Parameter sharing between dependency parsers for related languages

Miryam de Lhoneux[1*] **Johannes Bjerva**[2] **Isabelle Augenstein**[2] **Anders Søgaard**[2]
[1]Department of Linguistics and Philology
Uppsala University
Uppsala, Sweden
[2] Department of Computer Science
University of Copenhagen
Copenhagen, Denmark

Abstract

Previous work has suggested that parameter sharing between transition-based neural dependency parsers for related languages can lead to better performance, but there is no consensus on what parameters to share. We present an evaluation of 27 different parameter sharing strategies across 10 languages, representing five pairs of related languages, each pair from a different language family. We find that sharing transition classifier parameters always helps, whereas the usefulness of sharing word and/or character LSTM parameters varies. Based on this result, we propose an architecture where the transition classifier is shared, and the sharing of word and character parameters is controlled by a parameter that can be tuned on validation data. This model is linguistically motivated and obtains significant improvements over a mono-lingually trained baseline. We also find that sharing transition classifier parameters helps when training a parser on unrelated language pairs, but we find that, in the case of unrelated languages, sharing too many parameters does not help.

1 Introduction

The idea of sharing parameters between parsers of related languages goes back to early work in cross-lingual adaptation (Zeman and Resnik, 2008), and the idea has recently received a lot of interest in the context of neural dependency parsers (Duong et al., 2015; Ammar et al., 2016; Susanto and Lu, 2017). Modern neural dependency parsers, however, use different sets of parameters for representation and scoring, and it is not clear what parameters it is best to share.

The Universal Dependencies (UD) project (Nivre et al., 2016), which is seeking to harmonize the annotation of dependency treebanks across languages, has seen a steady increase in languages that have a treebank in a common standard. Many of these languages are low resource and have small UD treebanks. It seems interesting to find out ways to leverage the wealth of information contained in these treebanks, especially for low resource languages.

In this paper, we evaluate 27 different parameter sharing strategies. We focus on a particular transition-based neural dependency parser (de Lhoneux et al., 2017a,b), which performs close to the state of the art. This parser has three sets of parameters: i) the parameters of a character-based one-layer, bidirectional LSTM; ii) the parameters of a word-based two-layer, bidirectional LSTM; iii) and the parameters of a multi-layered perceptron (MLP) with a single hidden layer. The two first sets are for learning to represent configurations; the third for selecting the next transition. We consider all combinations of sharing these sets of parameters; and in addition, we consider two ways of sharing each set of parameters, namely with and without a prefixed language embedding. The latter enables partial, soft sharing. In sum, we consider all 3^3 combinations of no sharing, hard sharing and soft sharing of the three sets of parameters. We evaluate the 27 multilingual parsers on 10 languages from the UD project, representing five pairs of related languages, each pair from a different language family. We repeat the experiment with the same set of languages, but using pairs of unrelated languages.

Contributions This paper is, to the best of our knowledge, the first to evaluate different parameter sharing strategies for exploiting synergies between neural dependency parsers of related languages. We evaluate the different strategies on 10 languages, representing five different language families. We find that sharing (MLP) transition

[*] Work carried out during a stay at the University of Copenhagen.

Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, pages 4992–4997
Brussels, Belgium, October 31 - November 4, 2018.

Lang	Tokens	Family	Word order
ar	208,932	Semitic	VSO
he	161,685	Semitic	SVO
et	60,393	Finnic	SVO
fi	67,258	Finnic	SVO
hr	109,965	Slavic	SVO
ru	90,170	Slavic	SVO
it	113,825	Romance	SVO
es	154,844	Romance	SVO
nl	75,796	Germanic	No dom. order
no	76,622	Germanic	SVO

Table 1: Dataset characteristics

classifier parameters always helps, whereas the usefulness of sharing LSTM parameters depends on the language pair. This reflects the intuition that the transition classifier learns hierarchical structures that are likely to transfer across languages, based on parser configurations that abstract away from several linguistic differences. The similarity of the input to character- and word-level LSTMs, on the other hand, will vary depending on the phonological and morphosyntactic similarity of the languages in question. Motivated by this observation, we propose an architecture with hard-wired transition classifier parameter sharing, but in which sharing of LSTM parameters is tuned. The novel architecture significantly outperforms our monolingual baseline on our set of 10 languages. We additionally investigate parameter sharing of unrelated languages.

