UParse: the Edinburgh system for the CoNLL 2017 UD shared task

Clara Vania, Xingxing Zhang, and **Adam Lopez**
Institute for Language, Cognition and Computation
School of Informatics
University of Edinburgh
{c.vania, x.zhang}@ed.ac.uk, alopez@inf.ed.ac.uk

Abstract

This paper presents our submissions for the CoNLL 2017 UD Shared Task. Our parser, called UParse, is based on a neural network graph-based dependency parser. The parser uses features from a bidirectional LSTM to produce a distribution over possible heads for each word in the sentence. To allow transfer learning for low-resource treebanks and surprise languages, we train several multilingual models for related languages, grouped by their genus and language families. Out of 33 participants, our system achieves rank 9th in the main results, with 75.49 UAS and 68.87 LAS F-1 scores (average across 81 treebanks).

1 Introduction

Dependency parsing aims to automatically extract dependencies between words in a sentence, in the form of tree structure. These dependencies define the grammatical structure of the sentence, which makes it beneficial for many natural language applications, such as question answering (Cui et al., 2005), machine translation (Carreras and Collins, 2009), and information extraction (Angeli et al., 2015). The most common approaches for dependency parsing are transition-based (Nivre et al., 2006) or graph-based (McDonald et al., 2005). Recent works also apply neural network approaches for dependency parsing (Chen and Manning, 2014; Dyer et al., 2015; Kiperwasser and Goldberg, 2016; Zhang et al., 2017), particularly for learning rich feature representations that improve parser accuracy.

To train a high-quality parser, one typically needs a large treebank, annotated with some linguistic information, such as part of speech (POS) tags, lemmas, and morphological features. However, human annotations are expensive. As a result, most of the work has been focused on few languages, such as English, Czech, or Chinese.

The Universal Dependencies (UD; Nivre et al. (2016)) is an initiative to develop consistent treebank annotations across many languages. It provides an opportunity to perform model transfer – using model trained on high-resource languages to parse low-resource languages, allowing the development of treebanks for many more languages. Several works (McDonald et al., 2011; Zhang and Barzilay, 2015; Duong et al., 2015a,b; Guo et al., 2015, 2016) have shown that this technique can help improve accuracy for low-resource languages, and in fact recent work of Ammar et al. (2016) demonstrated that it is possible to train a single multilingual model that works well both in low-resource and high-resource settings.

The CoNLL 2017 UD Shared Task (Zeman et al., 2017) uses Universal Dependencies version 2.0 (Nivre et al., 2017), with training data consists of 64 treebanks from 45 languages. Some of the challenges are the truly low-resource treebanks (e.g., Kazakh and Uyghur with only 30 and 100 training sentences, respectively), small treebanks without development data (e.g., Irish, French-ParTUT, Galician-TreeGal, Ukrainian), and the surprise languages and treebanks needed to be parse during test phase.

To address these challenges, we designed our system for the shared task to use both monolingual and multilingual models. In particular:

- We train one monolingual model per high-resource treebank in the training set.
- For low-resource treebanks, we train several multilingual models, each for related languages grouped by their genus and language families.

Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies, pages 100–110,
Vancouver, Canada, August 3-4, 2017.

- For surprise languages, we train several delexalized parsers using treebanks that are closest to the surprise languages in terms of language family.

Our parsing model uses pretrained word vectors, gold universal POS tags (UPOS), and gold morphological analysis (XFEATS, if available). For the multilingual models, we also use language ID and replace pre-trained word vectors with multilingual word vectors. For the delexicalized models, we remove the word vectors from our feature set because we want to use the model for other languages which use different vocabularies.

We submitted three systems, which are described in Section 5. The final ranking of the shared task brings our parser to the ninth place, with average UAS and LAS, 75.49 and 68.87, respectively. On the surprise languages, our system reaches the 6th rank, with 39.17 LAS.

2 System Description

Our system, called UParse, is a combination of monolingual, multilingual, and delexicalized models. In this section, we describe our parsing model which extends DENSE, the neural network graph-based parser of Zhang et al. (2017).

2.1 DENSE Parser

DENSE (**De**pendency **N**eural **Se**lection) is a neural graph-based parser which generates dependency tree by predicting the heads of each word in a sentence. Given an input sentence of length N[1], the parser first produces N ⟨head, dependent⟩ dependency arcs by greedily selecting the most likely head word. If the predicted dependency arcs do not result a (projective) tree structure, a maximum spanning tree algorithm will be used to adjust the output to a (projective) tree. In the following, we will describe the DENSE parser in details.

