Hybrid Enhanced Universal Dependencies Parsing

Johannes Heinecke
Orange / Data AI / AITT / Deskiñ
2 avenue Marzin
22307 Lannion cedex, France
johannes.heinecke@orange.com

Abstract

This paper describes our system to predict enhanced dependencies for Universal Dependencies (UD) treebanks, which ranked 2nd in the Shared Task on Enhanced Dependency Parsing with an average ELAS of 82.60%. Our system uses a hybrid two-step approach. First, we use a graph-based parser to extract a basic syntactic dependency tree. Then, we use a set of linguistic rules which generate the enhanced dependencies for the syntactic tree. The application of these rules is optimized using a classifier which predicts their suitability in the given context. A key advantage of this approach is its language independence, as rules rely solely on dependency trees and UPOS tags which are shared across all languages.

1 Introduction

Parsing Enhanced Universal Dependencies (EUD) (Schuster and Manning, 2016) is an interesting extension of dependency parsing. EUDs provide syntactic information which can be crucial for any NLP processing based on syntactic analysis.

The shared task on EUD parsing (Bouma et al., 2020) provided the platform to develop and compare various systems. Our team participated using a hybrid system (machine learning/rule-based) which came second in both metrics, ELAS (82,6%) and EULAS (84,6%).

1.1 Related Work

Whereas basic dependencies are strict surface syntax trees, enhanced dependencies are implicite syntactic links in constructions like coordinations, raise/control constructions or relative clauses. EUDs also enrich existing basic dependencies such as `obl` and `nmod` relations by adding information about the adposition used and morphological cases. Finally EUDs propose the syntactic annotation of elided words, absent in the actual sentence (Nivre et al., 2018). Even though the basic dependencies tree (apart from the `orphan` relation) is part of the EUD graph, the latter is no longer a tree, since individual tokens can have more than one head. Most of the EUDs can be predicted deterministically (Nivre et al., 2018), others, notably the prediction of EUDs for elided words, are more complex (Schuster et al., 2018).

2 System description

The data of Universal Dependencies treebanks (Nivre et al., 2016)[1] used for the shared task and annotated with enhanced dependencies (other than copied basic dependencies) is small. In total, all training treebanks contain about 5.1 million words, only 5.6% of those have a second enhanced dependency attached to them (the first being the copied basic dependency). Another 7.2% and 8.3% of words have an enhanced dependency like `obl:...` or `nmod:...` which correspond to the basic dependency but also give the adposition and morphological case (if existing in the language in question). In total, only 21.1% or 1 million words have any non basic enhanced dependency.

The enhanced dependencies address specific and well known linguistic phenomena, and are relatively deterministic (Nivre et al., 2018), once the basic dependency tree is available. For this reason, we decided to utilise a hybrid system, using a graph-based parser to produce first a dependency tree and a rule system which uses the generated dependency tree to determin the enhanced dependencies. The latter uses a (learned) filter to control the application of rules in certain contexts. The system functions as a pipeline (cf. Figure 1). Thus errors in earlier parts of the pipeline will impact the results of the following components.

[1] https//universaldependencies.org/

Proceedings of the 16th International Conference on Parsing Technologies and the IWPT 2020 Shared Task, pages 174–180

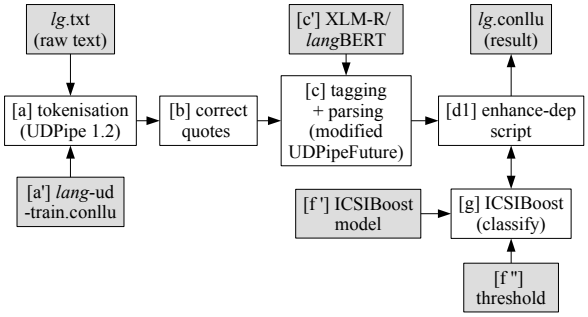

Figure 1: System architecture: gray boxes indicate data, white boxes indicate processing

2.1 Sentence segmentation, tokenisation

For sentence segmentation and tokenisation of the incoming raw text, we use UDPipe 1.2 (Straka and Straková, 2017) ([a] in Figure 1[2]). We trained a tokenizer per language using the training files ([a'] in Fig. 1) of each treebank. For languages with more than one treebank, we chose the one with the largest training file. A python postprocessing script ([b] in Fig. 1) deals with obvious tokenization errors, such as quotes concatenated to letters (e.g. like `word"` , `word″` or `"word`) and separates these tokens into two.