2 The Uppsala dependency parser

The Uppsala parser (de Lhoneux et al., 2017a,b) consists of three sets of parameters; the parameters of the character-based LSTM, those of the word-based LSTM, and the parameters of the MLP that predicts transitions. The character-based LSTM produces representations for the word-based LSTM, which produces representations for the MLP. The Uppsala parser is a transition-based parser (Kiperwasser and Goldberg, 2016), adapted to the Universal Dependencies (UD) scheme,[1] and using the arc-hybrid transition system from Kuhlmann et al. (2011) extended with a SWAP transition and a static-dynamic oracle, as described in de Lhoneux et al. (2017b). The SWAP transition is used to generate non-projective dependency trees (Nivre, 2009).

For an input sentence of length n with words $w_1, \ldots, w_n$, the parser creates a sequence of vectors $x_{1:n}$, where the vector x_i representing w_i is the concatenation of a word embedding and the final state of the character-based LSTM after processing the characters of w_i. The character vector $ch(w_i)$ is obtained by running a (bi-directional) LSTM over the characters ch_j ($1 \leq j \leq m$) of w_i. Each input element is represented by the word-level, bi-directional LSTM, as a vector $v_i = \text{BILSTM}(x_{1:n}, i)$. For each configuration, the feature extractor concatenates the LSTM representations of core elements from the stack and buffer. Both the embeddings and the LSTMs are trained together with the model.

A configuration c is represented by a feature function $\phi(\cdot)$ over a subset of its elements. For each configuration, transitions are scored by a classifier, in this case an MLP, and $\phi(\cdot)$ is a concatenation of BiLSTM vectors on top of the stack and the beginning of the buffer. The MLP scores transitions together with the arc labels for transitions that involve adding an arc. In practice, we use two interpolated MLPs, one which only scores the transitions, and one which scores transitions together with the arc label. For simplicity, we refer to that interpolated MLP as the MLP.

3 Parameter sharing

Since our parser has three basic sets of model parameters, we consider sharing all combinations of those three sets. We also introduce two ways of sharing, namely, with or without the addition of a vector representing the language. This language embedding enables the model, in theory, to learn what to share between the two languages in question. Since for all three model parameter sets, we now have three options – not sharing, sharing, or sharing in the context of a language embedding – we are left with $3^3 = 27$ parameter sharing strategies; see Table 2.

In the setting where we do not share (✗) word parameters (**W**), we construct a different word lookup table and a different word-level BiLSTM for each language. In the setting where we do hard parameter sharing (✓) of word parameters, we only construct one lookup table and one word BiLSTM for the languages involved. In the setting where we do soft sharing (ID) of word pa-

[1] http://universaldependencies.org/

Model	C	W	S	ar	he	es	it	et	fi	nl	no	hr	ru	Av
MONO				76.3	80.2	83.7	83.3	70.4	70.8	77.3	80.8	76.8	82.3	78.2
LANGUAGE-BEST				76.6	80.6	84.4	84.8	72.8	72.9	79.6	82.1	78.0	82.9	79.5
BEST	✗	✓	ID	76.3	80.3	84.2	84.5	72.1	72.5	78.8	81.4	77.6	82.8	79.1
CHAR	✓	✗	✗	76.4	80.3	84.3	84.0	72.3	71.0	78.3	81.3	77.0	82.3	78.7
WORD	✗	✓	✗	76.3	79.9	83.9	84.4	72.4	71.3	77.4	80.7	76.9	82.5	78.6
STATE	✗	✗	✓	76.6	80.3	84.0	83.7	71.5	72.9	78.3	81.5	77.4	82.8	78.9
	⋮													
ALL	✓	✓	✓	76.2	80.1	84.0	84.2	72.1	71.4	78.7	81.1	77.0	82.5	78.7
SOFT	ID	ID	ID	76.3	79.9	84.1	84.4	72.1	71.3	79.6	81.4	77.1	82.5	78.9

Table 2: Performance on development data (LAS; in %) across select sharing strategies. MONO is our single-task baseline; LANGUAGE-BEST is using the best sharing strategy for each language (as evaluated on development data); BEST is the overall best sharing strategy, across languages; CHAR shares only the character-based LSTM parameters; WORD shares only the word-based LSTM parameters; ALL shares all parameters. ✓ refers to hard sharing, ID refers to soft sharing, using an embedding of the language ID and ✗ refers to not sharing.

rameters, we share those parameters, and in addition, concatenate a language embedding l_i representing the language of word w_i to the vector of the word w_i at the input of the word BiLSTM: $x_i = e(w_i) \circ ch(w_i) \circ l_i$. Similarly for character parameters (**C**), we construct a different character BiLSTM and one character lookup for each language (✗), create those for all languages and share them (✓) or share them and concatenate a (randomly initialized) language embedding l_i representing the language of word w_i at the input of the character BiLSTM (ID): $ch_j = e(ch_j) \circ l_i$. At the level of configuration or parser states (**S**), we either construct a different MLP for each language (✗), share the MLP (✓) or share it and concatenate a language embedding l_i representing the language of word w_i to the vector representing the configuration, at the input of the MLP (ID): $c = \phi(\cdot) \circ l_i$.