Token Representations. In the first step, the parser computes the representation of each word in the sentence. The objective is to encode both local (lexical meaning and POS tag) and global information (word position and context). To do this, the parser uses a bidirectional LSTM (bi-LSTMs), which have shown to be effective in capturing long-term dependencies. More formally, let $S = (w_0, w_1, \ldots, w_N)$ be the input sentence of length N, where w_0 denotes the artificial `ROOT` token. Each input token w_i is represented by $\mathbf{x}_i$, which is a concatenation of its word and POS tag embeddings, $\mathbf{e}(w_i)$ and $\mathbf{e}(t_i)$, respectively.

$$\mathbf{x}_i = [\mathbf{e}(w_i); \mathbf{e}(t_i)] \quad (1)$$

These representations are the input to a bi-LSTM, which produces a sentence-specific representation of token w_i computed by concatenating the hidden states of a forward and a backward LSTM:

$$\mathbf{a}_i = [\mathbf{h}_i^f; \mathbf{h}_i^b] \quad (2)$$

where $\mathbf{h}_i^f$ and $\mathbf{h}_i^b$ denotes the hidden states of the forward and backward LSTMs.

Head Predictions. For each token w_i, the parser computes the probability of w_j being the head as:

$$P_{head}(w_j|w_i, S) = \frac{\exp(g(\mathbf{a}_j, \mathbf{a}_i))}{\Sigma_{k=0}^{N} \exp(g(\mathbf{a}_k, \mathbf{a}_i))} \quad (3)$$

where $\mathbf{a}_i$ and $\mathbf{a}_j$ are the word representations of w_i and w_j, respectively. Function g is a neural network with a single layer which computes the associative score between the two words:

$$g(\mathbf{a}_j, \mathbf{a}_i) = \mathbf{v}_a^T \cdot \tanh(\mathbf{U}_a \cdot \mathbf{a}_j + \mathbf{W}_a \cdot \mathbf{a}_i) \quad (4)$$

Note that, this step is similar to the neural attention mechanism in the sequence-to-sequence models (Bahdanau et al., 2015). The model is trained to minimize the negative log likelihood of the gold standard ⟨head, dependent⟩ arcs of all the training sentences. At test time, the parser greedily choose the most probable head for each word in the sentence.

Adjusting Tree Outputs. In many cases, the individual predictions form a tree. However, if this is not the case, a maximum spanning tree (MST) algorithm is used to constrain the set of predictions to form a tree. DENSE can use two algorithms: Chu-Liu-Edmonds (Chu and Liu, 1965; Edmonds, 1967) algorithm to generating non-projective trees; and the Eisner algorithm (Eisner, 1996) to generate projective trees. The decision of the MST algorithms depends on the language's treebank. For the shared task, we assume that each language can produce non-projective trees.

[1] As the convention in dependency parsing, we add a dummy `ROOT` token to the sentence. Therefore, the resulting length of the sentence will be $N + 1$.

Model	Word	Multi-Word	UPOS	XFEATS	LID
MONO	✓		✓	✓	
MULTI		✓	✓	✓	✓
DELEX			✓	✓	

Table 1: Feature set used for each type of model in UParse. Multi-Word denotes the multilingual word embeddings. XFEATS feature is only used if the annotation is available in the training data.

Label Predictions. After obtaining the unlabeled dependency trees, the parser needs to predict labels. To do this, a two-layer rectifier network (Glorot et al., 2011) is used. More formally, to predict the arc label between w_i and w_j, the classifier takes as input the concatenation of the local (Eq. 1) and global (Eq. 2) vector representations of both words, $[\mathbf{a}_i; \mathbf{a}_j; \mathbf{x}_i; \mathbf{x}_j]$ and predicts a valid dependency label.

Zhang et al. (2017) presents more detailed account of the parsing model.

2.2 UParse

Next, we describe UParse, the extended version of DENSE which we use for the UD shared task. As mentioned in Section 1, UParse is a combination of monolingual, multilingual, UDPipe baseline, and delexicalized models. In general, the key difference between DENSE and UParse is in the type of features used for training. UParse uses richer linguistic features, namely word embeddings, universal POS tag (UPOS), morphological analysis (XFEATS), and language ID (LID). This design is mostly inspired by the work of Ammar et al. (2016) and Straka et al. (2016) for monolingual and multilingual parsing models. Each feature is represented by its vector representations and we concatenate them together to represent each input token which will be fed into the bi-LSTMs. Specifically, we modify Eq. 1 to

$$\mathbf{x}_i = [\mathbf{e}(w_i); \mathbf{e}(t_i); \mathbf{e}(m_i); \mathbf{e}(lid_i)] \quad (5)$$

where $\mathbf{e}(m_i)$ and $\mathbf{e}(lid_i)$ denotes the embeddings of XFEATS and language ID, respectively.[2] During training our system uses gold annotations (tokenization, UPOS, and XFEATS) provided in the data. At test time, it uses predicted annotations produced by UDPipe (Straka et al., 2016).

