UD Treebank Sampling for Comparative Parser Evaluation

Miryam de Lhoneux and Joakim Nivre

Uppsala University
Department of Linguistics and Philology
{miryam.de_lhoneux,joakim.nivre}@lingfil.uu.se

Abstract

In this abstract, we attempt to address the problem that although the Universal Dependencies project makes it possible to evaluate parsers on a large variety of languages and domains, it is difficult to do that efficiently for two reasons. First, UD is growing rapidly and second, the new state-of-the-art parsing methods that make use of neural networks are expensive to optimize. We propose to incrementally evaluate parsers on small to large selections of the treebanks. We take a first step in testing this method by attempting a comparison of transition-based parsers with and without neural network enhancement and make first tentative observations on the results.

1. Introduction

Treebanks have recently been released for a large number of languages in a consistent annotation within the framework of the Universal Dependencies (UD) project (Nivre et al., 2016). With a variety of languages and domains, this project may help reshape the field of syntactic parsing which has long been dominated by research on one language and one domain: the English Penn Treebank (PTB). Simultaneously to the development of this project, syntactic parsing has seen a significant boost in accuracy in the last couple of years with methods that make use of neural networks to learn dense vectors for words, POS tags and dependency relations (Chen and Manning, 2014; Weiss et al., 2015; Andor et al., 2016) or stacks (Dyer et al., 2015).

Having a large and varied data set has the advantage that our parsing models will generalize better. A disadvantage is that it is expensive to train models for all the languages, especially with the new neural network models that need a search over a large hyperparameter space in order to be optimized. As UD grows, it may become more and more prohibitive to train models for all the languages when we want to evaluate how a parser does as opposed to another or as opposed to a modified version of itself. In this abstract, we would like to discuss how we can overcome this problem. In particular, we propose to incrementally evaluate parsers on small to large selections of the treebanks.

Motivated by the success of neural network models on the PTB and the Chinese treebank, Straka et al. (2015) trained Parsito, a neural network model, on UD treebanks and obtained good results, improving over their 'classical' counterpart MaltParser (Nivre et al., 2007). There was however no systematic comparison between the classical and the neural network approach and it may be that one approach is more suitable than another for specific settings. We propose to use our method of sampling treebanks to attempt a comparison of the two models. We take a first step in this direction by presenting preliminary results of the two parsing methods on a small sample of treebanks.

The aim of this abstract is thus twofold, first we discuss how to do comparative parser evaluation in UD parsing by sampling the selection of treebanks and second, we use this method to analyse the performance of two models on a small, but hopefully representative, selection of treebanks.

2. UD treebanks sampling

As said in the introduction, if we want to evaluate a parsing model, we might want to avoid training all models on all treebanks. It is especially true for neural network models which are expensive to optimize. We therefore propose that it might be wise to examine their behavior in a small-scaled setting before training them for the large number of treebanks in UD. We propose that parsing models can first be evaluated on a small sample of UD treebanks. Subsequently, depending on the observations on the small set, we can move to a medium sample before finally testing on all the treebanks if evidence points towards a clear direction. We have come up with a set of criteria to select the small sample which we now turn to.

The objective was to have a sample as representative of the whole treebank set as possible. To ensure typological variety we divided UD languages into coarse-grained and fine-grained language families. This led to a total of 15 different fine-grained families and 8 coarse-grained. We made it a requirement to not select two languages from the same fine-grained family and ensured to have some variety in coarse-grained families. We made sure to have at least one isolating, one morphologically-rich and one inflecting language. We additionally ensured a variability of treebank sizes and domains. Since parsing non-projective trees is notoriously harder than parsing projective trees, we also made sure to have at least one treebank with a large amount of non-projective trees. The quality of treebanks was also considered in the selection, in particular, there are known issues[1] about inconsistency in the annotation. We selected languages that had as little of those as possible. To ensure comparability, we finally made sure to select only treebanks with morphological features (with one exception for Kazakh). This resulted in a selection of 8 treebanks. The selection is given in Table 1 together with main arguments for inclusion for each.

[1]http://universaldependencies.org/svalidation.html

Table 1: Treebank Sample

	coarse	fine	main argument for inclusion
Czech	indo-european	balto-slavic	the largest UD treebank
Chinese	sino-tibetan	sinitic	the only isolating language (without copyrights)
Finnish	uralo-altaic	finno-ungric	morphologically rich; has many different domains
English	indo-european	germanic	largest treebank with full manual check of the data
Ancient_Greek-PROIEL	indo-european	hellenic	large percentage of non-projective trees
Kazakh	uralo-altaic	turkic	smallest treebank; full manual check of the annotations
Tamil	dravidian	tamil	small treebank; language family considerations
Hebrew	afro-asiatic	semitic	morphologically rich/language family considerations