4 Experiments

Language pairs We use 10 languages in our experiments, representing five language pairs from different language families. Our two SEMITIC languages are Arabic and Hebrew. These two languages differ in that Arabic tends to favour VSO word order whereas Hebrew tends to use SVO, but are similar in their rich transfixing morphology. Our two FINNO-UGRIC languages are Estonian and Finnish. These two languages differ in that Estonian no longer has vowel harmony, but share a rich agglutinative morphology. Our two SLAVIC languages are Croatian and Russian. These two languages differ in that Croatian uses gender in plural nouns, but otherwise share their rich inflectional morphology. Our two ROMANCE languages are Italian and Spanish. These two languages differ in that Italian uses a possessive adjective with a definite article, but share a fairly strict SVO order. Finally, our two GERMANIC languages are Dutch and Norwegian. These two languages differ in morphological complexity, but share word ordering features to some extent.

Datasets For all 10 languages, we use treebanks from the Universal Dependencies project. The dataset characteristics are listed in Table 1. To keep the results comparable across language pairs, we down-sample the training set to the size of the smallest of our languages, Hebrew: we randomly sample 5000 sentences for each training set. Note that while this setting makes the experiment somewhat artificial and will probably overestimate the benefits that can be obtained from sharing parameters when using larger treebanks, we find it interesting to see how much low resource languages can benefit from parameter sharing, as explained in the introduction.

Baselines and systems This is an evaluation paper, and our results are intended to explore a space of sharing strategies to find better ways of sharing parameters between dependency parsers of related languages. Our baseline is the Uppsala parser trained monolingually. Our systems are parsers trained bilingually by language pair where

we share subsets of parameters between the languages in the pair, and we report on what sharing strategies seem superior across the 10 languages that we consider.

Implementation details A flexible implementation of parameter strategies for the Uppsala parser was implemented in Dynet.[2] We make the code publicly available.[3]

5 Results and discussion

Our results on development sets are presented in Table 2. We use labeled attachment score (LAS) as our metric for evaluating parsers. Table 2 presents numbers for a select subset of the 27 sharing strategies. The other results can be found in the supplementary material. Our main observations are: **(i)** that, generally, and as observed in previous work, *multi-task learning helps*: all different sharing strategies are on average better than the monolingual baselines, with minor (0.16 LAS points) to major (0.86 LAS points) average improvements; and **(ii)** that sharing the MLP seems to be overall a better strategy than not sharing it: the 10 best strategies share the MLP. Whereas the usefulness of sharing the MLP seems to be quite robust across language pairs, the usefulness of sharing word and character parameters seems more dependent on the language pairs. This reflects the linguistic intuition that character- and word-level LSTMs are highly sensitive to phonological and morphosyntactic differences such as word order, whereas the MLP learns to predict less idiosyncratic, hierarchical relations from relatively abstract representations of parser configurations.

Based on this result, we propose a model (OURS) where *the MLP is shared and the sharing of word and character parameters is controlled by a parameter that can be set on validation data*. Results are given in Table 3. We obtain a 0.6 LAS improvement on average and our proposed model is significantly better than the monolingual baseline with $p < 0.01$. Significance testing is performed using a randomization test, with the script from the CoNLL 2017 Shared Task.[4]

[2] https://github.com/clab/dynet

[3] https://github.com/coastalcph/uuparser

[4] https://github.com/udapi/udapi-python/blob/master/udapi/block/eval/conll17.py

	W	C	OURS	MONO	δ
ar	✗	✗	77.2	77.1	0.1
es	ID	✓	84.3	83.8	0.5
et	✗	ID	71.4	70.5	0.8
fi	✗	✗	71.6	71.6	0.1
he	✓	✗	80.0	79.8	0.3
hr	✓	✗	77.9	78.0	-0.1
it	ID	✓	85.0	84.0	1.0
nl	ID	✓	75.5	74.1	1.4
no	✗	ID	81.1	80.1	1.0
ru	✓	✗	83.5	82.7	0.8
av.			78.8	78.2	0.6

Table 3: LAS on the test sets of the best of 9 sharing strategies and the monolingual baseline. δ is the difference between OURS AND MONO.

