Climbing the Tower of Treebanks: Improving Low-Resource Dependency Parsing via Hierarchical Source Selection

Goran Glavaš
University of Mannheim
Data and Web Science Group
goran@informatik.uni-mannheim.de

Ivan Vulić
University of Cambridge
Language Technology Lab
iv250@cam.ac.uk

Abstract

Recent work on multilingual dependency parsing focused on developing highly multilingual parsers that can be applied to a wide range of low-resource languages. In this work, we substantially outperform such "one model to rule them all" approach with a heuristic selection of languages and treebanks on which to train the parser for a specific target language. Our approach, dubbed TOWER, first hierarchically clusters all Universal Dependencies languages based on their mutual syntactic similarity computed from human-coded URIEL vectors. For each low-resource target language, we then climb this language hierarchy starting from the leaf node of that language and heuristically choose the hierarchy level at which to collect training treebanks. This treebank selection heuristic is based on: (i) the aggregate size of all treebanks subsumed by the hierarchy level and (ii) the similarity of the languages in the training sample with the target language. For languages without development treebanks, we additionally use (ii) for model selection (i.e., early stopping) in order to prevent overfitting to development treebanks of closest languages. Our TOWER approach shows substantial gains for low-resource languages over two state-of-the-art multilingual parsers, with more than 20 LAS point gains for some of those languages. Parsing models and code available at: https://github.com/codogogo/towerparse.

1 Introduction

Syntactic parsing – grounded in a wide variety of formalisms (Taylor et al., 2003; De Marneffe et al., 2006; Hockenmaier and Steedman, 2007; Nivre et al., 2016, *inter alia*) – has been the backbone of natural language processing (NLP) for decades, and an indispensable preprocessing step for tackling higher-level language understanding tasks. A recent major paradigm shift in NLP towards large-scale pretrained language models (PLMs) (Devlin et al., 2019; Liu et al., 2019; Brown et al., 2020) and their end-to-end fine-tuning for downstream tasks has reduced the downstream relevance of supervised syntactic parsing. What is more, there is more and more evidence that PLMs implicitly acquire rich syntactic knowledge through large-scale pretraining (Hewitt and Manning, 2019; Chi et al., 2020) and that exposing them to explicit syntax from human-coded treebanks does not offer significant language understanding benefits (Kuncoro et al., 2020; Glavaš and Vulić, 2021). In order to implicitly acquire syntactic competencies, however, PLMs need language-specific corpora at the scale at which it can only be obtained for a tiny portion of world's 7,000+ languages. For the remaining vast majority of languages – with limited-size monolingual corpora – explicit syntax still provides valuable linguistic bias for more sample-efficient learning in downstream NLP tasks.

Reliable syntactic parsing requires annotated treebanks of reasonable size: this prerequisite is, unfortunately, satisfied for even fewer languages. Despite the multi-year, well-coordinated annotation efforts such as the Universal Dependencies (Nivre et al., 2016, 2020) project, language-specific treebanks are unlikely to appear anytime soon for most world languages. This renders the transfer of syntactic knowledge from high-resource languages with annotated treebanks a necessity. A *truly zero-shot* transfer for *low-resource languages* assumes a set of training treebanks from resource-rich source languages and a target language without any syntactic annotations. Effectively, the task is then to identify the subset of source treebanks, the parser trained on which would yield the best parsing performance for the target language. An exhaustive search over all possible subsets of source treebanks is not only computationally intractable[1] but also

[1]One can create $2^N - 1$ different training sets from a

Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021, pages 4878–4888
August 1–6, 2021. ©2021 Association for Computational Linguistics

uninformative in true zero-shot scenarios in which there is no development treebank (i.e., any syntactically annotated data) for the target language. Most existing transfer methods therefore either (1) choose one (or a few) best source languages for each target language (Rosa and Zabokrtsky, 2015; Agić, 2017; Lin et al., 2019; Litschko et al., 2020) or (2) train a single multilingual parser on all available treebanks; such parsers, based on pretrained multilingual encoders, currently produce best results in low-resource parsing (Kondratyuk and Straka, 2019; Üstün et al., 2020). Other transfer approaches, e.g., based on data augmentation (Şahin and Steedman, 2018; Vania et al., 2019), violate the zero-shot transfer by assuming a small target-language treebank – a requirement unfulfilled for most world languages.[2]

In this work, we propose a simple and effective heuristic for selecting a good set of source treebanks for any given low-resource target language. In our approach, named TOWER, we first hierarchically cluster all Universal Dependencies (UD) languages. To this end, we compute syntactic similarity of languages by comparing manually coded vectors of their syntactic properties from the URIEL database (Littell et al., 2017). We then iteratively 'climb' that language hierarchy level by level, starting from the leaf node of the target language. We stop 'climbing' (i.e., select the set of source treebanks subsumed by the current hierarchy level), when the relative decrease in linguistic similarity of the training sample w.r.t the target language outweighs the increase in size of the training sample. We additionally exploit the linguistic similarity between the target language and its closest sources with existing development treebanks to inform a model selection (that is, early-stopping) heuristic. TOWER substantially outperforms state-of-the-art multilingual parsers – UDPipe (Straka, 2018), UDify (Kondratyuk and Straka, 2019), and UDapter (Üstün et al., 2020) on low-resource languages, while offering comparable performance for high-resource languages.