Table 1 shows different feature set used in each type of model in the UParse. We employ the original DENSE architecture for the monolingual models in UParse, with an additional feature (XFEATS, if available). For the multilingual models, we replace the standard word embeddings with multilingual word embeddings (Section 2.3). This is important since we need to project word vectors of different languages to the same vector space. We also use language ID as a feature, to inform the parser about the language of the sentence it is currently parsing. This allows the model to learn not only transferable dependency features across languages, but also the language-specific features.

2.3 Multilingual Word Embeddings

Following Ammar et al. (2016), we adapt the robust projection approach of Guo et al. (2016) to build our multilingual word embeddings. The idea is to train word embeddings of a source language and project them to obtain word embeddings for the target languages. For the shared task, we use English pre-trained word vectors trained on the Wikipedia data (Bojanowski et al., 2016) as our source embeddings. Next, we use OPUS data (Tiedemann, 2012, 2009) to build alignment dictionaries for languages that have parallel text with English. Specifically, we use parallel corpora of Europarl, Global Voices, Wikipedia, and hrWaC (for Croatian).

To build the alignment dictionaries, we use `fast_align` toolkit (Dyer et al., 2013). We then compute vector for each target word using the weighted average of its aligned English word embeddings, weighted by the alignment probabilities. A limitation of this approach is that it creates embeddings for target words that appear in the parallel data. Thus, the final step of this approach also compute embeddings for other target words not aligned with the source words by averaging the embeddings of all aligned target words within an edit distance of 1. The token level embeddings are shared across languages.

3 Preliminary Experiments

Prior to our participation in the shared task, we ran a number of preliminary experiments that informed the design of the final system. Our shared task submission is based on these results.

In our preliminary experiments, our main goal is to evaluate the multilingual model of UParse. These experiments are mainly inspired by the

[2] We treat XFEATS as an atomic symbol.

Type	Model	Languages							Average
		de	en	es	fr	it	pt	sv	
MONO	UDPipe	82.9	87.5	87.1	84.5	90.2	87.2	86.2	86.5
	UParse	**86.8**	**88.7**	**89.2**	87.1	91.4	88.0	88.2	**88.5**
MULTI	UParse	85.9	87.4	88.3	**87.6**	**91.8**	**89.0**	**88.8**	88.4

Table 2: UAS results for monolingual and multilingual model of UParse on the Universal Dependencies version 1.2.

Type	Model	Languages							Average
		de	en	es	fr	it	pt	sv	
MONO	UDPipe	78.6	85.0	84.5	81.0	88.1	84.7	83.2	83.6
	UParse	**80.4**	**85.5**	**85.5**	**83.1**	88.9	84.2	82.7	84.3
MULTI	MALOPA	78.9	85.4	84.3	82.4	**89.1**	86.2	**84.5**	**84.4**
	UParse	77.9	85.1	84.3	81.9	89.0	**86.5**	81.1	83.7

Table 3: LAS results for monolingual and multilingual model of UParse on the Universal Dependencies version 1.2. MALOPA is the multilingual parser of Ammar et al. (2016).

work of Ammar et al. (2016). To compare our results, we use the same datasets from Universal Dependencies version 1.2 (Nivre et al., 2015), for seven languages: English, French, German, Italian, Spanish, Swedish, and Portuguese. The training data for the first five languages consists of more than 10K training sentences, while for Portuguese and Swedish, there are 8.8K and 4.3K training sentences, respectively. For simplicity, we also follow their experimental setup for training optimization (more detail is reported in Section 4). In addition, we also compare our parser performance for the monolingual models with UDPipe parser.

Table 2 and 3 present the performance of our parser compared to UDPipe (monolingual) and MALOPA (multilingual) parsers. In terms of UAS, our multilingual model achieves the best scores, except for English, German, and French. The results for LAS are slightly different. We found that for languages where we have more than 10K training sentences, our monolingual model outperforms the other models, with an exception on Italian. For the smaller treebanks, although we see UAS improvements for Portuguese and Swedish when we use multilingual model, we only obtain LAS improvement on Portuguese. We believe that these mixed results are due to poor accuracy of our label classifier, since the UAS results demonstrate that the parser itself is quite effective in predicting the dependency arcs.

4 Experiments

This section describes the experimental design, training, and also our submissions to the shared task. After looking at the results of our preliminary experiments, we decided to train both monolingual and multilingual parsers, evaluate them on the shared task development data and choose the best settings for our submissions.

4.1 Language Groups

To build the multilingual models, we first group the treebanks such that treebanks of related languages will be trained in a single model. We use genus and language family information taken from the World Atlas of Language Structures(WALS; Dryer and Haspelmath (2013)) to group the languages. For each treebank in which the language is not related to any other treebanks, it will be in a singleton group, hence the same as a monolingual model. For classic languages like Ancient Greek, Latin, Gothic, and Old Church Slavonic, we group them to the same group, instead of using the WALS information. Table 4 shows the language groups used in UParse.

