Prague at EPE 2017: The UDPipe System

Milan Straka and **Jana Straková** and **Jan Hajič**
Charles University
Faculty of Mathematics and Physics
Institute of Formal and Applied Linguistics
{straka,strakova,hajic}@ufal.mff.cuni.cz

Abstract

We present our contribution to The First Shared Task on Extrinsic Parser Evaluation (EPE 2017). Our participant system, the UDPipe, is an open-source pipeline performing tokenization, morphological analysis, part-of-speech tagging, lemmatization and dependency parsing. It is trained in a language agnostic manner for 50 languages of the UD version 2. With a relatively limited amount of training data (200k tokens of English UD) and without any English specific tuning, the system achieves overall score 56.05, placing as the 7^{th} participant system.

1 Introduction

Language syntax has been a topic of interest and research for hundreds of years. Syntactical analysis, most commonly in form of constituency or dependency trees, has therefore been one of the long-standing goals of computational linguistics.

Syntactic analysis was considered crucial to understand the semantics of a message. Lately, statistical and especially neutral network models have achieved superb results in natural language processing without explicit syntax recognition, by considering sentences to be merely a sequence of words. However, quite recently, syntactic trees have been shown to improve performance compared to the sequential models, especially in tasks requiring deeper understanding of text, like text summarization (Kong et al., 2017) or textual entailment (Hashimoto et al., 2016).

Consequently, syntactic parsing has its merit both as a standalone application and as a preprocessing step for further language processing, resulting in two kinds of evaluation methods – either intrinsic or extrinsic. While the intrinsic evaluation is straightforward and commonly used, extrinsic evaluation is much more complex, and to our best knowledge, there had been no standardized set of tasks serving as extrinsic evaluation.

Recently, performance of raw text parsing has been evaluated in the *CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies* (Zeman et al., 2017),[1] providing a rich intrinsic evaluation of 33 systems across 81 treebanks in 49 languages of the latest version of UD, the Universal Dependencies project (Nivre et al., 2016), which seeks to develop cross-linguistically consistent treebank annotation of morphology and syntax.

The First Shared Task on Extrinsic Parser Evaluation (EPE 2017) (Oepen et al., 2017)[2] proposes an extrinsic parser evaluation metric, by means of three downstream applications that are known to depend heavily on syntactic analysis. All tasks, the *biological event extraction* (Björne et al., 2017), *negation scope resolution* (Lapponi et al., 2017) and *fine-grained opinion analysis* (Johansson, 2017), require pre-processing of raw English texts into the EPE interchange format, which is a general format encoding arbitrary dependency graphs. Each such graph represents a sentence and consists of several nodes, which correspond to substrings of the original document and include POS tags, lemmas and arbitrary morphological features. The aforementioned tasks then process these graphs and compute individual evaluation metrics, whose unweighted combination is the final EPE score.

This paper describes performance of the UDPipe system in the EPE 2017 shared task. UDPipe (Straka and Straková, 2017)[3] is an open-source

[1] http://ufal.mff.cuni.cz/conll-2017-shared-task
[2] http://epe.nlpl.eu
[3] http://ufal.mff.cuni.cz/udpipe

tool which automatically performs sentence segmentation, tokenization, POS tagging, lemmatization and dependency trees, using UD version 2 treebanks as training data. This system has been used both as a baseline system and also a participant system in CoNLL 2017 shared task, ranking 8^{th} in the official (intrinsic) evaluation (Straka and Straková, 2017).

Even if the EPE 2017 shared task is only in English, the submitted UDPipe system is trained in a strictly language-agnostic manner without any specific handling of English, using UD 2.0 training data only. It is therefore interesting to compare it to other English-tailored participating systems.

In Section 2, we briefly discuss related work. The UDPipe system including chosen hyperparameters for English is described in Section 3. The extrinsic evaluation results of UDPipe are presented in Section 4, together with intrinsic metrics and discussion of observed performance. Finally, we conclude in Section 5.

2 Related Work

Deep neural networks have achieved remarkable results in many areas of machine learning. In NLP, end-to-end approaches were initially explored by Collobert et al. (2011). With a practical method for precomputing word embeddings (Mikolov et al., 2013) and utilization of recurrent neural networks (Hochreiter and Schmidhuber, 1997; Graves and Schmidhuber, 2005) and sequence-to-sequence architecture (Sutskever et al., 2014; Cho et al., 2014), deep neural networks achieved state-of-the-art results in many NLP areas like POS tagging (Ling et al., 2015), named entity recognition (Yang et al., 2016) or machine translation (Vaswani et al., 2017).

The wave of neural network parsers was started recently by Chen and Manning (2014), who presented a fast and accurate transition-based parser. The proposed neural network architecture is simple, consisting of only an input layer, one hidden layer and an output softmax layer, without any recurrent connections. The UDPipe parser (Straka and Straková, 2017) is based on this architecture, and adds partially non-projective and fully non-projective transition systems, as well as a search-based oracle (Straka et al., 2015).