6 Unrelated languages

We repeated the same set of experiments with unrelated language pairs. We hypothesise that parameter sharing between unrelated language pairs will be less useful in general than with related language pairs. However, it can still be useful, it has been shown previously that unrelated languages can benefit from being trained jointly. For example, Lynn et al. (2014) have shown that Indonesian was surprisingly particularly useful for Irish.

The results are presented in Table 4. The table only presents part of the results, the rest can be found in the supplementary material. As expected, there is much less to be gained from sharing parameters between unrelated pairs. However, it is possible to improve the monolingual baseline by sharing some of the parameters. In general, sharing the MLP is still a helpful thing to do. It is most helpful to share the MLP and optionally one of the two other sets of parameters. Results are close to the monolingual baseline when everything is shared. Sharing word and character parameters but not the MLP hurts accuracy compared to the monolingual baseline.

7 Related work

Previous work has shown that sharing parameters between dependency parsers for related languages can lead to improvements (Duong et al., 2015; Ammar et al., 2016; Susanto and Lu, 2017). Smith et al. (2018) recently found that sharing parameters using the same parser as in this paper

Model	C	W	S	he	no	fi	hr	ru	es	it	et	nl	ar	Av
MONO				80.2	80.8	70.8	76.8	82.3	83.7	83.3	70.4	77.3	76.3	78.2
LANGUAGE-BEST				80.5	81.5	71.9	77.6	82.9	84.0	84.3	72.5	78.7	76.5	78.9
BEST	✗	✗	✓	80.3	81.5	71.9	77.6	82.7	84.0	83.8	72.5	78.7	76.3	78.9
WORST	ID	ID	✗	79.8	80.6	69.2	76.7	81.4	83.8	83.2	69.4	76.6	76.0	77.7
CHAR	✓	✗	✗	80.1	80.9	71.4	76.8	82.9	83.9	84.3	70.9	78.0	76.5	78.6
WORD	✗	✓	✗	79.6	80.9	71.9	76.9	82.2	83.7	83.8	70.9	77.0	76.4	78.3
ALL	✓	✓	✓	80.5	80.9	69.8	76.6	82.3	83.7	84.0	70.6	77.4	76.2	78.2
SOFT	ID	ID	ID	79.8	80.5	70.1	76.6	82.1	83.9	83.8	70.6	77.2	76.3	78.1

Table 4: Performance on development data (LAS; in %) across select sharing strategies for unrelated languages. MONO is our single-task baseline; LANGUAGE-BEST is using the best sharing strategy for each language (as evaluated on development data); BEST and WORST are the overall best and worst sharing strategy across languages; CHAR shares only the character-based LSTM parameters; WORD shares only the word-based LSTM parameters; ALL shares all parameters. ✓ refers to hard sharing, ID refers to soft sharing, using an embedding of the language ID and ✗ refers to not sharing.

(soft sharing of word parameters, hard sharing of the rest) improves parsing accuracy when training on related languages, and is especially useful in the low resource case. Similar effects have been observed in machine translation (Dong et al., 2015; Johnson et al., 2017), for example. Most studies have only explored a small number of parameter sharing strategies, however. Vilares et al. (2016) evaluate parsing with hard parameter sharing for 100 language pairs with a statistical parser. Naseem et al. (2012) proposed to selectively share subsets of a parser across languages in the context of a probabilistic parser.

Options we do not explore here are learning the architecture jointly with optimizing the task objective (Misra et al., 2016; Ruder et al., 2017), or learning an architecture search model that predicts an architecture based on the properties of datasets, typically with reinforcement learning (Zoph and Le, 2017; Wong and Gesmundo, 2018; Liang et al., 2018). We also do not explore the option of sharing selectively based on more fine-grained typological information about languages, which related work has indicated could be useful (Bjerva and Augenstein, 2018). Rather, we stick to sharing between languages of the same language families.

The strategies explored here do not exhaust the space of possible parameter sharing strategies. For example, we completely ignore soft sharing based on mean-constrained regularisation (Duong et al., 2015).

8 Conclusions

We present evaluations of 27 parameter sharing strategies for the Uppsala parser across 10 languages, representing five language pairs from five different language families. We repeated the experiment with pairs of unrelated languages. We made several observations: (a) Generally, multi-task learning helps. (b) Sharing the MLP parameters always helps. It helps to share MLP parameters when training a parser on a pair of related languages, and it also helps if the languages are unrelated. (c) Sharing word and character parameters is differently helpful depending on the language. (d) Sharing too many parameters does not help, when the languages are unrelated.

In future work, we plan to investigate what happens when training on more than 2 languages. Here, we focused on a setting with rather small amounts of balanced data. It would be interesting to experiment with using datasets that are not balanced with respect to size. Finally, we have restricted our experiments to a specific architecture, using fixed hyperparameters including word and character embedding dimensions. It would be interesting to experiment with different parsing architectures as well as varying those hyperparameters.