4.2 Training

In the preprocessing step, following the common setup in parsing, we remove multiword tokens and language specific dependency relations. For the multilingual training, we also combine treebanks of the same language in the same training data. We also use two additional datasets: pre-trained

Group	Languages
Classic	Ancient Greek, Latin, Gothic Old Church Slavonic
Finnic	Finnish
Germanic	Danish, Dutch, English, German Norwegian, Swedish
Indic	Hindi, Urdu
Romance	Catalan, French, Italian Portuguese, Spanish
Slavic	Bulgarian, Croatian, Czech Polish, Russian, Slovak Slovenian, Ukrainian
Semitic	Arabic, Hebrew
Turkic	Kazakh, Turkish, Uyghur

Table 4: Language groups used for building UPARSE multilingual models. Finish language has two treebanks, we group them together in the same group.

word embeddings from Bojanowski et al. (2016) and OPUS parallel data (Tiedemann, 2012, 2009).

Unless we explicitly mention in the description, we follow the same training configurations as described in Zhang et al. (2017). We use two-layer bi-LSTMs with 150 hidden units, and set embedding size for {words, UPOS, XFEATS, LID} to {300, 30, 40, 10}, respectively. The word embedding size matches that of the pre-trained embeddings. We did not use the Czech-CLLT or any ParTUT treebanks for training since they contain many long sentences (the longest sentence in the Czech-CLLT treebank consists of 534 words). At test time, we parse these treebanks using the models trained on the same language. We trained our models on an Nvidia GPU card; training a monolingual model takes 1-2 hours, while training a multilingual model takes 4-5 hours.

Word embeddings. For monolingual training, we initialize the embeddings with the pre-trained ones and keep them fixed during training. For the multilingual models, we first create multilingual word embeddings as described in Section 2.3, using OPUS parallel data and English as the source language. Unlike Ammar et al. (2016) and Guo et al. (2016), we also share representations for words which are used by more than one language. For example, if *system* appears in the English and German data, we only use a single vector to represent it. Of course, this means we allow parameter sharing across words with the same forms, but different meanings. But on the other hand, it also enables named entities and loanwords to have the same representation across languages. We initialize the embeddings with the multilingual word embeddings and update them during training.[3] For all models, embeddings for words with no pre-trained representation are initialized uniformly at random in the range [-0.1, 0.1].

Optimization for multilingual training. For multilingual training, we follow Ammar et al. (2016) when updating the parameters. Specifically, we use mini-batch updates in which we uniformly sampled (without replacement) the same number of sentences for each treebank, until all sentences in the smallest treebank are used. In other words, each epoch will use $N \times L$ sentences, where N is the number of sentences in the smallest treebank and L is the number of languages.

4.3 Truly Low-Resource Treebanks

There are some challenges when training the truly low-resource treebanks, i.e., treebanks with less than 2K sentences, with no other treebanks from the same language available. For example, Vietnamese treebank only has 1400 sentences with no related languages in terms of genus and language family. Ideally, we want to apply multilingual learning for these treebanks since we do not have enough examples to train them using monolingual models. Moreover, languages like Kazakh and Uyghur have 100 training sentences or fewer and no development data, which makes it difficult to do multilingual training as described above. Our initial experiments show that multilingual learning helps improve accuracy of the truly low-resource treebanks (with less than 1K training sentences), but degrades accuracy of the high-resource treebanks. This is because using our training set up, each epoch will only consists of small number of sentences per language. Irish is particularly challenging, with only 566 training sentences, no development data, and no related languages. Our training strategy for these particular cases are as follows:

Estonian and Hungarian. These languages are belong to the Uralic language family. Since Finnish has two treebanks with large training data, we train two more multilingual models for each, using additional Finnish treebanks.

[3] We did not fix the embeddings since in our preliminary experiments, it gave us lower accuracy.

We do not use a single model to train both Estonian and Hungarian since Estonian has more training sentences than Hungarian.

Greek. We train a multilingual model for Greek, using training data from Ancient Greek and Greek treebanks.[4]

Irish. Since this language does not have any related languages, we use delexicalized model of Czech. We chose Czech since the language has the largest treebank.

Kazakh and Uyghur. For the two languages, since the training data are very small, we use a single delexicalized model of Turkish. We only use Turkish data during training, but include both Kazakh and Uyghur training data in the development set.

4.4 Surprise Languages and Treebanks

For the surprise languages, since we do not have any training data, we train delexicalized models on related languages. In particular, we use delexicalized Russian for Buryat, Persian for Kurmanji, Finnic for North Sami, and Czech for Upper Sorbian. Note that the delexicalized models of Russian, Finnic, and Czech use all the treebanks of the language, thus allowing transfer learning between different treebanks of the same language. For example, to train a delexicalized model of Russian, we use both UD_Russian and UD_Russian-SynTagRus treebanks.

For the surprise treebanks from known languages, we simply use a parser trained on other treebanks in that language.