Many other parser models followed, employing various techniques like stack LSTM (Dyer et al., 2015), global normalization (Andor et al., 2016), contextualization of embeddings using bidirectional LSTM (Kiperwasser and Goldberg, 2016), biaffine attention (Dozat and Manning, 2017) or recurrent neural network grammars (Kuncoro et al., 2016), improving LAS score in English and Chinese dependency parsing by more than 2 points in 2016.

2.1 Motivation For Structured Sentence Processing

Although the sequence-to-sequence architecture provides overwhelming performance compared to traditional methods, there have been several attempts to enhance it utilizing syntactic information. Many such designs employ dependency trees not merely as features, but either to encode an input sentence according to syntactic tree (Tai et al., 2015; Li et al., 2017; Chen et al., 2017) or generate an output sentence as a dependency tree (Wu et al., 2017; Rabinovich et al., 2017).

A comprehensive comparison of processing input sentence either as a sequence or as a tree was performed by Yogatama et al. (2016). Additionally, the authors also considered the eventuality of utilizing task-specific syntax trees, both in semi-supervised manner (i.e., bootstrapping the task-specific syntax with manually annotated trees and allowing their change later) and in unsupervised manner. While the supervised syntax did not demonstrate much improvement, both semi-supervised and unsupervised approach (i.e., learning task-specific syntax) yielded substantial gains in all four examined tasks.

3 UDPipe

UDPipe[4] is an open-source pipeline which performs tokenization, morphological analysis, part-of-speech tagging, lemmatization and dependency parsing. It is a simple-to-use tool consisting of one binary and one model (per language) and can be easily trained using solely data in CoNLL-U format, without additional linguistic knowledge on the users' part. Precompiled binaries for Windows, Linux and OS X are available, as are bindings for Python,[5] Perl,[6] Java and C#. Source code is available on GitHub[7] under MPL license.

The initial UDPipe 1.0 release (Straka et al., 2016) processed CoNLL-U v1 files and was dis-

[4] `http://ufal.mff.cuni.cz/udpipe`
[5] PyPI package `ufal.udpipe`
[6] CPAN package `UFAL::UDPipe`
[7] `http://github.com/ufal/udpipe`

tributed with 36 pretrained models[8] on UD 1.2 data. Updated UDPipe 1.1 and UDPipe 1.2 versions (Straka and Straková, 2017) process CoNLL-U v2 files and were employed in CoNLL 2017 shared task, with UDPipe 1.1 serving as a baseline system and UDPipe 1.2 attending as a participant system. Recently, pretrained models for 50 languages were released based on UD 2.0 data.[9]

All UDPipe models, especially the ones participating in the CoNLL 2017 shared task, have been evaluated using several intrinsic metrics (Zeman et al., 2017). Therefore, we employed these exact models to participate in EPE 2017, in order to facilitate comparison between the extrinsic and intrinsic measurements.

We now briefly describe the most recent version, UDPipe 1.2, together with the chosen hyperparameters for the English model (according to performance on the development set). More detailed language-independent description of the system and its gradual updates are available in Straka and Straková (2017) and in Straka et al. (2016).

3.1 Tokenizer

The tokenizer performing sentence segmentation and tokenization is trained purely with the UD training data. The CoNLL-U format allows for the reconstruction of the original pre-tokenized text using the `SpaceAfter=No` feature, which indicates that a given token was not followed by a space separator in the original text. This facilitates training a model which predicts the probability of a token break after every character in a given plain text.

Sentence breaks can be trained analogously. However, the CoNLL-U v1 format does not provide markup for paragraph and document boundaries. These are often indicated by visual layout and/or spacing, but if not annotated in the data, a sentence segmenter has to predict the sentence boundary at the end of a paragraph only from the raw text. Considering the examples from the English UD data presented in Figure 1, such sentence breaks confuse the segmenter and prompt it to split sentences more often than necessary.

The CoNLL-U v2 format has been updated to include markup for paragraph and document boundaries. Unfortunately, only document boundaries are marked in English UD 2.0 data, resulting in the segmenter still being trained on sentence boundaries marked in Figure 1.

Keep in touch, / Mike / Michael J. McDermott

i have two options / using the metro or the air france bus / can anybody tell me if the metro runs directly ...

Figure 1: Examples of sentence breaks (denoted with slash) in English UD data which are hard to predict without inter-sentence spacing and layout.

Technically, the UDPipe tokenizer predicts for each character whether it is followed by a token break, sentence break or none of above. Each character is represented using randomly initialized embedding of dimension d and a bidirectional GRU (Cho et al., 2014) network is employed during the prediction. The details of the architecture, training and inference are explained in Straka et al. (2016).