Acknowledgments

We acknowledge the computational resources provided by CSC in Helsinki and Sigma2 in Oslo through NeIC-NLPL (www.nlpl.eu). The second author is partially funded by an AdeptMind scholarship. The last author was funded by an ERC Starting Grant.

References

Waleed Ammar, George Mulcaire, Miguel Ballesteros, Chris Dyer, and Noah A. Smith. 2016. More languages, one parser. In *TACL*.

Johannes Bjerva and Isabelle Augenstein. 2018. From Phonology to Syntax: Unsupervised Linguistic Typology at Different Levels with Language Embeddings. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 907–916. Association for Computational Linguistics.

Daxiang Dong, Hua Wu, Wei He, Dianhai Yu, and Haifeng Wang. 2015. Multi-task learning for multiple language translation. In *ACL*.

Long Duong, Trevor Cohn, Steven Bird, and Paul Cook. 2015. Low Resource Dependency Parsing: Cross-lingual Parameter Sharing in a Neural Network Parser. In *Proceedings of ACL*.

Melvin Johnson, Mike Schuster, Quoc V. Le, Maxim Krikun, Yonghui Wu, Zhifeng Chen, Nikhil Thorat, Fernanda Vigas, Martin Wattenberg, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2017. Google's neural machine translation system. In *TACL*.

Eliyahu Kiperwasser and Yoav Goldberg. 2016. Simple and Accurate Dependency Parsing Using Bidirectional LSTM Feature Representations. *TACL*, 4:313–327.

Marco Kuhlmann, Carlos Gómez-Rodríguez, and Giorgio Satta. 2011. Dynamic Programming Algorithms for Transition-Based Dependency Parsers. In *Proceedings of ACL*, pages 673–682, Portland, Oregon, USA.

Miryam de Lhoneux, Yan Shao, Ali Basirat, Eliyahu Kiperwasser, Sara Stymne, Yoav Goldberg, and Joakim Nivre. 2017a. From raw text to universal dependencies - look, no tags! In *Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 207–217, Vancouver, Canada.

Miryam de Lhoneux, Sara Stymne, and Joakim Nivre. 2017b. Arc-Hybrid Non-Projective Dependency Parsing with a Static-Dynamic Oracle. In *Proceedings of the 15th International Conference on Parsing Technologies*, pages 99–104, Pisa, Italy.

Jason Liang, Elliot Meyerson, and Risto Miikkulainen. 2018. Evolutionary Architecture Search For Deep Multitask Networks. In *GECCO*.

Teresa Lynn, Jennifer Foster, Mark Dras, and Lamia Tounsi. 2014. Cross-lingual transfer parsing for low-resourced languages: An irish case study. In *Proceedings of the First Celtic Language Technology Workshop*, pages 41–49.

Ishan Misra, Abhinav Shrivastava, Abhinav Gupta, and Martial Hebert. 2016. Cross-Stitch Networks for Multi-Task Learning. In *Proceedings of CVPR*.

Tahira Naseem, Regina Barzilay, and Amir Globerson. 2012. Selective sharing for multilingual dependency parsing. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 629–637. Association for Computational Linguistics.

Joakim Nivre. 2009. Non-Projective Dependency Parsing in Expected Linear Time. In *Proceedings of ACL*, pages 351–359, Suntec, Singapore.

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)*.

Sebastian Ruder, Joachim Bingel, Isabelle Augenstein, and Anders Søgaard. 2017. Sluice networks: Learning what to share between loosely related tasks. In *CoRR, abs/1705.08142*.

Aaron Smith, Bernd Bohnet, Miryam de Lhoneux, Joakim Nivre, Yan Shao, and Sara Stymne. 2018. 82 Treebanks, 34 Models: Universal Dependency Parsing with Multi-Treebank Models. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*.

Raymond Hendy Susanto and Wei Lu. 2017. Neural architectures for multilingual semantic parsing. In *ACL*.

David Vilares, Carlos Gómez-Rodríguez, and Miguel A Alonso. 2016. One model, two languages: training bilingual parsers with harmonized treebanks. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, volume 2, pages 425–431.

Catherine Wong and Andrea Gesmundo. 2018. Transfer Learning to Learn with Multitask Neural Model Search. In *ICPR*.

Daniel Zeman and Philip Resnik. 2008. Cross-Language Parser Adaptation between Related Languages. In *IJCNLP*.

Barret Zoph and Quoc Le. 2017. Neural Architecture Search with Reinforcement Learning. In *ICPR*.