5 Results and Analysis

5.1 Initial Results on Development Data

During the training phase, we evaluated the performance of our monolingual and multilingual systems using the official development data. Since we use gold annotations (tokenization, UPOS, and XFEATS) as our features, we compare our performance with UDPipe baseline which also use gold annotations. Table 5 shows the average UAS and LAS of the monolingual and multilingual systems. Similar to our preliminary results, we see improvements on UAS for the multilingual model, but with LAS lower than the monolingual or even the UDPipe system. When we look at the results for individual treebanks, we found that our models are especially achieved lower LAS than the baseline system on the smaller treebanks.

Model	Avg. UAS	Avg. LAS
Baseline	83.29	**79.53**
MONO	79.53	78.44
MULTI	**85.76**	77.55

Table 5: Average UAS and LAS of the monolingual versus multilingual models. The baseline is UDPipe with gold annotations.

5.2 Submission

The UD shared task employs TIRA (Potthast et al., 2014) to evaluate all systems. When we deployed our system on the TIRA virtual machine, we encountered two problems which break the evaluation script. First, our system sometimes produces multiple roots in the prediction, which the script rejects. To address this, we post-processed the predicted tree by taking the first prediction as the root, and connect other roots to the first root with a clausal component label, `ccomp`.[5] The second problem occurs when the test data has sentence longer than the maximum sentence length in the training data.[6] Because we had limited time to address this, we used the following algorithm: Let n be the maximum length of sentence allowed by the parsing model. For each sentences with length k, where $k > n$:

1. Parse the first n words in the sentence.
2. For the rest $k - n$ words, connect each word with the previous word, and label the arc between them using a heuristic label (`DIST`), or a random label (`RAND`). We simply take the most frequent label between the head POS and the dependent POS in the training data for `DIST`.

We decided to use the combination of monolingual, multilingual, UDPipe (only for the primary system, `UP-1`), and delexicalized models for our primary system. For each treebank, we pick the

[4]This specific model is slightly different, one might assume that Ancient Greek and Modern Greek are highly related.

[5]We choose this label based from our observation on the multiple roots prediction. Most of the time, our parser predicts multiple roots if the sentences are too long and contain multiple clauses.

[6]In the current training setup, the maximum sentence length is fixed.

Model	Avg. LAS
UP-1	**73.66**
UP-2	73.30
UP-3	73.29

Table 6: Macro-averaged LAS F1 score on development data.

Treebank name	Model
Estonian*	Finnic-Estonian
Gothic	Classic
Hungarian*	Finnic-Hungarian
Irish*	DEL-Czech
Kazakh*	DEL-Turkic
Old Church Slavonic*	Classic
Slovak*	Slavic
Swedish	Germanic
Swedish-LinES	Germanic
Uyghur*	DEL-Turkic
Buryat	DEL-Russian
Kurmanji	DEL-Persian
North_Sami	DEL-Finnic
Upper_Sorbian	DEL-Czech

Table 7: List of treebanks which use multilingual or delexicalized models in UParse. (*) denotes treebanks which use UDPipe models in UP-1. The bottom part of the table shows the models used to parse surprise languages.

best model based on its performance on the development data. We use UDPipe models for 24 treebanks in which we achieved lower performance than the baseline on the development data (denoted by (*) in Table 9). UP-2 and UP-3 do not use any UDPipe models. Table 7 lists all treebanks which use multilingual or delexicalized models for parsing. Our final submission consists of three different systems:

1. UP-1: UParse + DIST + UDPipe
2. UP-2: UParse + DIST
3. UP-3: UParse + RAND

Table 6 shows the macro-averaged LAS F1 scores for all the systems.

5.3 Results on Test Data

Table 8 shows the results of our primary system of LAS, UAS, and CLAS (Nivre and Fang, 2017). The more detailed results for each treebank and system is given in Table 9. Similar to the results

Metric	Score
LAS	68.87
UAS	75.49
CLAS	63.55

Table 8: LAS, UAS, and CLAS results of our primary system, UP-1.

on development data (Table 6), UP-1 achieves the best macro-average F1 score out of the three systems. The results of UP-2 and UP-3 are quite similar, which is not surprising since there are only a few long sentences in the test data.

We further observe the performance of the UDPipe baseline model versus UParse models, by comparing the performance of UP-1 and UP-2 on the 24 treebanks (treebanks with (*) in Table 9). Based on the results, our system achieves lower LAS-F1 scores on 16 treebanks, which are either treebanks with small training data or treebanks with long sentences, for which we did not train any model. For the other six treebanks, our system achieves higher LAS-F1 scores than the UDPipe baseline system, with 4 treebanks predicted using the multilingual models.

Our system is deployed on the TIRA virtual machine, which is a quad-core CPU with 16GB RAM. It took 2 hours and 43 minutes for our primary system to parse the official test data.