For English, embedding and GRU dimension are set to 64. The network is trained using dropout rate of 10% before and after the GRU cells for 100 epochs, each consisting of 200 batches containing 50 segments of 50 characters. The network weights are updated using Adam (Kingma and Ba, 2014) with initial learning rate of 0.002. Additionally, space is assumed to always separate tokens (due to no training token containing a space) and the network is therefore trained to only predict token breaks which do not precede a space character.

3.2 Tagger

Part-of-speech tagging is performed in two steps: firstly, a set of candidate *(UPOS, XPOS, FEATS)* triples are generated for each token using its suffix of length at most 4, and secondly, these candidates are disambiguated on a sentence-level using averaged perceptron (Collins, 2002) with Viterbi decoding of order 3.

To facilitate the candidate generation, a guesser dictionary with a predefined number of most common candidate triples for every possible suffix of length 4 is constructed according to the training data. When searching the guesser dictionary, entires with longer suffix match are always preferred. Additionally, for every token in the training data, all its appearing *(UPOS, XPOS, FEATS)* analyses are kept.

In order to generate candidates for a given token, two cases are considered. If the token was

[8] `http://hdl.handle.net/11234/1-1659`

[9] `http://hdl.handle.net/11234/1-2364`

present in the training data, all its analyses appearing in the training data are returned, together with 6 another (differing) most common candidates from the guesser dictionary. If the tokes was not present in the training data, 10 most common candidates from the guesser dictionary are generated.

The candidates are disambiguated using averaged perceptron utilizing a predefined rich set of feature templates based on classification features developed by Spoustová et al. (2009) for Czech.

A lemmatizer is nearly identical to the above described part-of-speech tagger. For every token, the candidates are *(UPOS, lemma rule)* pairs, where the *lemma rule* is the shortest formula for generating a lemma from a given token, using any combination of "remove a specific prefix", "remove a specific suffix", "append a prefix" and "append a suffix" operations. For the English lemmatizer, at most 4 candidates are generated for every token, because of the smaller number of lemmas (compared to XPOS and morphological features).

Theoretically, both the part-of-speech tagging and lemmatizing could be performed jointly using candidate quadruples, but such approach results in lower performance (we hypothesise that the required number of candidate quadruples is too high for the disambiguation step to be performed effectively).

3.3 Dependency Parser

UDPipe utilizes fast transition-based neural dependency parser inspired by Chen and Manning (2014). The parser is based on a simple neural network with just one hidden layer and without any recurrent connections, using locally-normalized scores.

The parser offers several transition systems, from projective and partially non-projective to fully non-projective. For English, a projective arc-standard system (Nivre, 2008) with both a dynamic oracle (Goldberg et al., 2014) and a search-based oracle (Straka et al., 2015) yield the best performance. It is possible to combine both oracles, because the search-based oracle employs an arbitrary transition-based parser – even the one utilizing a dynamic oracle.

Even if a projective transition system is used, non-projective trees are used during training, with the parser trying to predict a (projective) subset of dependency edges.

The parser employs FORM, UPOS, FEATS and DEPREL embeddings. The form embeddings of dimension 64 are precomputed with `word2vec` on the training data only, with the following options:

```
word2vec -cbow 0 -size 64 -window 10 -negative 5
  -hs 0 -sample 1e-1 -iter 15 -min-count 2
```

The precomputed embeddings are used only for forms occurring at least twice in the training data; forms appearing only once are considered unknown forms and used to train the (initially random) embedding of unknown words. The UPOS, FEATS and DEPREL embeddings have dimension 20 and are initialized randomly. All embeddings are updated during training.

The size of the hidden layer is 200. The network is trained using SGD with minibatches of size 10, starting with learning rate 0.01 and gradually decaying it to the final 0.001. L2 regularization with weight 1.5e-6 is applied to reduce overfitting.

3.4 Training UDPipe

UDPipe is trained without any language specific knowledge. Even if we have so far described specific hyperparameter values used by the English models, the hyperparameters for each treebank are extensively tuned on the development set.

The UD 2.0 data contain three English treebanks. Consequently, in addition to training treebank-specific models, we also experiment with training a model using a union of all these tree treebanks. Even if the treebanks use different XPOS tags and there are annotation inconsistencies among the treebanks (which are observable using the intrinsic evaluation of the merged model on the individual treebanks' test sets), we hypothesise that the larger training data should benefit real-word applications.

3.5 Example Parse Tree

To illustrate the UD-style trees with universal dependency relations and universal POS tags, we provide an example of a (correctly parsed) tree in Figure 2.