2.2 Tagging and parsing

To tag and parse the texts, we use a special version of UdpipeFuture ([c] in Fig. 1) (Straka, 2018), winner in terms of the Morphology-aware Labeled Attachment Score (MLAS)[3] metric of the shared task on Dependency Parsing in 2018. In our version, we also use contextual embeddings, however instead of using ELMo (Peters et al., 2018), we experimented with a range of contextual embeddings, either multilingual as BERT (Devlin et al., 2019) or XLM-R (Conneau et al., 2019), or language specific models like RoBERTa-large (English, Liu et al. (2019)), CamemBERT (French, Martin et al. (2019)). Experiments with these embeddings on the CoNLL 2018 Shared Task (Zeman et al., 2018) data show that XLM-R outperforms the best score for nearly all treebanks of the 2018 Shared Task. The Content-Word Labeled Attachment Score (CLAS)[4] scores for these experiments on the treebanks which are used for the Enhanced Dependencies Shared Task are given in Table 1.

treebank	*BERT*	*XLM-R*	*CoNLL 2018 best score*
ar-padt	80.35	**82.43**	74.00
bg-btb	88.49	**90.23**	88.40
cs-cac	89.79	**92.08**	90.06
cs-fictree	87.89	**90.22**	89.61
cs-pdt	90.09	**92.18**	90.53
cs-pud	79.97	79.97	**83.57**
en-ewt	86.89	**87.24**	81.64
en-pud	**87.60**	87.60	85.68
et-edt	83.46	**86.07**	83.74
fi-pud	88.15	**90.00**	88.72
fi-tdt	86.65	**89.90**	87.42
fr-sequoia	90.03	**90.47**	87.26
it-isdt	89.11	**90.01**	88.32
lv-lvtb	79.75	**83.18**	81.17
nl-alpino	85.57	**88.20**	85.23
nl-lassysmall	84.07	**85.18**	81.71
pl-lfg	93.02	**94.28**	93.18
ru-syntagrus	91.60	**93.30**	91.00
sk-snk	87.42	**89.35**	87.01
sv-pud	81.19	**82.25**	79.01
sv-talbanken	86.68	**88.72**	86.94
uk-iu	83.83	**86.97**	85.99

Table 1: Results (CLAS) on CoNLL 2018 Shared Tasks treebanks also present in the IWPT 2020 Shared Task (best values in bold)

Although the CoNLL 2018 Shared Task is based on UD v2.2, we were able to produce similar promising results with the data provided by the current shared task, based on UD v2.5.

To prepare for the shared task, we first merged treebanks of the same language when more than one was available: this was the case for Czech, Dutch, Estonian and Polish. Then, we trained and tested the tagging and parsing using UDPipeFuture with all contextual word embedding models available for the given language (unless the treebank did not provide a dev-file, as e.g. the PUD treebanks). In addition to the multilingual contextual embeddings BERT and XLM-R, we also tested some language-specific transformers such as Arabic BERT[5], CamemBERT[6] (French, Martin et al.

[2]Letters in brackets refer to the architecture diagrams shown in Figures 1 and 4. Identical letters refer to the same component.

[3]MLAS is metric inspired by the Content-Word Labeled Attachment Score (CLAS) (Zeman et al., 2018) which takes into account POS tags and morphological features.

[4]CLAS is a variant or the classical Labeled Attachment Score (Nivre and Fang, 2017). It only takes into account dependency reations between content words, in order to be able to compare parsing results of typologically different languages

[5]https://github.com/alisafaya/Arabic-BERT

[6]https://camembert-model.fr/

(2019)), Finnish BERT[7] (Virtanen et al., 2019), Italian BERT[8], Swedish BERT[9], Slavic BERT[10] (Czech, Bulgarian, Polish, Russian; Arkhipov et al. (2019)) and BERTje[11] (Dutch, Vries et al. (2019)). The results are shown in Table 2.

lang.	*XLMR*	*BERT (ml)*	*language specific embeddings*	
ar	**84.44**	82.86	84.86	Arabic BERT
bg	91.30	90.52	**91.45**	Slavic BERT
cs	**93.95**	92.35	93.29	Slavic BERT
en	**90.56**	89.03	90.11	RoBERTa
et	**89.13**	86.68	(no emb. available)	
fi	90.69	87.96	**90.98**	Finnish BERT
fr	92.70	92.91	**93.43**	CamemBERT
it	93.04	92.46	**93.25**	Italian BERT
lt	**84.92**	81.82	(no emb. available)	
lv	**89.03**	86.14	(no emb. available)	
nl	90.79	90.63	**91.78**	BERTje
pl	**93.16**	91.63	92.47	Slavic BERT
ru	**93.65**	92.04	92.73	Slavic BERT
sk	**90.85**	89.94	88.79	Slavic BERT[12]
sv	87.78	86.46	**89.29**	Swedish BERT
ta	**72.84**	69.44	(no emb. available)	
uk	**90.78**	88.86	85.84	Slavic BERT