2 Climbing the TOWER of Treebanks

Constructing the TOWER. We start by hierarchically clustering the set of 89 languages from Universal Dependencies [3] based on their syntactic

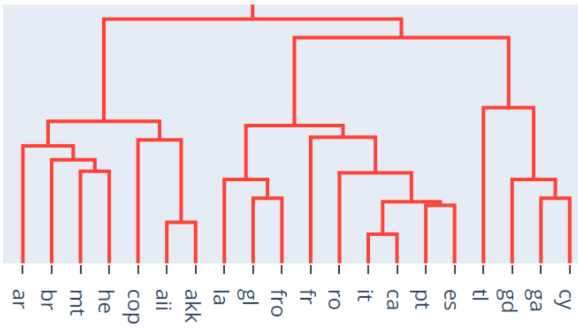

Figure 1: Part of the syntax-based hierarchical clustering of UD languages (ISO 639-1 codes).

similarity. To this end, we represent each language with its `syntax_knn` vector from the URIEL database (Littell et al., 2017). Features of these 103-dimensional vectors correspond to individual syntactic properties from manually coded linguistic resources such as WALS (Dryer and Haspelmath, 2013) and SSWL (Collins and Kayne, 2009). URIEL's `syntax_knn` strategy replaces feature values missing in those resources with kNN-based predictions (cf. (Littell et al., 2017) for more details). We then carry out hierarchical agglomerative clustering with Ward's linkage (Anderberg, 2014) with Euclidean distances between URIEL vectors guiding the clustering. Figure 1 shows a dendrogram of one part of the resulting hierarchy. We display the complete hierarchy in the Appendix. The syntax-based clustering largely reflects memberships in language (sub)families, with a few notable exceptions: e.g., Tagalog (*tl*), from the Austronesian family appears to be syntactically similar to (and is joined with) Scottish (*gd*), Irish (*ga*), and Welsh (cy) from the Celtic branch of the Indo-European family.

Treebank Selection (TBS). For a given test treebank, we start climbing the hierarchy from the leaf node of the treebank's language. Let s_l denote the number of climbing steps we take from the target leaf node l. If the target test treebank also has the corresponding training portion, in-treebank training constitutes the first training configuration (we denote this configuration with $s_l = -1$). For resource-rich languages with several training treebanks, we create the next training sample by concatenating all of those treebanks (we denote this level with $s_l = 0$).[4] For low-resource target lan-

collection of N source treebanks.

[2]For the vast majority of world languages there does not exist a single manually annotated syntactic tree.

[3]We worked with the UD version 2.5.

[4]For example, for the Russian test treebank *SynTagRus*, the training set at $s_l = -1$ consists of the train portion of the same *SynTagRus* treebank; at $s_l = 0$, we concatenate

guages without any training treebanks, the first training sample is collected at $s_l = 1$, where the language is joined with other languages. The training set corresponding to a hierarachy level (i.e., each *join* in the tree) concatenates all training treebanks of all languages (i.e., leaf nodes) of the respective hierarchy subtree.[5]

Let $\{S_n\}_{n=0\text{ (or }-1)}^{N}$ be the set of training configurations collected by climbing the hierarchy starting from the target language l and let $S_n = \cup\{T_k\}_{k=1}^{K}$ be the n-th training set consisting of K training treebanks. As we climb the hierarchy (i.e., as n increases), the training set S_n is bound to grow; at the same time, the sample of training languages becomes increasingly dissimilar w.r.t. the target language l. In other words, as we climb higher up the induced syntactic hierarchy of languages, we train on more data but from a mixture of (syntactically) more distant languages. Let l_k be the language of the training treebank T_k. We then quantify the syntactic similarity $sim(S_n, l)$ between the training set S_n and the target language l as follows:

$$sim(S_n, l) = \frac{1}{|S_n|}\sum_{k=1}^{K}|T_k| \cdot \cos(\mathbf{l}_k, \mathbf{l}) \quad (1)$$

with $\cos(\mathbf{l_k}, \mathbf{l})$ as cosine similarity between URIEL vectors of l_k and l, and relative sizes of individual treebanks $|T_k|/|S_n|$ as weights. We then use the following simple heuristic to select the best training set S_n: we stop climbing when the relative growth of the training set becomes smaller than the relative decrease of the similarity with the target language, i.e., we select the smallest n for which the following condition is satisifed:

$$\frac{|S_{n+1}|}{|S_n|} < \frac{sim(S_n, l)}{sim(S_{n+1}, l)}. \quad (2)$$

Model Selection (MS). Early stopping based on the model performance on a development set (*dev*) is an important mechanism for preventing model overfitting in supervised machine learning. In a truly zero-shot transfer setup, on the one hand, we do not have any development data in the target language. Model selection based on the development set of the source language, on the other hand, overfits the model to the source language, which may hurt effectiveness of the cross-lingual transfer (Keung et al., 2020; Chen and Ritter, 2020). For test treebanks with a respective development portion, TOWER uses that development set for model selection. For low-resource languages l without development treebanks, we compile a *proxy* development set $D_l = \cup\{D_k\}_{k=1}^{K}$ by collecting all development treebanks D_k from the hierarchy level closest to l that encompasses at least one treebank with a development set.[6] Intuitively, the more syntactically similar D_l is to l, the more beneficial the model selection based on D_l will be for performance on l, the optimal model checkpoint w.r.t. l should be closer to the model checkpoint exhibiting best performance on D_l. Accordingly, with M as the model checkpoint with best performance on D_l, we select the model chekpoint $M' = \lfloor sim(D_l, l) \cdot M \rfloor$ (see Eq.(1)) as the "optimal" checkpoint for the target language l.

Shallow Biaffine Parser. TOWER employs the shallow biaffine parser of Glavaš and Vulić (2021), stacked on top of the pretrained XLM-R (Conneau et al., 2020). Compared to the standard biaffine parser (Dozat and Manning, 2017; Kondratyuk and Straka, 2019; Üstün et al., 2020), this shallow variant forwards word-level representations (aggregated from subword output) directly into biaffine products, bypassing deep feed-forward transformations that produce dependent- and head-specific vectors (Dozat and Manning, 2017). The shallow variant is reported to perform comparably (Glavaš and Vulić, 2021), while being faster to train.

3 Evaluation and Discussion

Treebanks and Baselines. We evaluate TOWER on 138 (test) treebanks from Universal Dependencies (Nivre et al., 2020).[7] We compare TOWER against two state-of-the-art multilingual parsers: (1) UDify (Kondratyuk and Straka, 2019) couples the multilingual BERT (mBERT) (Devlin et al.,

the training portions of Russian *GSD*, *PUD*, and *SynTagRus* treebanks.