4 Experiments and Results

We submitted five different UDPipe configurations to the EPE 2017 shared tasks. These runs are described in Table 1. The run 0 is the English treebank model of UDPipe 1.2 from the CoNLL 2017 shared task (Zeman et al., 2017). The exactly same model is used as a run 1, but using the tok-

Run name	Run	Description	Tokens
UD2.0 En/UDPipe/20	0	UDPipe 1.2, UD 2.0 English data, UDPipe tokenizer, beam size 20	204.5k
UD2.0 En/EPE/20	1	UDPipe 1.2, UD 2.0 English data, EPE provided tokenizer, beam size 20	204.5k
UD2.0 EnMerged/UDPipe/20	2	UDPipe 1.2, UD 2.0 English + English LinES + English ParTUT data, UDPipe tokenizer, beam size 20	292.2k
UD2.0 EnMinus/UDPipe/5	3	UDPipe 1.1, first 95% of UD 2.0 English data, UDPipe tokenizer, beam size 5	192.5k
UD1.2 En/UDPipe/5	4	UDPipe 1.0, UD 1.2 English data, UDPipe tokenizer, beam size 5	204.5k
Stanford-Paris	*6*	*UD v1 enhanced dependencies, WSJ+Brown+GENIA data*	*1692.0k*

Table 1: Description of employed systems

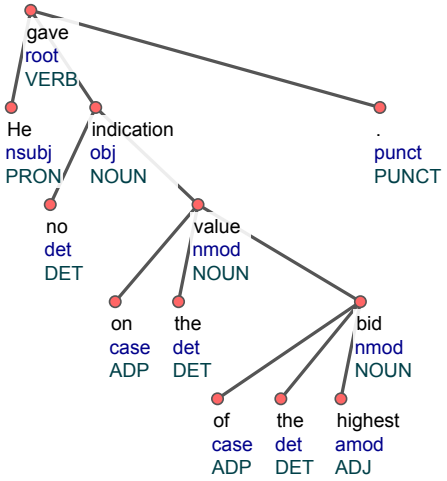

Figure 2: An example parse tree from UD 2.0 English data. For every word, its form, dependency relation and universal POS tag is displayed.

enization and segmentation of the data provided by the EPE 2017 organizers. The consequent run is the CoNLL 2017 shared task model trained on all three English treebanks. The last two runs are the English treebank models of UDPipe 1.1 and UDPipe 1.0. The first four runs are based on UD 2.0, and only the last run utilizes UD 1.2 data. Note that the run 3 uses only a subset of the training data, because the models were released for the CoNLL 2017 shared task before the release of the test data, using official development data for evaluation. For comparison, we additionally include the overall best participant system of the shared task.

The overall results of UDPipe in the EPE 2017 shared task are presented in Table 2. According to the overall score, UDPipe placed 7$^{\text{th}}$ out of 8 participants of the shared task, by a large margin compared to the best participating system. The performance on the three individual tasks are detailed in Tables 3, 4 and 5.

To enable interpretation of the results, we also provide the intrinsic evaluation of the employed models on the UD test sets in Table 6.

Overall Results

With the overall score of 56.05, UDPipe lacks behind nearly all other participant systems. The overall scores of the systems ranking immediately above UDPipe are 56.23, 56.24, 56.65, 56.81 and 58.57, with the best system achieving a respectable score of 60.51. The best overall UDPipe score is achieved by the English-only CoNLL 2017 UDPipe 1.2 model with EPE-provided tokenization.

One of the probable cause of our lower performance is the size of the training data – while the UD 2.0 data offer training data of 200k tokens (290k if all three English treebanks are merged), most other participants use Wall Street Journal corpus (Marcus et al., 1993) with 800k tokens, sometimes also together with Brown corpus (Francis and Kucera, 1979) an GENIA corpus (Ohta et al., 2002), resulting in circa 1700k tokens.

Furthermore, we emphasize that even though the EPE 2017 shared task focused on English language only, UDPipe is trained in a language agnostic manner for 50 languages without any adaptation for English other than setting up the hyperparameters of the artificial neural networks.

Tokenization Issues

The overall results in Table 2 indicate that the UDPipe tokenization is of lower quality – using the EPE-provided tokenizer improves the overall score by 2 points. By contrast, the evaluation on the UD 2.0 data (the Words and Sentences columns of Table 6) show opposite results, with the EPE-provided tokenizer substantially degrading performance on UD 2.0 test sets.

We therefore hypothesise that the lower extrinsic performance of UDPipe tokenization is a consequence of the tokenization and sentence segmentation annotated in the UD data. We argue that to improve the annotation, one possible course of action is to indicate paragraph boundaries in English UD 2.0 data, which might improve the performance of trained sentence segmenter.