Table 2: Parsing test results (LAS) using different contextual word embeddings, best results in bold (train/dev corpora of the shared task)

Evaluations on the development corpora showed that XLM-R gave the best results for nearly all languages, with some exceptions: for Arabic (Arabic BERT), Bulgarian (Slavic BERT), Finnish (Finnish BERT), French (CamemBERT), Italian (Italian BERT) and Dutch (Dutch BERT) the language-specific versions of BERT gave better results in terms of Labeled Attachment Score (LAS) for the parsing.

2.3 Determining enhanced dependencies

To extract enhanced dependencies we implemented a script ([d1] in Figure 1) which interprets the basic dependency tree and applies some linguistic rules. We built the rules by analysing manually the different types of EUDs in all the provided treebanks.

To obtain a language-independent system which can predict enhanced dependencies on any language, we need homogeneous annotations in all the treebanks. Since these annotations, which require time-consuming manual work, are currently missing in many UD treebanks, and the existing annotations are not always homogeneous, we opted for a rule-based system. For example, `dep` is used frequently as an additional[13] enhanced dependency in the Czech and Arabic treebanks. Other differences stem from language differences, e.g. in Finnish-TDT and Polish-PDB case information is sometimes given with the `nmod:poss` enhanced dependency, which is absent in other treebanks for languages without morphological case. Similarly, the `conj` enhanced dependency is enriched with the lemma of the `cc` relation only in the treebanks of Dutch, English, Italian and Swedish. Similar differences can be observed for relative pronouns or case information for oblique nominals (`obl:<prep>:<case>`) or nominal modifiers (`nmod:<prep>:<case>`). The French-Sequoia treebank frequently employs `nmod:enh`, `amod:enh`, `nsubj:enh` and `nsubj:passxoxobjenh` which are not defined in the guidelines.

Our script takes into account these language specific differences. For example, it discards prepositions and case information in `nmod`/`obl` enhanced dependencies for languages where this information has not been annotated. In general, the script mainly exploits basic dependencies and UPOS, i.e. universal information, to determine the enhanced dependencies.

The script first initialises enhanced dependencies by copying all basic dependencies (except `orphan`). In a second step we look for all words with a `obl` and `nmod` relation and check whether they have a `case`-dependant. If so, we enrich the enhanced dependencies with the lemma of this dependant. If present, we add the `Case`-feature to `obl:<ADP>` and `nmod:<ADP>` as well.

For coordinations of nouns, we simply take the heads of words with a `conj`-relation (cf. relation (A) in Figure 2) and determine the dependency relation of its head (relation (B) in Fig. 2). With this

[7] https://huggingface.co/TurkuNLP/bert-base-finnish-cased-v1

[8] https://huggingface.co/dbmdz/bert-base-italian-cased

[9] https://huggingface.co/KB/bert-base-swedish-cased

[10] https://huggingface.co/DeepPavlov/bert-base-bg-cs-pl-ru-cased

[11] https://huggingface.co/wietsedv/bert-base-dutch-cased

[12] Slavic BERT works nearly as well as XLM-R for Polish and Russian. However, for Ukrainian and Slovak, which are not part of Slavic BERT), the comparatively lower result is not surprising.

[13] I.e. an enhanced dependency which is not a basic dependency.

information we can add the enhanced dependency relation of the coordinated noun to its enhanced head (relation (C) in Fig. 2). We also enrich the `conj`-relation (relation (D) in Fig. 2) using the lemma of the `cc`-dependant (relation (E) in Fig. 2).

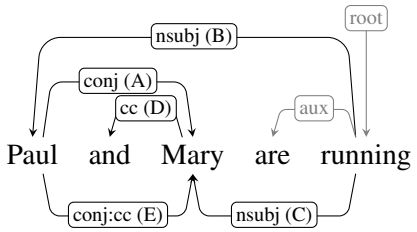

Figure 2: Rule to predict EUDs for coordination: A + B → C and A + D → E

`xcomp` (relation (F) in Fig. 3) relations are processed in a similar way by going through the dependency tree to find the subject (rel. (G) in Fig. 3) of the head of the `xcomp`. We then can add the enhanced dependency relation `nsubj` (rel. (H) in Fig. 3). Referents for relative clauses are processed in an equivalent manner too.