[5]Note that the number of climbs s_l needed to reach some hierarchy level depends on the language l: e.g., the hierarchy level joining Tagalog (*tl*) with Scottish, Irish, and Welsh ({*gd*, *ga*, *cy*}) is reached in $s_l = 1$ climbs from Tagalog, $s_l = 2$ climbs from Scottish and $s_l = 3$ climbs from Irish and Welsh.

[6]E.g., D_l for *l*=*tl* consists of development portions of *ga* and *gd* treebanks, whereas D_l for *l*=*cy* consists only of the development set of *ga*.

[7]We work with UD v2.5. Due to mismatches between XLM-R's subword tokenizer and word-level treebank tokens we skip: all Chinese treebanks, Assyrian (AS), Old Russian (RNC and TOROT), Skolt Sami (Giellagas), Japanese (Modern and BCCWJ), A. Greek (Perseus), Gothic (PROIEL), Coptic (Scriptorium), OC Slavonic (PROIEL) and Yoruba (YTB).

	∩UDify		∩UDapt		HIGH		LOW	
Model	UAS	LAS	UAS	LAS	UAS	LAS	UAS	LAS
UDify	80.9	73.9	–	–	89.2	85.3	39.9	22.2
UDapter	–	–	63.8	52.8	**90.9**	**87.6**	43.9	29.3
TOWER	**82.4**	**74.3**	**68.9**	**56.0**	90.0	86.3	**53.7**	**33.8**
-TBS	80.8	73.2	62.8	51.7	89.4	85.6	47.0	30.1
-MS	82.1	74.1	67.9	55.2	89.4	85.6	51.2	32.2
-TBS-MS	80.7	83.1	62.4	51.3	89.4	85.6	45.9	29.0

Table 1: Parsing performance (UAS, LAS) on different UD treebank subsets for state-of-the-art multilingual parsers UDify and UDapter and variants of our TOWER method. **Bold:** best performance in each column.

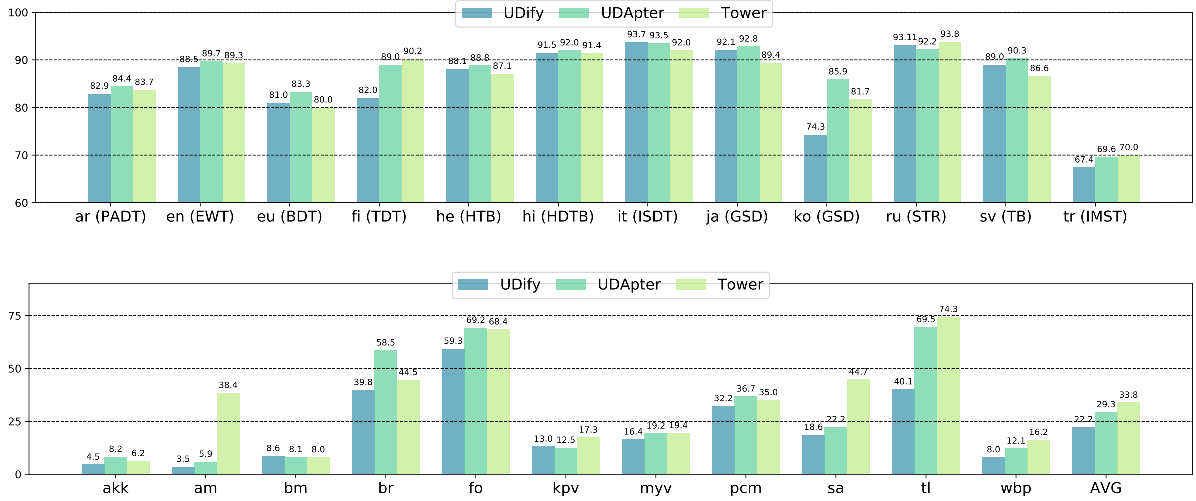

Figure 2: LAS performance of UDify, UDapter and TOWER on 12 high-resource treebanks (top figure), and 11 low-resource languages (bottom figure).

2019) with the deep biaffine parser (Dozat and Manning, 2017) and trains on all UD treebanks; (2) UDapter (Üstün et al., 2020) extends mBERT with adapter parameters (Houlsby et al., 2019; Pfeiffer et al., 2020) that are contextually generated (Platanios et al., 2018) from URIEL vectors – the parameters of the adapter generator are trained on treebanks of 13 diverse resource-rich languages selected by Kulmizev et al. (2019). We additionally quantify the contributions of TOWER's heuristic components (TBS and MS, see §2) by evaluating variants in which we (1) remove TBS and train on the closest language with training data (-TBS), (2) remove MS and just select the model checkpoint that performs best on the proxy dev set D_l (-MS), and (3) remove both TBS and MS (-TBS-MS).

Training and Optimization Details. We limit input sequences to 128 subword tokens. We use XLM-R *Base* with $L = 12$ layers and hidden size $H = 768$ and apply a dropout ($p = 0.1$) on its outputs before forwarding them to the shallow parsing head. We train in batches of 32 sentences and optimize parameters with Adam (Kingma and Ba, 2015) (starting learning rate 10^{-5}). We train for 30 epochs, with early stopping based on dev loss.[8]

Results and Discussion. We show detailed results for all 138 treebanks in the Appendix. In Table 1, we show averages over different treebank subsets: treebanks on which both TOWER and (1) UDify (∩UDify; 111 treebanks) and (2) UDapter (∩UDapt; 39 treebanks) have been evaluated, (3) 12 high-resource languages on which UDapter was trained (HIGH) and (4) 11 low-resource treebanks (LOW) for which all three models have been evaluated. We show LAS scores for languages from

[8]For low-resource languages without the dev set, we use the proxy D_l (see 2). We checkpoint the model (i.e., measure the dev loss) 10 times per epoch and stop training when the loss does not decrease over 10 consecutive checkpoints.