UDPipe run	Event extraction	Negation resolution	Opinion analysis	Overall score
0-UD2.0 En/UDPipe/20	43.58	58.83	59.79	54.07
1-UD2.0 En/EPE/20	45.54	61.62	61.00	56.05
2-UD2.0 EnMerged/UDPipe/20	44.25	59.95	58.71	54.30
3-UD2.0 EnMinus/UDPipe/5	42.70	59.95	58.90	53.85
4-UD1.2 En/UDPipe/5	43.22	50.85	58.53	50.86
Stanford-Paris, run 6	*50.23*	*66.16*	*65.14*	*60.51*

Table 2: Overall EPE evaluation results

UDPipe run	Approximate span & recursive mode		
	Precision	Recall	F1-score
0-UD2.0 En/UDPipe/20	53.84	36.61	43.58
1-UD2.0 En/EPE/20	56.35	38.21	45.54
2-UD2.0 EnMerged/UDPipe/20	53.22	37.87	44.25
3-UD2.0 EnMinus/UDPipe/5	51.91	36.27	42.70
4-UD1.2 En/UDPipe/5	51.71	37.12	43.22
Stanford-Paris, run 6	*58.36*	*44.09*	*50.23*

Table 3: Event extraction evaluation results

UDPipe run	Scope match			Scope tokens			Event match			Full negation		
	P	R	F	P	R	F	P	R	F	P	R	F
0-UD2.0 En/UDPipe/20	99.39	65.73	79.13	90.12	86.76	88.41	66.23	61.82	63.95	99.10	41.83	58.83
1-UD2.0 En/EPE/20	98.77	64.26	77.86	88.58	87.75	88.16	70.44	66.67	68.50	99.16	44.70	61.62
2-UD2.0 EnMerged/UDPipe/20	99.40	67.34	80.29	91.45	87.49	89.43	65.81	61.82	63.75	99.12	42.97	59.95
3-UD2.0 EnMinus/UDPipe/5	99.38	64.92	78.54	90.84	85.48	88.08	66.46	63.25	64.82	99.12	42.97	59.95
4-UD1.2 En/UDPipe/5	97.81	54.03	69.61	90.40	83.36	86.74	62.11	59.88	60.97	98.90	34.22	50.85
Stanford-Paris, run 6	99.44	*70.68*	*82.63*	*93.06*	*85.48*	*89.11*	*72.33*	*68.45*	*70.34*	*99.24*	*49.62*	*66.16*

Table 4: Negation resolution evaluation results

UDPipe run	Expressions			Holders			Polarity			Holders (in vitro)		
	P	R	F	P	R	F	P	R	F	P	R	F
0-UD2.0 En/UDPipe/20	64.32	55.07	59.33	49.03	41.71	45.08	54.44	45.36	49.49	62.61	57.21	59.79
1-UD2.0 En/EPE/20	63.57	55.15	59.06	48.81	44.31	46.46	53.68	45.93	49.50	62.31	59.74	61.00
2-UD2.0 EnMerged/UDPipe/20	64.57	54.58	59.15	50.01	39.79	44.32	54.93	45.22	49.60	63.45	54.63	58.71
3-UD2.0 EnMinus/UDPipe/5	64.15	54.43	58.89	48.20	41.25	44.46	54.30	44.82	49.11	61.26	56.72	58.90
4-UD1.2 En/UDPipe/5	63.98	54.46	58.84	48.38	40.99	44.38	53.99	44.50	48.79	61.00	56.25	58.53
Stanford-Paris, run 6	*63.90*	*56.07*	*59.73*	*54.14*	*46.41*	*49.98*	*54.04*	*45.87*	*49.62*	*68.86*	*61.81*	*65.14*

Table 5: Opinion analysis evaluation results

Row	Data	Plain text processing						Using gold tokenization			
		Words	Sents	UPOS	XPOS	UAS	LAS	UPOS	XPOS	UAS	LAS
0-UD2.0 En/UDPipe/20	UD 2.0 En	**99.0**	**75.3**	**93.5**	**92.9**	**80.3**	**77.2**	94.4	93.8	**84.6**	**81.3**
	UD 2.0 EnMerged	**98.9**	**79.5**	91.8	—-	78.4	73.9	92.7	—-	81.4	76.6
	UD 1.2 En	**99.0**	**75.3**	87.9	**92.9**	75.7	63.7	88.8	93.8	79.1	66.8
1-UD2.0 En/EPE/20	UD 2.0 En	96.2	59.9	90.7	90.0	74.6	71.8	94.4	93.8	**84.6**	**81.3**
	UD 2.0 EnMerged	97.8	71.0	90.6	—-	75.8	71.4	92.7	—-	81.4	76.6
	UD 1.2 En	96.2	59.9	85.1	90.0	70.3	58.7	88.8	93.8	79.1	66.8
2-UD2.0 EnMerged/UDPipe/20	UD 2.0 En	**99.0**	**75.3**	93.4	92.6	79.8	76.7	94.4	93.6	84.0	80.6
	UD 2.0 EnMerged	**98.9**	**79.5**	**92.0**	—-	**79.1**	**74.9**	**92.9**	—-	**82.2**	**77.7**
	UD 1.2 En	**99.0**	**75.3**	87.8	92.6	75.6	63.4	88.7	93.6	78.9	66.3
3-UD2.0 EnMinus/UDPipe/5	UD 2.0 En	98.7	73.2	93.1	92.4	78.9	75.8	**94.5**	**93.9**	83.8	80.7
	UD 2.0 EnMerged	98.8	78.6	91.6	—-	77.7	73.1	92.8	—-	81.1	76.3
	UD 1.2 En	98.7	73.2	87.5	92.4	74.6	62.6	88.9	**93.9**	78.6	66.3
4-UD1.2 En/UDPipe/5	UD 2.0 En	98.4	72.3	87.3	92.2	73.9	62.0	88.8	93.8	78.8	66.3
	UD 2.0 EnMerged	98.7	77.8	86.5	—-	73.9	60.1	87.6	—-	77.2	63.0
	UD 1.2 En	98.4	72.3	**92.9**	92.2	**78.3**	**75.1**	**94.5**	93.8	**84.2**	**80.7**