Table 2: Parsing models comparison (LAS).
*The result for Czech is only with improvements from phase 1 out of the 3 phases due to memory heap issues

	tokens	MaltParser	MaltOptimizer	UDPipe	SyntaxNet
Anc. Greek-PR	206K	59.90	67.40	69.60	**73.15**
Chinese	123K	64.80	68.10	-	**71.24**
Czech	1,503K	77.20	78.40*	82.60	**85.93**
English	254K	77.50	79.60	**80.60**	80.38
Finnish	181K	60.30	73.20	76.50	**79.60**
Hebrew	115K	74.10	74.90	76.80	**78.71**
Kazakh	4K	**49.60**	49.10	-	43.95
Tamil	8K	58.70	**59.90**	58.50	55.35

3. Comparing Transition-based Parsers with and without Neural Network enhancement

As said in the introduction, a fair comparison of classical models for transition-based parsing as opposed to models enhanced by neural network training is lacking. Straka et al. (2015) reported results with MaltParser but using it with default settings. If optimized, results obtained with MaltParser can be significantly higher. For this reason, we used MaltOptimizer (Ballesteros and Nivre, 2016). In Table 2, we report those results together with MaltParser results with default settings. We additionally compare those results with Parsito. For Parsito, we used the pretrained models that the authors made available. They are trained on UD version 1.2 but we tested them on version 1.3 since that is the version used for the other parsers. We also add the best reported results for SyntaxNet (Andor et al., 2016). UDPipe (Straka et al., 2016) tagger models were trained for all treebanks and both MaltParser and Parsito were tested on the tagged test set. Note that SyntaxNet results are not directly comparable as they use their own tagger.

Results are given in Table 2. As appears from the table, SyntaxNet is largely the best model. However, even with default settings, MaltParser performs better than neural network enhanced parsers on very small data sets. With bigger data sets, MaltParser with default settings largely lags behind the other models. With automatically optimized settings, however, those models are much closer to neural network enhanced models.

In future work, we hope to further investigate the impact of training size on neural network parsers. Looking at these results, we can as a matter of fact hypothesize that the success of neural network parsers over MaltParser depends mostly on the size of the training set.

4. Conclusion and Future Work

In this abstract, we have given some points of discussion on how to do parser evaluation for parsing with UD as it grows and taking into account the fact that current state-of-the-art models are expensive to optimize. We have proposed to do incremental parsing model evaluation on small, medium and large samples of the treebanks as well as proposed a small selection with well-defined criteria. We have also emphasized the value of comparing neural network models to the more classical approach and preliminary results further supported that. Preliminary results pointed to the importance of training size for the neural network models. In the future, we hope to compare the two models further and investigate for example the impact of training size on each model.

References

Daniel Andor, Chris Alberti, David Weiss, Aliaksei Severyn, Alessandro Presta, Kuzman Ganchev, Slav Petrov, and Michael Collins. 2016. Globally normalized transition-based neural networks. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 1: Long Papers*.

Miguel Ballesteros and Joakim Nivre. 2016. MaltOptimizer: Fast and effective parser optimization. *Natural Language Engineering*, 22(2):187–213.

Danqi Chen and Christopher D Manning. 2014. A fast and accurate dependency parser using neural networks. In *Empirical Methods in Natural Language Processing (EMNLP)*.

Chris Dyer, Miguel Ballesteros, Wang Ling, Austin Matthews, and Noah A. Smith. 2015. Transition-based dependency parsing with stack long short-term memory. In *Proceedings of the 53rd Annual Meeting of the Associ-*

ation for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), pages 334–343, Beijing, China, July. Association for Computational Linguistics.

Joakim Nivre, Johan Hall, Jens Nilsson, Atanas Chanev, Gülşen Eryiğit, Sandra Kübler, Svetoslav Marinov, and Erwin Marsi. 2007. MaltParser: A language-independent system for data-driven dependency parsing. *Natural Language Engineering*, 13(2):95–135.

Joakim Nivre, Marie-Catherine de Marneffe, Filip Ginter, Yoav Goldberg, Jan Hajic, Christopher D Manning, Ryan McDonald, Slav Petrov, Sampo Pyysalo, Natalia Silveira, et al. 2016. Universal dependencies v1: A multilingual treebank collection. In *Proceedings of the 10th International Conference on Language Resources and Evaluation (LREC 2016)*.

Milan Straka, Jan Hajič, Jana Straková, and Jan Hajič jr. 2015. Parsing universal dependency treebanks using neural networks and search-based oracle. In *Proceedings of Fourteenth International Workshop on Treebanks and Linguistic Theories (TLT 14)*, December.

Milan Straka, Jan Hajič, and Straková. 2016. UDPipe: trainable pipeline for processing CoNLL-U files performing tokenization, morphological analysis, pos tagging and parsing. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, Paris, France, May. European Language Resources Association (ELRA).

David Weiss, Chris Alberti, Michael Collins, and Slav Petrov. 2015. Structured training for neural network transition-based parsing. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 323–333, Beijing, China, July. Association for Computational Linguistics.