HIGH and LOW in Figure 2. Similar trends are observed with UAS scores.

TOWER outperforms UDify and UDapter in all setups except HIGH, with especially pronounced gains for LOW. This renders TOWER particularly successful for the intended use case: low-resource languages without any training data. Admittedly, the fact that TOWER is built on XLM-R, whereas UDify and UDapter use mBERT, impedes the direct "apples-to-apples" comparison. Two sets of results, however, strongly suggest that it is TOWER's heuristics (TBS & MS) that drive its performance rather than the XLM-R (instead of mBERT) encoder. First, UDapter outperforms TOWER on high-resource languages with large training treebanks (i.e., the HIGH setup). For these languages, however, TOWER effectively does not employ its heuristics: (i) TBS selects the large language-specific treebank(s), as adding any other language prohibitively reduces the perfect similarity $sim(S_0, l) = 1$ (see Eq. (1)); (ii) MS is not used because each high-resource treebank has its own dedicated dev set. Secondly, removing TOWER's heuristics (see -TBS-MS in Table 1) brings its performance slightly below that of UDapter, rendering TBS (primarily) and MS (rather than the XLM-R encoder) crucial for TOWER's gains. Comparing -TBS and -MS reveals that, somewhat expectedly, selecting the "optimal" training sample (TBS) contributes to the overall performance more than the heuristic early stopping (MS).

Looking at individual low-resource languages (Fig. 2), we observe largest gains for Amharic (*am*) and Sanskrit (*sa*). While Sanskrit benefits from TOWER selecting training languages from the same family (Marathi, Urdu, and Hindi), Amharic (Afro-Asiatic family), interestingly, benefits from treebanks of syntactically similar languages from another family (cf. the full TOWER hierarchy in the Appendix) – Tamil and Telugu (Dravidian family). Similarly, Tagalog (Austronesian language) parsing massively benefits from training on Scottish and Irish treebanks (Indo-European, Celtic).

4 Conclusion

We proposed TOWER, a simple yet effective approach to the crucial problem of source language selection for multilingual and cross-lingual dependency parsing. It leverages the language hierarchy, induced from syntax-based manually coded URIEL language vectors, and simple treebank selection heuristics to inform the source selection. A wide-scale UD evaluation and comparisons to current state-of-the-art multilingual dependency parsers validated the effectiveness of TOWER, especially in low-resource languages. Moreover, while the main experiments in this work were based on one particular state-of-the-art parsing architecture, TOWER is fully independent of the chosen underlying parsing model, and thus widely applicable.

Acknowledgments

Goran Glavaš is supported by the Baden Württemberg Stiftung (Eliteprogramm, AGREE grant). The work of Ivan Vulić is supported by the ERC Consolidator Grant LEXICAL: Lexical Acquisition Across Languages (no. 648909).

References

Željko Agić. 2017. Cross-lingual parser selection for low-resource languages. In *Proceedings of the NoDaLiDa 2017 Workshop on Universal Dependencies*, pages 1–10.

Michael R. Anderberg. 2014. *Cluster Analysis for Publications*. Academic Press.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. In *Proceedings of NeurIPS 2020*.

Yang Chen and Alan Ritter. 2020. Model selection for cross-lingual transfer using a learned scoring function. *arXiv preprint arXiv:2010.06127*.

Ethan A. Chi, John Hewitt, and Christopher D. Manning. 2020. Finding universal grammatical relations in multilingual BERT. In *Proceedings of ACL 2020*, pages 5564–5577.

Chris Collins and Richard Kayne. 2009. Syntactic structures of the world's languages.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of ACL 2020*, pages 8440–8451.

Marie-Catherine De Marneffe, Bill MacCartney, Christopher D Manning, et al. 2006. Generating typed dependency parses from phrase structure parses. In *Proceedings of LREC 2006*, pages 449–454.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL-HLT 2019*, pages 4171–4186.

Timothy Dozat and Christopher D. Manning. 2017. Deep biaffine attention for neural dependency parsing. In *Proceedings of ICLR 2017*.

M. S. Dryer and M. Haspelmath. 2013. *WALS Online*. Max Planck Institute for Evolutionary Anthropology, Leipzig.

Goran Glavaš and Ivan Vulić. 2021. Is supervised syntactic parsing beneficial for language understanding? An empirical investigation. In *Proceedings of EACL 2021*, pages 3090–3104.

John Hewitt and Christopher D. Manning. 2019. A structural probe for finding syntax in word representations. In *Proceedings of NAACL-HLT 2019*, pages 4129–4138.

Julia Hockenmaier and Mark Steedman. 2007. CCGbank: A corpus of CCG derivations and dependency structures extracted from the Penn Treebank. *Computational Linguistics*, 33(3):355–396.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for NLP. In *Proceedings of ICML 2019*, pages 2790–2799.

Phillip Keung, Yichao Lu, Julian Salazar, and Vikas Bhardwaj. 2020. Don't use English dev: On the zero-shot cross-lingual evaluation of contextual embeddings. In *Proceedings of EMNLP 2020*, pages 549–554.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *Proceedings of ICLR 2015*.

Dan Kondratyuk and Milan Straka. 2019. 75 languages, 1 model: Parsing Universal Dependencies universally. In *Proceedings of EMNLP-IJCNLP 2019*, pages 2779–2795.

Artur Kulmizev, Miryam de Lhoneux, Johannes Gontrum, Elena Fano, and Joakim Nivre. 2019. Deep contextualized word embeddings in transition-based and graph-based dependency parsing - a tale of two parsers revisited. In *Proceedings of EMNLP-IJCNLP 2019*, pages 2755–2768.

Adhiguna Kuncoro, Lingpeng Kong, Daniel Fried, Dani Yogatama, Laura Rimell, Chris Dyer, and Phil Blunsom. 2020. Syntactic structure distillation pretraining for bidirectional encoders. *Transactions of the ACL*, 8:776–794.