Table 6: Intrinsic evaluation of UDPipe runs on the Universal Dependencies test data (Nivre et al., 2017).

Merged English Treebanks

Although the model trained on the three merged UD 2.0 English treebanks provide inconsistent XPOS tags and shows slight performance drop on the main English UD 2.0 treebank (cf. Table 6), the extrinsic evaluation of this model shows noticeable improvement. The improvement may be attributed both to the increased size and diversity of the training data, but also to the different annotation, which might serve as a regularization.

According to the extrinsic results, the merged model used together with the EPE-provided tokenizer should surpass the overall score of the best submitted UDPipe run.

Negation Resolution Results Drop of Run 4

The Table 2 indicates a surprising drop of performance of the run 4 (UDPipe 1.0 English model trained using UD 1.2 data) on the Negation resolution task, without a corresponding change on the two other tasks.

Note that the three EPE tasks are able to use only one kind of POS tags, i.e., either UPOS or XPOS in case of UDPipe. The decision on the type of POS tags used is performed by the EPE organizers according to the performance on the development set. For the UDPipe systems, XPOS tags are utilized overwhelmingly, with the UPOS tags being used only once – by the run 4 on the Negation resolution task. Therefore, we initially hypothesised that the drop is caused by the fact that the UPOS tags are much more coarse than the XPOS tags.

However, after evaluating the results on both XPOS and UPOS tags, we found out that XPOS tags are in fact used, and the information about used UPOS tags in the official results is incorrect.

After further investigation, we found out that the Negation resolution task texts are the only one containing non-ASCII characters (i.e., Unicode quotation marks, apostrophes and hyphens) and that the UDPipe 1.0 tokenizer does not correctly process them, resulting in poor tokenization and segmentation.

5 Conclusions and Future Work

We described the UDPipe systems used in the EPE 2017 shared task, presented the extrinsic and intrinsic evaluation of the submitted models, discussed the results and offered several hypotheses to interpret the data. For the immediate future work, we will carry out several experiments to support the hypotheses:

- When the paragraph boundaries are annotated in the UD data, does the trained sentence segmenter achieve better performance?
- Can a rule-based English tokenizer also improve the results?
- What effect would larger training data (like WSJ) have?
- What performance would a state-of-the-art dependency parser attain using the UD 2.0 data only?

Acknowledgments

This work has been partially supported and has been using language resources and tools developed, stored and distributed by the LINDAT/CLARIN project of the Ministry of Education, Youth and Sports of the Czech Republic (project LM2015071). This research was also partially supported by OP VVV projects CZ.02.1.01/0.0/0.0/16_013/0001781 and CZ.02.2.69/0.0/0.0/16_018/0002373, by project No. DG16P02R019 supported by the Ministry of Culture of the Czech Republic and by SVV project number 260 453.

References

Daniel Andor, Chris Alberti, David Weiss, Aliaksei Severyn, Alessandro Presta, Kuzman Ganchev, Slav Petrov, and Michael Collins. 2016. Globaly normalized transition-based neural networks. In *Association for Computational Linguistic*. http://arxiv.org/abs/1603.06042.

Jari Björne, Filip Ginter, and Tapio Salakoski. 2017. EPE 2017: The Biomedical event extraction downstream application. In *Proceedings of the 2017 Shared Task on Extrinsic Parser Evaluation at the Fourth International Conference on Dependency Linguistics and the 15th International Conference on Parsing Technologies*. Pisa, Italy, page 13 – 20.

Danqi Chen and Christopher Manning. 2014. A fast and accurate dependency parser using neural networks. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, Doha, Qatar, pages 740–750. http://www.aclweb.org/anthology/D14-1082.

Huadong Chen, Shujian Huang, David Chiang, and Jiajun Chen. 2017. Improved neural machine translation with a syntax-aware encoder and decoder. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume*

1: Long Papers). Association for Computational Linguistics, Vancouver, Canada, pages 1936–1945. http://aclweb.org/anthology/P17-1177.

KyungHyun Cho, Bart van Merrienboer, Dzmitry Bahdanau, and Yoshua Bengio. 2014. On the properties of neural machine translation: Encoder-decoder approaches. *CoRR* abs/1409.1259. http://arxiv.org/abs/1409.1259.

