvement of 2.9% and 1.9% f-score for the two smallest training subsets. With increasing size of the training set the gap between the results achieved on the original TiGer training data and on the expanded training sets becomes smaller, but even for the largest training set we achieve a significant improvement of 0.9%.

While results for the Stanford parser are much lower than the ones for Berkeley and self-training fails to outperform the baseline in all cases, the general trend for the self-training settings (PERPLEXITY, FIRST) is the same. Selecting new training instances on the basis of similarity helps mostly for smaller data sets, while for the larger training sets there does not seem to be a significant difference between the two settings. This finding is quite intuitive. In the self-training setting we have a trade-off between new information provided to the parser and noise added to the training set. For small training sets new context information has a far higher impact, while for training sets of increasing size we already have more information in the labelled data, and thus the gains from providing additional context to the parser are lower than the harm we cause by

[4] Parameters: hmarkov=1, vmarkov=2

[5] For significance testing we use the Randomized Parsing Evaluation Comparator provided by Dan Bikel (http://www.cis.upenn.edu/~dbikel/software.html#comparator)

subset	**5000**	**10000**	**15000**	**20000**	**25000**	**30000**	**35000**	**40000**
		Stanford parser (BASELINE)						
precision	57.77***	62.22***	64.32***	65.57***	66.18***	66.38***	67.34***	68.13***
recall	61.24 .	64.37***	65.85***	66.81***	67.17***	76.29***	68.07***	68.81***
f-score	**59.46**	**63.27**	**65.08**	**66.18**	**66.67**	**66.84**	**67.70**	**68.40**
		Stanford parser, self-trained (PERPLEXITY)						
precision	55.02	59.89	62.43	63.20	63.82	64.86	65.61	66.53
recall	60.57	63.38	64.94	65.28	65.61	66.33	66.81	67.62
f-score	57.66	61.59	63.66	64.22	64.70	65.58	66.20	67.02
		Stanford parser, self-trained (FIRST)						
precision	54.60	59.89	62.42	63.34	64.36	64.93	65.94	66.75
recall	60.20	63.52	64.85	65.42	66.11	66.49	67.17	67.86
f-score	57.26	61.65	63.61	64.36	65.22	65.70	66.55	67.30
		Berkeley parser (BASELINE)						
precision	63.39	66.65	69.16	70.50	71.03	72.54	72.79	73.06
recall	63.22	66.50	68.88	70.07	70.72	72.11	72.41	72.71
f-score	63.30	66.58	69.02	70.28	70.87	72.32	72.60	72.88
		Berkeley parser, self-trained (PERPLEXITY)						
precision	66.39***	68.66***	70.23**	71.42**	71.55	73.59***	73.44 .	74.08***
recall	65.98***	68.43***	69.82**	71.05**	71.02	73.10***	72.74	73.55**
f-score	**66.18**	**68.54**	**70.02**	**71.23**	**71.28**	**73.34**	**73.09**	**73.82**
		Berkeley parser, self-trained (FIRST)						
precision	65.79***	68.20***	70.15**	71.02	71.03	72.23	73.20	73.21
recall	65.46***	67.69***	69.71*	70.47	70.72	71.82	72.55	72.71
f-score	65.63	67.94	69.93	70.74	70.87	72.03	72.88	72.96

Table 1: Parsing results (PARSEVAL) for the different self-training settings, including GF in the evaluation (asterisks indicate significant differences between self-training and the baseline: p=0.001***, p=0.005**, p=0.01*, p=0.05 .)

including erroneous parser output trees.

So far, it is not clear to us why the lexicalised parser performs poorly in the self-training setting. This result is in line with (Huang and Harper, 2009), who observed that the PCFG-LA parser used in their experiments benefitted more from self-training as compared to a lexicalised generative parser. However, our results are not necessarily an effect of lexicalisation, but might be due to the overall lower accuracy of the Stanford parser on German (see Kübler (2008)). A quantiative and qualitative error analysis might give us some interesting insight into the underlying reasons and into the question when and why self-training will work for parsing.

4 Conclusions and future work

We presented preliminary results on self-training experiments for German, a language with rich morphology and semi-free word order. We proposed a new approach to self-training where we select new instances on the basis of similarity to the seed training data. Our results show that this strategy helps to boost self-training results especially for small seed data, but also obtains a significant improvement for larger training sets.

Our approach offers plenty of room for improvement. In future work we plan to investigate the adequacy of different similarity measures for self-training, and also to measure similarity on different levels (so far we have only considered the part-of-speech level). An obvious extension is the integration of a reranker in order to add a different view on the selection process. We expect that this will have a positive impact on our results.