Yu-Hsiang Lin, Chian-Yu Chen, Jean Lee, Zirui Li, Yuyan Zhang, Mengzhou Xia, Shruti Rijhwani, Junxian He, Zhisong Zhang, Xuezhe Ma, Antonios Anastasopoulos, Patrick Littell, and Graham Neubig. 2019. Choosing transfer languages for cross-lingual learning. In *Proceedings of ACL 2019*, pages 3125–3135.

Robert Litschko, Ivan Vulić, Željko Agić, and Goran Glavaš. 2020. Towards instance-level parser selection for cross-lingual transfer of dependency parsers. In *Proceedings of COLING 2020*, pages 3886–3898.

Patrick Littell, David R Mortensen, Ke Lin, Katherine Kairis, Carlisle Turner, and Lori Levin. 2017. URIEL and lang2vec: Representing languages as typological, geographical, and phylogenetic vectors. In *Proceedings of EACL 2017*, pages 8–14.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A robustly optimized BERT pretraining approach. *arXiv:1907.11692*.

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 LREC 2016*, pages 1659–1666.

Joakim Nivre, Marie-Catherine de Marneffe, Filip Ginter, Jan Hajič, Christopher D Manning, Sampo Pyysalo, Sebastian Schuster, Francis Tyers, and Daniel Zeman. 2020. Universal Dependencies v2: An evergrowing multilingual treebank collection. *arXiv preprint arXiv:2004.10643*.

Jonas Pfeiffer, Andreas Rücklé, Clifton Poth, Aishwarya Kamath, Ivan Vulić, Sebastian Ruder, Kyunghyun Cho, and Iryna Gurevych. 2020. AdapterHub: A framework for adapting Transformers. In *Proceedings of EMNLP 2020: System Demonstrations*, pages 46–54.

Emmanouil Antonios Platanios, Mrinmaya Sachan, Graham Neubig, and Tom Mitchell. 2018. Contextual parameter generation for universal neural machine translation. In *Proceedings of EMNLP 2018*, pages 425–435.

Rudolf Rosa and Zdenek Zabokrtsky. 2015. Klcpos3-a language similarity measure for delexicalized parser transfer. In *Proceedings of ACL-JCNLP 2015*, pages 243–249.

Gözde Gül Şahin and Mark Steedman. 2018. Data augmentation via dependency tree morphing for low-resource languages. In *Proceedings of EMNLP 2018*, pages 5004–5009.

Milan Straka. 2018. UDPipe 2.0 prototype at CoNLL 2018 UD shared task. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 197–207.

Ann Taylor, Mitchell Marcus, and Beatrice Santorini. 2003. The Penn treebank: An overview. In *Treebanks*, pages 5–22.

Ahmet Üstün, Arianna Bisazza, Gosse Bouma, and Gertjan van Noord. 2020. UDapter: Language adaptation for truly universal dependency parsing. In *Proceedings of EMNLP 2020*.

Clara Vania, Yova Kementchedjhieva, Anders Søgaard, and Adam Lopez. 2019. A systematic comparison of methods for low-resource dependency parsing on genuinely low-resource languages. In *Proceedings of EMNLP 2019*, pages 1105–1116.

Appendix

Treebank	Method	UAS	LAS
Afrikaans AfriBooms (af)	UDPipe	89.38	86.58
	UDify	86.97	83.48
	Tower	88.26	85.28
Akkadian PISANDUB (akk)	UDify	27.65	4.54
	UDapter	26.4	8.2
	Tower	33.45	6.12
Amharic ATT (am)	UDify	17.38	3.49
	UDapter	12.8	5.91
	Tower	72.64	38.41
Ancient Greek PROIEL (grc)	UDPipe	85.93	82.11
	UDify	78.91	72.66
	Tower	85.04	79.85
Arabic NYUAD (ar)	Tower	33.53	15.94
Arabic PADT (ar)	UDPipe	87.54	82.94
	UDify	87.72	82.88
	UDapter	88.66	84.42
	Tower	88.92	83.72
Arabic PUD (ar)	UDify	76.17	67.07
	Tower	75.94	59.72
Armenian ArmTDP (hy)	UDPipe	78.62	71.27
	UDify	85.63	78.61
	Tower	86.57	80.51
Bambara CRB (bm)	UDify	30.28	8.6
	UDapter	28.7	8.1
	Tower	31.33	8.03
Basque BDT (eu)	UDPipe	86.11	82.86
	UDify	84.94	80.97
	UDapter	87.25	83.33
	Tower	84.28	80.02
Belarusian HSE (be)	UDPipe	78.58	72.72
	UDify	91.82	87.19
	UDapter	84.16	79.33
	Tower	86.40	81.56
Bhojpuri BHTB (bho)	UDapter	52.9	37.34
	Tower	52.62	35.86
Breton KEB (br)	UDify	63.52	39.84
	UDapter	72.91	58.5
	Tower	67.73	44.47
Bulgarian BTB (bg)	UDPipe	93.38	90.35
	UDify	95.54	92.4
	Tower	95.67	92.03
Buryat BDT (bxr)	UDPipe	32.6	18.83
	UDify	48.43	26.28
	UDapter	48.68	28.89
	Tower	51.53	29.16
Catalan AnCora (ca)	UDPipe	93.22	91.06
	UDify	94.25	92.33
	Tower	94.04	92.08
Croatian SET (hr)	UDPipe	91.1	86.78
	UDify	94.08	89.79
	Tower	92.22	87.02
Czech CAC (cs)	UDPipe	92.99	90.71
	UDify	94.33	92.41
	Tower	94.91	92.10
Czech CLTT (cs)	UDPipe	86.9	84.03
	UDify	91.69	89.96
	Tower	94.11	91.38
Czech FicTree (cs)	UDPipe	92.91	89.75
	UDify	95.19	92.77
	Tower	95.12	91.83
Czech PDT (cs)	UDPipe	93.33	91.31
	UDify	94.73	92.88
	Tower	95.01	92.41
Czech PUD (cs)	UDify	92.59	87.95
	Tower	93.26	87.06