Michael Collins. 2002. Discriminative Training Methods for Hidden Markov Models: Theory and Experiments with Perceptron Algorithms. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, pages 1–8. https://doi.org/10.3115/1118693.1118694.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *The Journal of Machine Learning Research* 12:2493–2537.

Timothy Dozat and Christopher D. Manning. 2017. Deep biaffine attention for neural dependency parsing. In *5th International Conference on Learning Representations (ICLR 2017)*.

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 Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. Association for Computational Linguistics, Beijing, China, pages 334–343. http://www.aclweb.org/anthology/P15-1033.

W. N. Francis and H. Kucera. 1979. Brown corpus manual. Technical report, Department of Linguistics, Brown University, Providence, Rhode Island, US. http://icame.uib.no/brown/bcm.html.

Yoav Goldberg, Francesco Sartorio, and Giorgio Satta. 2014. A tabular method for dynamic oracles in transition-based parsing. *TACL* 2:119–130. http://www.aclweb.org/anthology/Q14-1010.

Alex Graves and Jürgen Schmidhuber. 2005. Framewise phoneme classification with bidirectional lstm and other neural network architectures. *Neural Networks* pages 5–6.

Kazuma Hashimoto, Caiming Xiong, Yoshimasa Tsuruoka, and Richard Socher. 2016. A joint many-task model: Growing a neural network for multiple NLP tasks. *CoRR* abs/1611.01587. http://arxiv.org/abs/1611.01587.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Comput.* 9(8):1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735.

Richard Johansson. 2017. EPE 2017: The Trento–Gothenburg opinion extraction system. In *Proceedings of the 2017 Shared Task on Extrinsic Parser Evaluation at the Fourth International Conference on Dependency Linguistics and the 15th International Conference on Parsing Technologies*. Pisa, Italy, page 27 – 35.

Diederik P. Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *CoRR* abs/1412.6980. http://arxiv.org/abs/1412.6980.

Eliyahu Kiperwasser and Yoav Goldberg. 2016. Simple and accurate dependency parsing using bidirectional LSTM feature representations. *TACL* 4:313–327. https://transacl.org/ojs/index.php/tacl/article/view/885.

Lingpeng Kong, Chris Alberti, Daniel Andor, Ivan Bogatyy, and David Weiss. 2017. DRAGNN: A transition-based framework for dynamically connected neural networks. *CoRR* abs/1703.04474. http://arxiv.org/abs/1703.04474.

Adhiguna Kuncoro, Miguel Ballesteros, Lingpeng Kong, Chris Dyer, Graham Neubig, and Noah A. Smith. 2016. What do recurrent neural network grammars learn about syntax? *CoRR* abs/1611.05774. http://arxiv.org/abs/1611.05774.

Emanuele Lapponi, Stephan Oepen, and Lilja Øvrelid. 2017. EPE 2017: The Sherlock negation resolution downstream application. In *Proceedings of the 2017 Shared Task on Extrinsic Parser Evaluation at the Fourth International Conference on Dependency Linguistics and the 15th International Conference on Parsing Technologies*. Pisa, Italy, page 21 – 26.

Junhui Li, Deyi Xiong, Zhaopeng Tu, Muhua Zhu, Min Zhang, and Guodong Zhou. 2017. Modeling source syntax for neural machine translation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, Vancouver, Canada, pages 688–697. http://aclweb.org/anthology/P17-1064.

Wang Ling, Tiago Luís, Luís Marujo, Ramón Fernandez Astudillo, Silvio Amir, Chris Dyer, Alan W. Black, and Isabel Trancoso. 2015. Finding function in form: Compositional character models for open vocabulary word representation. *CoRR* abs/1508.02096. http://arxiv.org/abs/1508.02096.

Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a Large Annotated Corpus of English: The Penn Treebank. *COMPUTATIONAL LINGUISTICS* 19(2):313–330.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013.*

Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States.. pages 3111–3119. http://papers.nips.cc/paper/5021-distributed-representations-of-words-and-phrases-and-their-compositionality.

Joakim Nivre. 2008. Algorithms for deterministic incremental dependency parsing. *Comput. Linguist.* 34(4):513–553. https://doi.org/10.1162/coli.07-056-R1-07-027.

Joakim Nivre, Željko Agić, Lars Ahrenberg, et al. 2017. Universal Dependencies 2.0. LINDAT/CLARIN digital library at the Institute of Formal and Applied Linguistics, Charles University, Prague. http://hdl.handle.net/11234/1-1983.

Joakim Nivre, Marie-Catherine de Marneffe, Filip Ginter, Yoav Goldberg, Jan Hajič, Christopher Manning, Ryan McDonald, Slav Petrov, Sampo Pyysalo, Natalia Silveira, Reut Tsarfaty, and Daniel Zeman. 2016. Universal Dependencies v1: A multilingual treebank collection. In *Proceedings of the 10th International Conference on Language Resources and Evaluation (LREC 2016)*. European Language Resources Association, Portorož, Slovenia, pages 1659–1666.