6 Conclusion and Future Work

We described UParse, our system for the CoNLL UD Shared Task 2017. Our observation from the overall results suggested that our parsing model outperforms the UDPipe baseline model, except in cases when there is little training data available. Our approach to perform multilingual learning by transferring models from high-resource to low-resource treebank seems to be quite effective in predicting the dependency arcs, but less for the label predictions. However, we observed some improvements for a number of treebanks when we use a multilingual model trained using treebanks from related languages.

In the light of these results, some possible directions for the future work include improving the label predictions of the parsing model and exploring the possibilities to use character-level models, as they have shown to be effective for parsing morphologically rich languages (Ballesteros et al., 2015). Another interesting direction is to combine

Treebank Code	LAS F-1 score		
	UP-1	UP-2	UP-3
ar_pud	45.3	45.3	45.3
ar	66.35	66.35	66.3
bg*	**83.64**	83.46	83.46
bxr	21.63	21.63	21.63
ca	86.8	86.8	86.8
cs_cac	85.57	85.57	85.57
cs_cltt*	**71.64**	66.74	66.37
cs_pud	81.06	81.06	81.06
cs	85.24	85.24	85.24
cu*	62.76	**64.24**	64.24
da	73.46	73.46	73.46
de_pud	67.36	67.36	67.36
de	70.09	70.09	70.09
el*	**79.26**	76.93	76.93
en_lines	73.28	73.28	73.28
en_partut*	**73.64**	69.63	69.63
en_pud	79.54	79.54	79.54
en	76.42	76.42	76.41
es_ancora	86.01	86.01	86.01
es_pud	79.2	79.2	79.2
es	83.02	83.02	83.02
et*	**58.78**	56.26	56.26
eu	69.85	69.85	69.85
fa	79.97	79.97	79.97
fi_ftb*	**74.04**	73.77	73.77
fi_pud	79.66	79.66	79.66
fi	75.35	75.35	75.35
fr_partut*	**77.38**	76.05	76.05
fr_pud	74.44	74.44	74.44
fr_sequoia	78.57	78.57	78.57
fr	81.58	81.58	81.58
ga*	**61.52**	36.31	36.2
gl_treegal	64.18	64.18	64.18
gl	78.08	78.08	78.08
got	60.71	60.71	60.71
grc_proiel	64.48	64.48	64.45
grc	57.22	57.22	57.22
he	57.6	57.6	57.6
hi_pud	51.89	51.89	51.89
hi	87.2	87.2	87.2
hr*	**77.18**	76.28	76.28

Treebank Code	LAS F-1 score		
	UP-1	UP-2	UP-3
hsb	59.24	59.24	59.24
hu*	**64.3**	57.37	57.37
id	75.01	75.01	75.01
it_pud	85.13	85.13	85.13
it	86.62	86.62	86.62
ja_pud	74.64	74.64	74.64
ja*	**72.21**	70.51	70.51
kk	21.96	21.96	21.96
kmr	39.76	39.76	39.76
ko*	**59.09**	58.74	58.74
la_ittb	79.35	79.35	79.35
la_proiel	56.93	56.93	56.91
la*	43.77	**46.07**	46.07
lv*	**59.95**	57.09	57.09
nl_lassysmall	79.56	79.56	79.56
nl	69.9	69.9	69.9
no_bokmaal	83.81	83.81	83.81
no_nynorsk	81.91	81.91	81.91
pl*	78.78	**79.69**	79.69
pt_br	86.38	86.38	86.38
pt_pud	74.76	74.76	74.76
pt	83.12	83.12	83.12
ro	80.45	80.45	80.45
ru_pud	68.64	68.64	68.64
ru_syntagrus	89.18	89.18	89.18
ru*	74.03	**74.86**	74.76
sk*	72.75	**74.77**	74.77
sl_sst	46.97	46.97	46.97
sl*	**81.15**	81.09	81.09
sme	36.04	36.04	36.04
sv_lines	74.04	74.04	74.04
sv_pud	70.44	70.44	70.44
sv	75.29	75.29	75.29
tr_pud	32.63	32.63	32.63
tr*	**53.22**	51.69	51.69
ug*	**34.18**	20.8	20.8
uk*	60.76	**60.78**	60.78
ur	76.35	76.35	76.35
vi*	**37.47**	37.14	37.14
zh*	**57.4**	56.14	56.14
Average LAS	**68.87**	68.09	68.09

Table 9: LAS F-1 scores for each treebank in the test data. (*) denotes treebanks which are predicted using UDPipe baseline models in the UP-1 system and the best accuracies are shown in **bold**.

both morphological analysis and also sub-word unit representation (characters, character n-grams, or morphemes) and investigate whether these features are transferable across languages with similar typology.

Acknowledgments

We would like to thank Sameer Bansal, Jianpeng Cheng, Jonathan Mallinson, and the anonymous reviewers for the helpful feedbacks. Clara Vania is supported by the Indonesian Endowment Fund for Education (LPDP), the Centre for Doctoral Training in Data Science, funded by the UK EPSRC (grant EP/L016427/1), and the University of Edinburgh.