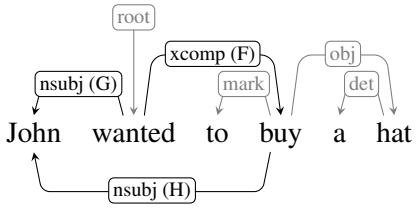

Figure 3: Rule to predict EUDs for `xcomp` subjects: F + G → H

In order to insert elided nodes, we interpret the `orphan` relation. Whereas the insertion itself works fine, we were not able to predict correctly the needed enhanced dependencies for elided nodes and have abandoned this prediction for the shared task.

We validated the rules for enhanced dependency extraction on gold basic dependencies from the validation corpora to avoid the accumulation of errors from the tagging and parsing step. This yielded encouraging results presented in Table 3.

To further improve the performance of the rule-based approach, and to take into account the errors in the tagging/parsing step, we add the ICSIBoost[14] classifier (Favre et al., 2008).

This classifier (cf. [g] in Figures 1 and 4) estimates the probability of success of a given rule in a given context. For this task, we trained a single classifier using the following features:

- rule name
- treebank language
- enhanced dependency label
- UPOS of enhanced dependency head
- (basic) dependency relation of the enhanced dependency head,
- distance (in words) of the basic dependency head
- distance of the enhanced dependency head

To generate the training corpus of ICSIBoost, we ran our enhance-script ([d2] in Fig. 4) on the training CoNLL-U files ([a'] in Fig. 4), with gold UPOS and basic dependencies) of each language to obtain the list of appropriate features and the information whether the rule produced a correct EUD or not within the given context ([d'] in Fig. 4). We then trained ICSIBoosts on this list to obtain a classifier model ([f'] in Fig. 4) which we integrated into the enhance-script ([d3] in Fig. 4) to obtain more accurate predictions.

To get the best threshold for each language, we ran our script ([d3]) on the development CoNLL-U files with various thresholds for each language with UPOS and basic dependencies predicted by UdpipeFuture using contextual embeddings. Rules

Language	*ELAS*	*EULAS*
ar	95.18	96.84
bg	97.84	98.34
cs	94.67	95.79
en	98.04	98.96
et	92.61	95.90
fi	94.40	97.07
fr	96.42	98.21
it	98.41	99.32
lt	94.55	96.61
lv	91.03	95.74
nl	94.40	98.42
pl	91.13	97.23
ru	95.42	96.94
sk	95.44	96.42
sv	96.07	98.38
ta	96.95	99.54
uk	94.58	96.64
Average	**95.13**	**97.43**

Table 3: Enhanced dependencies on gold basic dependencies (development files; without ICSIBoost)

[14] https://github.com/benob/icsiboost

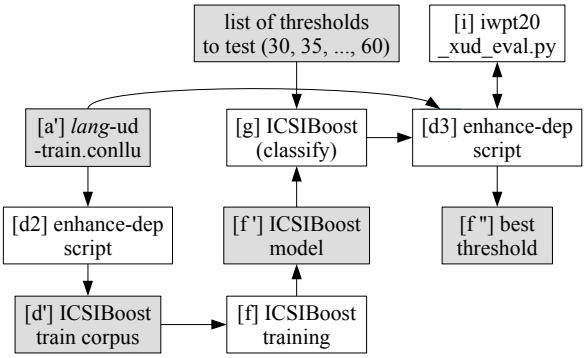

Figure 4: ICSIBoost training

whose score fell below the threshold, in a given context, were not applied. It turned out that thresholds between 30% and 60% gave the best result in terms of ELAS (cf. Table 4). However, the gain from this classification is small for most languages. We observed the biggest increase in ELAS for Estonian, where a threshold of 40% results in 2.13 percent points more than without any filter, whereas for Bulgarian or Italian the difference is only 0.12 or 0.14 points, respectively. On average, the gain is about 0.7 points.

Arabic	40%	Latvian	50%
Bulgarian	55%	Dutch	40%
Czech	40%	Polish	40%
English	55%	Russian	35%
Estonian	40%	Slovak	30%
Finnish	47%	Swedish	40%
French	50%	Tamil	55%
Italian	60%	Ukrainian	50%
Lithuanian	60%		

Table 4: ICSIBoost thresholds to apply a rule in a given context

Running our entire pipeline on gold UPOS and basic dependencies shows that we can predict enhanced dependencies with a very high precision (cf. Table 5).