Treebank	Method	UAS	LAS
Danish DDT (da)	UDPipe	86.88	84.31
	UDify	87.76	84.5
	Tower	85.60	82.14
Dutch Alpino (nl)	UDPipe	91.37	88.38
	UDify	94.23	91.21
	Tower	93.42	90.31
Dutch LassySmall (nl)	UDPipe	90.2	86.39
	UDify	94.34	91.22
	Tower	92.45	88.29
English ESL (en)	Tower	30.22	6.45
English EWT (en)	UDPipe	89.63	86.97
	UDify	90.96	88.5
	UDapter	93.12	89.67
	Tower	92.16	89.29
English GUM (en)	UDPipe	87.27	84.12
	UDify	89.14	85.73
	Tower	90.07	86.61
English LinES (en)	UDPipe	84.15	79.71
	UDify	87.33	83.71
	Tower	87.12	82.91
English PUD (en)	UDify	91.52	88.66
	Tower	90.89	87.33
English ParTUT (en)	UDPipe	90.29	87.27
	UDify	92.84	90.14
	Tower	89.36	85.63
English Pronouns (en)	Tower	89.50	85.37
Erzya JR (myv)	UDify	31.9	16.38
	UDapter	34.21	19.15
	Tower	36.44	19.38
Estonian EDT (et)	UDPipe	88.0	85.18
	UDify	89.53	86.67
	Tower	90.24	87.08
Estonian EWT (et)	Tower	88.80	84.54
Faroese OFT (fo)	UDify	67.24	59.26
	UDapter	77.15	69.2
	Tower	77.43	68.41
Finnish FTB (fi)	UDPipe	90.68	87.89
	UDify	86.37	81.4
	Tower	91.91	89.05
Finnish PUD (fi)	UDify	89.76	86.58
	Tower	88.24	82.48
Finnish TDT (fi)	UDPipe	89.88	87.46
	UDify	86.42	82.03
	UDapter	91.87	89.01
	Tower	92.78	90.22
French FQB (fr)	Tower	93.36	87.00
French FTB (fr)	Tower	28.04	14.80
French GSD (fr)	UDPipe	90.65	88.06
	UDify	93.6	91.45
	Tower	94.06	91.31
French PUD (fr)	UDify	88.36	82.76
	Tower	91.02	83.52
French ParTUT (fr)	UDPipe	92.17	89.63
	UDify	90.55	88.06
	Tower	87.90	79.33
French Sequoia (fr)	UDPipe	92.37	90.73
	UDify	92.53	90.05
	Tower	92.07	89.93
French Spoken (fr)	UDPipe	82.9	77.53
	UDify	85.24	80.01
	Tower	84.41	74.77
Galician CTG (gl)	UDPipe	86.44	83.82
	UDify	84.75	80.89
	Tower	83.85	80.65

Treebank	Method	UAS	LAS
Galician TreeGal (gl)	UDPipe UDify TOWER	82.72 84.08 77.57	77.69 76.77 66.87
German GSD (de)	UDPipe UDify TOWER	85.53 87.81 89.11	81.07 83.59 84.19
German HDT (de)	TOWER	97.65	96.54
German LIT (de)	TOWER	86.55	78.74
German PUD (de)	UDify TOWER	89.86 89.15	84.46 81.02
Greek GDT (el)	UDPipe UDify TOWER	92.1 94.33 94.13	89.79 92.15 91.16
Hebrew HTB (he)	UDPipe UDify UDapter TOWER	89.7 91.63 91.86 90.71	86.86 88.11 88.75 87.05
Hindi HDTB (hi)	UDPipe UDify UDapter TOWER	94.85 95.13 95.29 95.12	91.83 91.46 91.96 91.42
Hindi PUD (hi)	UDify TOWER	71.64 73.02	58.42 50.68
Hungarian Szeged (hu)	UDPipe UDify TOWER	84.04 89.68 87.87	79.73 84.88 81.02
Indonesian GSD (id)	UDPipe UDify TOWER	85.31 86.45 83.71	78.99 80.1 76.84
Indonesian PUD (id)	UDify TOWER	77.47 76.71	56.9 53.16
Irish IDT (ga)	UDPipe UDify TOWER	80.39 80.05 80.33	72.34 69.28 66.80
Italian ISDT (it)	UDPipe UDify UDapter TOWER	93.49 95.54 95.32 94.47	91.54 93.69 93.46 91.98
Italian PUD (it)	UDify TOWER	94.18 94.13	91.76 89.01
Italian ParTUT (it)	UDPipe UDify TOWER	92.64 95.96 95.06	90.47 93.68 91.57
Italian PoSTWITA (it)	TOWER	86.95	81.75
Italian TWITTIRO (it)	TOWER	86.93	80.91
Italian VIT (it)	TOWER	91.80	87.05
Japanese GSD (ja)	UDPipe UDify UDapter TOWER	95.06 94.37 94.87 92.58	93.73 92.08 92.84 89.44
Japanese PUD (ja)	UDify TOWER	94.89 91.12	93.62 88.41
Karelian KKPP (krl)	UDapter TOWER	61.86 62.18	48.35 45.60
Kazakh KTB (kk)	UDPipe UDify UDapter TOWER	53.3 74.77 74.13 73.70	33.38 63.66 60.74 59.88
Komi Permyak UH (koi)	UDapter TOWER	36.89 42.36	23.05 25.81
Komi Zyrian IKDP (kpv)	UDify TOWER	36.01 40.87	22.12 24.71
Komi Zyrian Lattice (kpv)	UDify UDapter TOWER	28.85 28.4 33.29	12.99 12.5 17.33