References

Waleed Ammar, George Mulcaire, Miguel Ballesteros, Chris Dyer, and Noah Smith. 2016. Many languages, one parser. *Transactions of the Association for Computational Linguistics* 4:431–444. https://www.transacl.org/ojs/index.php/tacl/article/view/892.

Gabor Angeli, Melvin Jose Johnson Premkumar, and Christopher D. Manning. 2015. Leveraging linguistic structure for open domain information extraction. 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 344–354. http://www.aclweb.org/anthology/P15-1034.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. *Proceedings of the 3rd International Conference on Learning Representations* .

Miguel Ballesteros, Chris Dyer, and Noah A. Smith. 2015. Improved transition-based parsing by modeling characters instead of words with lstms. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Lisbon, Portugal, pages 349–359. http://aclweb.org/anthology/D15-1041.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2016. Enriching word vectors with subword information. *arXiv preprint arXiv:1607.04606* .

Xavier Carreras and Michael Collins. 2009. Non-projective parsing for statistical machine translation. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 1 - Volume 1*. Association for Computational Linguistics, Stroudsburg, PA, USA, EMNLP '09, pages 200–209. http://dl.acm.org/citation.cfm?id=1699510.1699537.

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.

Y. J. Chu and T. H. Liu. 1965. On the shortest arborescence of a directed graph. *Science Sinica* 14.

Hang Cui, Renxu Sun, Keya Li, Min-Yen Kan, and Tat-Seng Chua. 2005. Question answering passage retrieval using dependency relations. In *Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, New York, NY, USA, SIGIR '05, pages 400–407. https://doi.org/10.1145/1076034.1076103.

Matthew S. Dryer and Martin Haspelmath, editors. 2013. *WALS Online*. Max Planck Institute for Evolutionary Anthropology, Leipzig. http://wals.info/.

Long Duong, Trevor Cohn, Steven Bird, and Paul Cook. 2015a. Low resource dependency parsing: Cross-lingual parameter sharing in a neural network parser. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*. Association for Computational Linguistics, Beijing, China, pages 845–850. http://www.aclweb.org/anthology/P15-2139.

Long Duong, Trevor Cohn, Steven Bird, and Paul Cook. 2015b. A neural network model for low-resource universal dependency parsing. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Lisbon, Portugal, pages 339–348. http://aclweb.org/anthology/D15-1040.

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.

Chris Dyer, Victor Chahuneau, and Noah A. Smith. 2013. A simple, fast, and effective reparameterization of ibm model 2. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human*

Language Technologies. Association for Computational Linguistics, Atlanta, Georgia, pages 644–648. http://www.aclweb.org/anthology/N13-1073.

Jack Edmonds. 1967. Optimum branchings. *Journal of Research of the National Bureau of Standards B* 71(4):233–240.

Jason Eisner. 1996. Efficient normal-form parsing for combinatory categorial grammar. In *Proceedings of the 34th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, Santa Cruz, California, USA, pages 79–86. https://doi.org/10.3115/981863.981874.

Xavier Glorot, Antoine Bordes, and Yoshua Bengio. 2011. Deep sparse rectifier neural networks. In *International Conference on Artificial Intelligence and Statistics*. Cadiz, Spain, pages 315–323.

Jiang Guo, Wanxiang Che, David Yarowsky, Haifeng Wang, and Ting Liu. 2015. Cross-lingual dependency parsing based on distributed representations. 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 1234–1244. http://www.aclweb.org/anthology/P15-1119.

Jiang Guo, Wanxiang Che, David Yarowsky, Haifeng Wang, and Ting Liu. 2016. A representation learning framework for multi-source transfer parsing. https://www.aaai.org/ocs/index.php/AAAI/AAAI16/paper/view/12236.

Eliyahu Kiperwasser and Yoav Goldberg. 2016. Simple and accurate dependency parsing using bidirectional lstm feature representations. *Transactions of the Association for Computational Linguistics* 4:313–327. https://transacl.org/ojs/index.php/tacl/article/view/885.

Ryan McDonald, Koby Crammer, and Fernando Pereira. 2005. Online large-margin training of dependency parsers. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*. Association for Computational Linguistics, Ann Arbor, Michigan, pages 91–98. https://doi.org/10.3115/1219840.1219852.

Ryan McDonald, Slav Petrov, and Keith Hall. 2011. Multi-source transfer of delexicalized dependency parsers. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Edinburgh, Scotland, UK., pages 62–72. http://www.aclweb.org/anthology/D11-1006.