Applying the entire pipeline on the raw text files provided for the evaluation produced the results shown in Table 6. Since the script which generates the enhanced dependencies depends on basic dependencies and indirectly on the UPOS tags, a lower LAS yields a lower ELAS. By definition, EULAS is always slightly above ELAS. We do not exploit XPOS, since they are too language-specific. Thus the bad results for Finnish XPOS tags do not have an impact on the E(U)LAS score (Table 6). Interestingly the poor sentence segmentation results for Arabic, English and Dutch did not impact the final results since tagging (UPOS) and parsing (LAS) nevertheless gave good results.

Language	*ELAS*	*EULAS*
ar	95.33	97.06
bg	97.74	98.27
cs	94.74	95.88
en	98.19	99.02
et	92.40	96.08
fi	94.94	97.02
fr	97.69	99.10
it	98.18	99.34
lt	93.88	96.10
lv	90.65	95.62
nl	96.36	98.36
pl	88.85	97.08
ru	95.58	97.12
sk	96.00	96.80
sv	95.93	97.86
ta	98.11	99.58
uk	94.82	96.91
Average	**95.26**	**97.48**

Table 5: Predicting enhanced dependencies on gold basic dependencies

3 Conclusion and perspectives

Considering that training data was heterogeneous, partially incomplete, and in general not very voluminuous, our hybrid machine-learning (ML)/rule-based approach gave very good results for the shared task. A possible extension would be the processing of elided nodes.

Even if for the long term a purely ML-based approach may prove more efficient, at least our language-independent system can help to pre-annotate existing UD treebanks which, after human validation, can be the basis of an ML approach on predicting enhanced dependencies.

Acknowledgments

We are grateful to two reviewers for constructive comments on the first version of the paper.

References

Mikhail Arkhipov, Maria Trofimova, Yuri Kuratov, and Alexey Sorokin. 2019. Tuning Multilingual Transformers for Language-Specific Named Entity Recognition. In *ACL 2019*, pages 89–93, Florence.

Language	*Tokens*	*Words*	*Sent.*	*UPOS*	*XPOS*	*Lemmas*	*UAS*	*LAS*	*CLAS*	*MLAS*	*EULAS*	*ELAS*
ar	99.97	94.57	81.67	91.87	89.54	90.58	79.39	75.54	72.73	67.43	72.91	70.96
bg	99.89	99.89	93.63	99.17	97.40	98.30	94.46	91.83	89.09	86.11	90.40	89.42
cs	99.85	99.85	92.55	98.98	96.95	98.83	93.93	92.25	91.18	87.16	88.38	86.95
en	99.22	99.22	81.19	96.21	95.47	96.88	89.21	86.94	84.37	78.78	86.02	85.21
et	99.70	99.70	88.02	97.45	98.10	95.18	88.69	86.20	84.96	80.31	84.70	81.03
fi	99.69	99.68	88.52	98.11	56.30	92.12	91.97	90.31	89.07	85.13	87.79	86.24
fr	99.54	99.13	93.27	96.04	99.13	96.53	90.98	86.47	81.11	70.69	85.81	83.63
it	99.90	99.82	96.29	98.46	98.34	98.50	94.74	93.09	89.61	86.84	91.99	90.83
lv	99.34	99.34	98.36	96.69	90.25	96.28	90.69	87.96	85.85	78.45	84.51	82.11
lt	99.94	99.94	87.46	96.57	91.05	94.29	84.91	81.54	79.84	69.95	77.61	75.89
nl	99.71	99.71	77.23	96.75	95.52	97.19	90.57	88.05	83.25	78.73	86.58	85.14
pl	99.33	99.79	96.73	98.58	94.36	98.01	94.26	92.11	90.58	80.97	89.15	80.39
ru	99.48	99.48	97.93	98.80	99.48	98.26	94.41	93.31	92.24	89.93	90.97	89.84
sk	99.99	99.99	84.06	97.16	88.14	96.63	91.20	89.06	87.37	78.03	86.17	84.36
se	99.72	99.72	88.02	97.51	95.64	93.39	88.65	85.91	84.28	69.67	84.36	83.27
ta	99.49	94.96	93.83	87.41	80.36	90.09	71.85	66.23	63.71	54.37	65.68	64.23
uk	99.79	99.75	95.31	98.03	94.69	97.47	90.72	88.59	85.98	79.26	85.46	84.64
Average	**99.68**	**99.09**	**90.24**	**96.69**	**91.81**	**95.80**	**89.45**	**86.79**	**84.42**	**77.75**	**84.62**	**82.60**