Treebank	Method	UAS	LAS
Korean GSD (ko)	UDPipe UDify UDapter TOWER	87.7 82.74 89.39 86.04	84.24 74.26 85.91 81.70
Korean Kaist (ko)	UDPipe UDify TOWER	88.42 87.57 88.78	86.48 84.52 86.11
Korean PUD (ko)	UDify TOWER	63.57 61.78	46.89 38.40
Kurmanji MG (kmr)	UDPipe UDify UDapter TOWER	45.23 35.86 26.37 72.00	34.32 20.4 12.1 51.02
Latin ITTB (la)	UDPipe UDify TOWER	91.06 92.43 91.25	88.8 90.12 87.67
Latin PROIEL (la)	UDPipe UDify TOWER	83.34 84.85 83.74	78.66 80.52 77.75
Latin Perseus (la)	UDPipe UDify TOWER	71.2 78.33 73.53	61.28 69.6 62.16
Latvian LVTB (lv)	UDPipe UDify TOWER	87.2 89.33 92.26	83.35 85.09 88.52
Lithuanian ALKSNIS (lt)	TOWER	87.35	81.58
Lithuanian HSE (lt)	UDPipe UDify TOWER	51.98 79.06 79.25	42.17 69.34 65.47
Livvi KKPP (olo)	UDapter TOWER	57.86 62.77	43.34 44.62
Maltese MUDT (mt)	UDPipe UDify TOWER	84.65 83.07 76.64	79.71 75.56 67.31
Marathi UFAL (mr)	UDPipe UDify UDapter TOWER	70.63 79.37 61.01 70.39	61.41 67.72 44.4 57.77
Mbya Guarani Dooley (gun)	TOWER	18.10	5.82
Mbya Guarani Thomas (gun)	TOWER	32.36	11.23
Moksha JR (mdf)	UDapter TOWER	40.15 44.21	26.55 27.45
Naija NSC (pcm)	UDify UDapter TOWER	45.75 49.24 52.03	32.16 36.72 34.95
North Sami Giella (sme)	UDPipe UDify TOWER	78.3 74.3 53.53	73.49 67.13 42.05
Norwegian Bokmaal (no)	UDPipe UDify TOWER	92.39 93.97 94.77	90.49 92.18 93.12
Norwegian Nynorsk (no)	UDPipe UDify TOWER	92.09 94.34 93.96	90.01 92.37 91.65
Norwegian NynorskLIA (no)	UDPipe UDify TOWER	68.08 75.4 75.43	60.07 69.6 69.82
Old French SRCMF (fro)	UDPipe UDify TOWER	91.74 91.74 89.75	86.83 86.65 83.48
Persian Seraji (fa)	UDPipe UDify TOWER	90.05 89.59 91.29	86.66 85.84 87.43

Treebank	Method	UAS	LAS
Polish LFG (pl)	UDPipe	96.58	94.76
	UDify	96.67	94.58
	Tower	97.06	95.18
Polish PDB (pl)	Tower	94.99	89.95
Polish PUD (pl)	Tower	94.13	87.44
Portuguese Bosque (pt)	UDPipe	91.36	89.04
	UDify	91.37	87.84
	Tower	91.50	88.29
Portuguese GSD (pt)	UDPipe	93.01	91.63
	UDify	94.22	92.54
	Tower	93.80	91.98
Portuguese PUD (pt)	UDify	87.02	80.17
	Tower	87.27	77.86
Romanian Nonstandard (ro)	UDPipe	89.12	84.2
	UDify	90.36	85.26
	Tower	90.59	84.41
Romanian RRT (ro)	UDPipe	91.31	86.74
	UDify	93.16	88.56
	Tower	93.61	87.70
Romanian SiMoNERo (ro)	Tower	91.19	86.75
Russian GSD (ru)	UDPipe	88.15	84.37
	UDify	90.71	86.03
	Tower	91.85	88.28
Russian PUD (ru)	UDify	93.51	87.14
	Tower	94.59	88.26
Russian SynTagRus (ru)	UDPipe	93.8	92.32
	UDify	94.83	93.13
	UDapter	94.04	92.24
	Tower	95.28	93.75
Russian Taiga (ru)	UDPipe	75.45	69.11
	UDify	84.02	77.8
	Tower	84.83	77.71
Sanskrit UFAL (sa)	UDify	40.21	18.56
	UDapter	44.32	22.22
	Tower	63.05	44.66
Scottish Gaelic ARCOSG (gd)	Tower	81.32	73.82
Serbian SET (sr)	UDPipe	92.7	89.27
	UDify	95.68	91.95
	Tower	94.36	90.93
Slovak SNK (sk)	UDPipe	89.82	86.9
	UDify	95.92	93.87
	Tower	93.77	90.87
Slovenian SSJ (sl)	UDPipe	92.96	91.16
	UDify	94.74	93.07
	Tower	94.91	93.50
Slovenian SST (sl)	UDPipe	73.51	67.51
	UDify	80.37	75.03
	Tower	78.64	73.10
Spanish AnCora (es)	UDPipe	92.34	90.26
	UDify	92.99	90.5
	Tower	92.67	90.44
Spanish GSD (es)	UDPipe	90.71	88.03
	UDify	90.82	87.23
	Tower	92.12	89.64
Spanish PUD (es)	UDify	90.45	83.08
	Tower	89.66	80.23
Swedish LinES (sv)	UDPipe	86.07	81.86
	UDify	88.77	85.49
	Tower	88.63	85.07
Swedish PUD (sv)	UDify	89.17	86.1
	Tower	89.20	84.95
Swedish Talbanken (sv)	UDPipe	89.63	86.61
	UDify	91.91	89.03
	UDapter	92.62	90.26
	Tower	89.70	86.60
Swedish Sign Language SSLC (swl)	UDPipe	50.35	37.94
	UDify	40.43	26.95
	Tower	31.56	20.57
Swiss Ger. UZH (gsw)	UDapter	59.74	45.49
	Tower	55.61	40.17
Tagalog TRG (tl)	UDify	64.04	40.07
	UDapter	84.78	69.52
	Tower	91.78	74.32
Tamil TTB (ta)	UDPipe	74.11	66.37
	UDify	79.34	71.29
	UDapter	70.28	46.05
	Tower	71.28	64.36
Telugu MTG (te)	UDPipe	91.26	85.02
	UDify	92.23	83.91
	UDapter	83.52	71.1
	Tower	90.43	81.97
Thai PUD (th)	UDify	49.05	26.06
	Tower	78.23	53.80
Turkish GB (tr)	Tower	75.36	59.39
Turkish IMST (tr)	UDPipe	74.19	67.56
	UDify	74.56	67.44
	UDapter	76.97	69.63
	Tower	77.90	70.00
Turkish PUD (tr)	UDify	67.68	46.07
	Tower	62.29	41.57
Ukrainian IU (uk)	UDPipe	88.29	85.25
	UDify	92.83	90.3
	Tower	92.54	89.89
Upper Sorbian UFAL (hsb)	UDPipe	45.58	34.54
	UDify	71.55	62.82
	UDapter	62.28	54.2
	Tower	70.98	60.90
Urdu UDTB (ur)	UDPipe	87.5	81.62
	UDify	88.43	82.84
	Tower	87.43	81.62
Uyghur UDT (ug)	UDPipe	78.46	67.09
	UDify	65.89	48.8
	Tower	79.11	66.41
Vietnamese VTB (vi)	UDPipe	70.38	62.56
	UDify	74.11	66.0
	Tower	72.40	63.50
Warlpiri UFAL (wbp)	UDify	21.66	7.96
	UDapter	24.2	12.1
	Tower	31.85	16.24
Welsh CCG (cy)	UDapter	70.75	54.43
	Tower	77.22	57.56
Wolof WTB (wo)	Tower	69.06	58.13