Joakim Nivre, Željko Agić, Maria Jesus Aranzabe, Masayuki Asahara, Aitziber Atutxa, Miguel Ballesteros, John Bauer, Kepa Bengoetxea, Riyaz Ahmad Bhat, Cristina Bosco, Sam Bowman, Giuseppe G. A. Celano, Miriam Connor, Marie-Catherine de Marneffe, Arantza Diaz de Ilarraza, Kaja Dobrovoljc, Timothy Dozat, Tomaž Erjavec, Richárd Farkas, Jennifer Foster, Daniel Galbraith, Filip Ginter, Iakes Goenaga, Koldo Gojenola, Yoav Goldberg, Berta Gonzales, Bruno Guillaume, Jan Hajič, Dag Haug, Radu Ion, Elena Irimia, Anders Johannsen, Hiroshi Kanayama, Jenna Kanerva, Simon Krek, Veronika Laippala, Alessandro Lenci, Nikola Ljubešić, Teresa Lynn, Christopher Manning, Ctlina Mrnduc, David Mareček, Héctor Martínez Alonso, Jan Mašek, Yuji Matsumoto, Ryan McDonald, Anna Missilä, Verginica Mititelu, Yusuke Miyao, Simonetta Montemagni, Shunsuke Mori, Hanna Nurmi, Petya Osenova, Lilja Øvrelid, Elena Pascual, Marco Passarotti, Cenel-Augusto Perez, Slav Petrov, Jussi Piitulainen, Barbara Plank, Martin Popel, Prokopis Prokopidis, Sampo Pyysalo, Loganathan Ramasamy, Rudolf Rosa, Shadi Saleh, Sebastian Schuster, Wolfgang Seeker, Mojgan Seraji, Natalia Silveira, Maria Simi, Radu Simionescu, Katalin Simkó, Kiril Simov, Aaron Smith, Jan Štěpánek, Alane Suhr, Zsolt Szántó, Takaaki Tanaka, Reut Tsarfaty, Sumire Uematsu, Larraitz Uria, Viktor Varga, Veronika Vincze, Zdeněk Žabokrtský, Daniel Zeman, and Hanzhi Zhu. 2015. Universal dependencies 1.2 LINDAT/CLARIN digital library at Institute of Formal and Applied Linguistics, Charles University in Prague. http://hdl.handle.net/11234/1-1548.

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, Portoro, Slovenia, pages 1659–1666.

Joakim Nivre and Chiao-Ting Fang. 2017. Universal dependency evaluation. In *Proceedings of the NoDaLiDa 2017 Workshop on Universal Dependencies (UDW 2017)*. pages 86–95.

Joakim Nivre, Johan Hall, Jens Nilsson, Gülşen Eryiğit, and Svetoslav Marinov. 2006. Labeled pseudo-projective dependency parsing with support vector machines. In *Proceedings of the Tenth Conference on Computational Natural Language Learning (CoNLL-X)*. Association for Computational Linguistics, New York City, pages 221–225. http://www.aclweb.org/anthology/W/W06/W06-2933.

Joakim Nivre 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`. http://hdl.handle.net/11234/1-1983.

Martin Potthast, Tim Gollub, Francisco Rangel, Paolo Rosso, Efstathios Stamatatos, and Benno Stein. 2014. Improving the reproducibility of PAN's shared tasks: Plagiarism detection, author identification, and author profiling. In Evangelos Kanoulas, Mihai Lupu, Paul Clough, Mark Sanderson, Mark Hall, Allan Hanbury, and Elaine Toms, editors, *Information Access Evaluation meets Multilinguality, Multimodality, and Visualization. 5th International Conference of the CLEF Initiative (CLEF 14)*. Springer, Berlin Heidelberg New York, pages 268–299. https://doi.org/10.1007/978-3-319-11382-1_22.

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, Portoro, Slovenia.

Jörg Tiedemann. 2009. News from OPUS - A collection of multilingual parallel corpora with tools and interfaces. In N. Nicolov, K. Bontcheva, G. Angelova, and R. Mitkov, editors, *Recent Advances in Natural Language Processing*, John Benjamins, Amsterdam/Philadelphia, Borovets, Bulgaria, volume V, pages 237–248.

Jörg Tiedemann. 2012. Parallel data, tools and interfaces in opus. In Nicoletta Calzolari (Conference Chair), Khalid Choukri, Thierry Declerck, Mehmet Ugur Dogan, Bente Maegaard, Joseph Mariani, Jan Odijk, and Stelios Piperidis, editors, *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12)*. European Language Resources Association (ELRA), Istanbul, Turkey.

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.

Xingxing Zhang, Jianpeng Cheng, and Mirella Lapata. 2017. Dependency parsing as head selection. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*. Association for Computational Linguistics, Valencia, Spain, pages 665–676. http://www.aclweb.org/anthology/E17-1063.

Yuan Zhang and Regina Barzilay. 2015. Hierarchical low-rank tensors for multilingual transfer parsing. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Lisbon, Portugal, pages 1857–1867. http://aclweb.org/anthology/D15-1213.