Table 6: Official results of our system

Gosse Bouma, Djamé Seddah, and Daniel Zeman. 2020. Overview of the IWPT 2020 Shared Task on Parsing into Enhanced Universal Dependencies. In *Proceedings of the 16th International Conference on Parsing Technologies and the IWPT 2020 Shared Task on Parsing into Enhanced Universal Dependencies*, Seattle, US. Association for Computational Linguistics.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grace, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Unsupervised Cross-lingual Representation Learning at Scale. https://arxiv.org/abs/1911.02116.

Jacob Devlin, Ming-Wei Chang Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language under-standing. In *NAACL*, page 4171–4186, Minneapolis. Association for Computational Linguistics.

Benoît Favre, Dilek Hakkani-Tür, Slav Petrov, and Klein Dan. 2008. Efficient sentence segmentation using syntactic features. In *Spoken Language Technology Workshop*, pages 77–80. IEEE.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach. https://arxiv.org/abs/1907.11692.

Louis Martin, Benjamin Muller, Pedro Javier Ortiz Suárez, Yoann Dupont, Laurent Romary, Éric Villemonte de la Clergerie, Djamé Seddah, and Benoît Sagot. 2019. CamemBERT: a Tasty French Language Model. https://arxiv.org/abs/1911.03894.

Joakim Nivre and Chiao-Ting Fang. 2017. Universal Dependency Evaluation. In *Proceedings of the NoDaLiDa 2017 Workshop on Universal Dependencies*, pages 86–95, Göteborg.

Joakim Nivre, Marie-Catherine de Marneffe, Filip Ginter, Yoav Goldberg, Yoav Goldberg, Jan Hajič, Manning Christopher D., Ryan McDonald, Slav Petrov, Sampo Pyysalo, Natalia Silveira, Reut Tsarfaty, and Daniel Zeman. 2016. Universal Dependencies v1: A Multilingual Treebank Collection. In *the tenth international conference on Language Resources and Evaluation*, pages 23–38, Portorož, Slovenia. European Language Resources Association.

Joakim Nivre, Paola Marongiu, Filip Ginter, Jenna Kanerva, Simonetta Montemagni, Sebastian Schuster, and Maria Simi. 2018. Enhancing Universal Dependency Treebanks: A Case Study. In *Proceedings of the Second Workshop on Universal Dependencies (UDW 2018)*, pages 102–107, Brussels, Belgium. Association for Computational Linguistics.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kendon Lee, and Luke Zettlemoyer. 2018. Deep Contextualized Word Representations. In *NAACL*, page 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

Sebastian Schuster and Christopher D. Manning. 2016. Enhanced English Universal Dependencies: An Improved Representation for Natural Language Understanding Tasks. In *the tenth international conference on Language Resources and Evaluation*, pages 2371–2378, Portorož, Slovenia. European Language Resources Association.

Sebastian Schuster, Joachim Nivre, and Christopher D. Manning. 2018. Sentences with Gapping: Parsing and Reconstructing Elided Predicates. In *NAACL*, pages 1156–1168, New Orleans, Louisiana. Association for Computational Linguistics.

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,

Brussels. Association for Computational Linguistics.

Milan Straka and Jana Straková. 2017. Tokenizing, POS Tagging, Lemmatizing and Parsing UD 2.0 with UDPipe. In *ACL 2017*, pages 88–99, Vancouver.

Antti Virtanen, Jenna Kanerva, Rami Ilo, Juhani Luotolahti, Tapio Salakoski, Filip Ginter, and Sampo Pyysalo. 2019. Multilingual is not enough: BERT for Finnish. https://arxiv.org/abs/1912.07076.

Wietse de Vries, Andreas van Cranenburgh, Arianna Bisazza, Tommaso Caselli, Gertjan van Noord, and Malvina Nissim. 2019. BERTje: A Dutch BERT Model. https://arxiv.org/abs/1912.09582.

Daniel Zeman, Jan Hajič, Martin Popel, Martin Potthast, Milan Straka, Filip Ginter, Joakim Nivre, and Slav Petrov. 2018. CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 1–21, Brussels. Association for Computational Linguistics.