Finally, we plan to have a closer look at the impact of language-specific properties on self-training. Our intuition is that the potential of self-training might be larger for morphologically rich languages, but this claim has yet to be tested.

Acknowledgments

We would like to thank the anonymous reviewers for their helpful comments and suggestions.

References

Michiel Bacchiani, Michael Riley, Brian Roark, and Richard Sproat. 2006. Map adaptation of stochastic grammars. *Computer Speech and Language*, 20.

Sabine Brants, Stefanie Dipper, Silvia Hansen, Wolfgang Lezius, and George Smith. 2002. The TIGER treebank. In *Proceedings of the First Workshop on Treebanks and Linguistic Theories*.

Eugene Charniak and Mark Johnson. 2005. Coarse-to-fine n-best parsing and maxent discriminative reranking. In *The 43rd Meeting of the Association for Computational Linguistics (ACL)*.

Eugene Charniak. 1997. Statistical parsing with a context-free grammar and word statistics. In *Proceedings of the Fourteenth National Conference on Artificial Intelligence (AAAI)*.

Amit Dubey. 2004. *Statistical Parsing for German: Modeling syntactic properties and annotation differences*. Ph.D. thesis, Saarland University, Germany.

Jennifer Foster, Joachim Wagner, Djamé Seddah, and Josef van Genabith. 2007. Adapting WSJ-trained parsers to the British National Corpus using in-domain self-training. In *Proceedings of the Tenth International Workshop on Parsing Technologies (IWPT)*.

Zhongqiang Huang and Mary Harper. 2009. Self-training pcfg grammars with latent annotations across languages. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, EMNLP '09.

Dan Klein and Chris Manning. 2003. Accurate unlexicalized parsing. In *The 41st Annual Meeting of the Association for Computational Linguistics (ACL)*.

Sandra Kübler. 2005. How do treebank annotation schemes influence parsing results? or how not to compare apples and oranges. In *Proceedings of the 5th International Conference on Recent Advances in Natural Language Processing (RANLP)*.

Sandra Kübler. 2008. The page 2008 shared task on parsing german. In *Proceedings of the ACL Workshop on Parsing German*, Columbus, Ohio.

David McClosky, Eugene Charniak, and Mark Johnson. 2006a. Effective self-training for parsing. In *Human Language Technology conference - North American chapter of the Association for Computational Linguistics annual meeting (HLT-NAACL)*.

David McClosky, Eugene Charniak, and Mark Johnson. 2006b. Reranking and self-training for parser adaptation. In *The 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics (ACL-COLING)*.

David McClosky, Eugene Charniak, and Mark Johnson. 2008. When is self-training effective for parsing? In *Proceedings of COLING*.

David McClosky, Eugene Charniak, and Mark Johnson. 2010. Automatic domain adaptation for parsing. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*.

Slav Petrov, Leon Barrett, Romain Thibaux, and Dan Klein. 2006. Learning accurate, compact, and interpretable tree annotation. In *The 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics (ACL-COLING)*.

Ines Rehbein and Josef van Genabith. 2007. Why is it so difficult to compare treebanks? TIGER and TüBa-D/Z revisited. In *Proceedings of the Sixth International Workshop on Treebanks and Linguistic Theories*.

Roi Reichart and Ari Rappoport. 2007. Self-training for enhancement and domain adaptation of statistical parsers trained on small datasets. In *The 45th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Anne Schiller, Simone Teufel, and Christine Thielen. 1995. Guidelines für das Tagging deutscher Textcorpora mit STTS. Technical report, IMS-CL, University Stuttgart, Germany.

Helmut Schmid and Florian Laws. 2008. Estimation of conditional probabilities with decision trees and an application to fine-grained pos tagging. In *Proceedings of the 22nd International Conference on Computational Linguistics*, COLING-08.

Wolfgang Seeker, Ines Rehbein, Jonas Kuhn, and Josef van Genabith. 2010. Hard constraints for grammatical function labelling. In *The 48th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Anders Søgaard. 2011. Data point selection for cross-language adaptation of dependency parsers. In *The 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL-HLT)*.

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 *The 11th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*.

Heike Telljohann, Erhard W. Hinrichs, Sandra Kübler, and Heike Zinsmeister. 2005. Stylebook for the Tübingen Treebank of Written German (TüBa-D/Z). Technical report, University Tübingen, Germany.

Daniel Zeman and Philip Resnik. 2008. Cross-language parser adaptation between related languages. In *Proceedings of IJCNLP 2008 Workshop on NLP for Less Privileged Languages*.