Stephan Oepen, Lilja Øvrelid, Jari Björne, Richard Johansson, Emanuele Lapponi, Filip Ginter, and Erik Velldal. 2017. The 2017 Shared Task on Extrinsic Parser Evaluation. Towards a reusable community infrastructure. In *Proceedings of the 2017 Shared Task on Extrinsic Parser Evaluation at the Fourth International Conference on Dependency Linguistics and the 15th International Conference on Parsing Technologies*. Pisa, Italy, page 1 – 12.

Tomoko Ohta, Yuka Tateisi, and Jin-Dong Kim. 2002. The genia corpus: An annotated research abstract corpus in molecular biology domain. In *Proceedings of the Second International Conference on Human Language Technology Research*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, HLT '02, pages 82–86. http://dl.acm.org/citation.cfm?id=1289189.1289260.

Maxim Rabinovich, Mitchell Stern, and Dan Klein. 2017. Abstract syntax networks for code generation and semantic parsing. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, Vancouver, Canada, pages 1139–1149. http://aclweb.org/anthology/P17-1105.

Drahomíra "johanka" Spoustová, Jan Hajič, Jan Raab, and Miroslav Spousta. 2009. Semi-Supervised Training for the Averaged Perceptron POS Tagger. In *Proceedings of the 12th Conference of the European Chapter of the ACL (EACL 2009)*. Association for Computational Linguistics, Athens, Greece, pages 763–771. http://www.aclweb.org/anthology/E09-1087.

Milan Straka, Jan Hajič, and Jana Straková. 2016. UDPipe: trainable pipeline for processing CoNLL-U files performing tokenization, morphological analysis, POS tagging and parsing. In *Proceedings of the 10th International Conference on Language Resources and Evaluation (LREC 2016)*. European Language Resources Association, Portorož, Slovenia.

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)*.

Milan Straka and Jana Straková. 2017. Tokenizing, pos tagging, lemmatizing and parsing ud 2.0 with udpipe. In *Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*. Association for Computational Linguistics, Vancouver, Canada, pages 88–99. http://www.aclweb.org/anthology/K/K17/K17-3009.pdf.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks. *CoRR* abs/1409.3215. http://arxiv.org/abs/1409.3215.

Kai Sheng Tai, Richard Socher, and Christopher D. Manning. 2015. Improved semantic representations from tree-structured long short-term memory networks. *CoRR* abs/1503.00075. http://arxiv.org/abs/1503.00075.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *CoRR* abs/1706.03762. http://arxiv.org/abs/1706.03762.

Shuangzhi Wu, Dongdong Zhang, Nan Yang, Mu Li, and Ming Zhou. 2017. Sequence-to-dependency neural machine translation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, Vancouver, Canada, pages 698–707. http://aclweb.org/anthology/P17-1065.

Zhilin Yang, Ruslan Salakhutdinov, and William W. Cohen. 2016. Multi-task cross-lingual sequence tagging from scratch. *CoRR* abs/1603.06270. http://arxiv.org/abs/1603.06270.

Dani Yogatama, Phil Blunsom, Chris Dyer, Edward Grefenstette, and Wang Ling. 2016. Learning to compose words into sentences with reinforcement learning. *CoRR* abs/1611.09100. http://arxiv.org/abs/1611.09100.

Daniel Zeman, Martin Popel, Milan Straka, Jan Hajič, Joakim Nivre, Filip Ginter, Juhani Luotolahti, Sampo Pyysalo, Slav Petrov, Martin Potthast, Francis Tyers, Elena Badmaeva, Memduh Gökırmak,

Anna Nedoluzhko, Silvie Cinková, Jan Hajič jr., Jaroslava Hlaváčová, Václava Kettnerová, Zdeňka Urešová, Jenna Kanerva, Stina Ojala, Anna Missilä, Christopher Manning, Sebastian Schuster, Siva Reddy, Dima Taji, Nizar Habash, Herman Leung, Marie-Catherine de Marneffe, Manuela Sanguinetti, Maria Simi, Hiroshi Kanayama, Valeria de Paiva, Kira Droganova, Hěctor Martínez Alonso, Hans Uszkoreit, Vivien Macketanz, Aljoscha Burchardt, Kim Harris, Katrin Marheinecke, Georg Rehm, Tolga Kayadelen, Mohammed Attia, Ali Elkahky, Zhuoran Yu, Emily Pitler, Saran Lertpradit, Michael Mandl, Jesse Kirchner, Hector Fernandez Alcalde, Jana Strnadova, Esha Banerjee, Ruli Manurung, Antonio Stella, Atsuko Shimada, Sookyoung Kwak, Gustavo Mendonça, Tatiana Lando, Rattima Nitisaroj, and Josie Li. 2017. CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies. In *Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*. Association for Computational Linguistics.