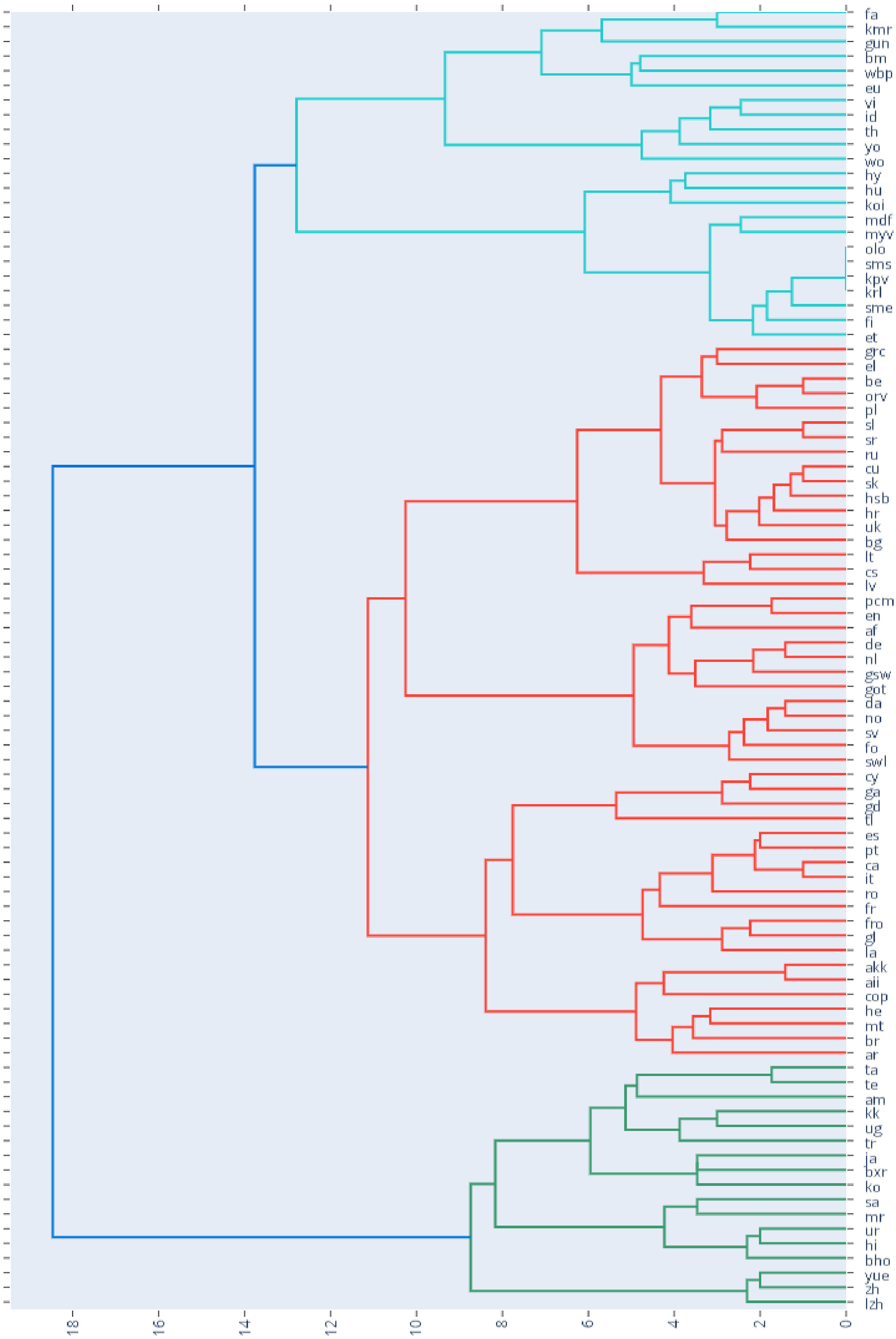

Figure 3: Dendrogram of the full syntax-based hierarchical clustering of 89 languages from UD v2.5. Languages are denoted with their ISO 639-1 codes.