A Simple yet Effective Joint Training Method for Cross-Lingual Universal Dependency Parsing

Danlu Chen*[1], Mengxiao Lin*[12], Zhifeng Hu*[1], Xipeng Qiu[1]
[1]Fudan University
[2]Megvii Inc
danluchen@fb.com, linmengxiao@megvii.com, {zfhu16,xipeng}@fudan.edu.cn

Abstract

This paper describes Fudan's submission to CoNLL 2018's shared task Universal Dependency Parsing. We jointly train models when two languages are similar according to linguistic typology and then do an ensemble of the models using a simple re-parse algorithm. Our system outperforms the baseline method by 4.4% and 2.1% on the development and test set of *CoNLL 2018 UD Shared Task*, separately.[1]. Our code is available on https://github.com/taineleau/FudanParser.

1 Introduction

Dependency Parsing has been a fundamental task in Natural Language Processing (NLP). Recently, universal dependency parsing (Zeman et al., 2018a,b; Nivre et al., 2018) has unified the annotations of different languages and thus made transfer learning among languages possible. Several works using cross-lingual embedding (Duong et al., 2015; Guo et al., 2015) have successfully increased the accuracy of cross-lingual parsing. Beyond embedding-based methods, a natural question is whether we can use a simple way to utilize the universal information. Some previous research either regarded the universal information as extra training signals (e.g., delexicalized embedding (Dehouck and Denis, 2017)), or implicitly trained a network with all features (e.g., adversarial training for parsing in Sato et al. (2017)). In our system, we manually and explicitly share the universal annotations via a shared LSTM component.

Similar to Vania et al. (2017), different languages are first grouped based on typology, as shown in table 1. Then, we train a shared model for each pair of languages within the same group, and apply a simple ensemble method over all trained models. Note that our method is orthogonal to other cross-lingual approaches for universal parsing such as cross-lingual embedding.

In the following parts, we first describe the baseline method (Section 2) and our system (Section 3). We show the result on both development set and test set in Section 4 and provide some analysis of the model in Section 5.

2 Baseline

In this section, we briefly introduce the baseline system, UDPipe 1.2 (Straka and Straková, 2017), which is an improved version of original UDPipe (Straka et al., 2016). The tokenizing, POS tagging and lemma outputs of UDPipe are utilized by FudanParser.

UDPipe employs a GRU network during the inference of segmentation and tokenization. The tagger uses characters features to predict the POS and lemma tags. Finally, a transition-based neural dependency parser with one hidden layer predicts the transition actions. The parser also makes use of the information from lemmas, POS taggings and dependency relationships through a group of embeddings precomputed by `word2vec`.

In the later discussion, we take the baseline performance result from the web page of the shared task [2] for comparison.

3 System Description

In this submission, we only consider parsing in an end-to-end manner and handle each treebank sep-

* Authors contributed equally.

[1]Unfortunately, we did not finish the run before the deadline. As a result, the official accuracy gain for test set is only 0.54% and we ranks 17th out of 27 teams.

[2]http://universaldependencies.org/conll18/baseline.html

Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies, pages 256–263
Brussels, Belgium, October 31 – November 1, 2018.
https://doi.org/10.18653/v1/K18-2026

Group	Datasets
germanic	Afrikaans-AfriBooms Danish-DDT Dutch-Alpino Dutch-LassySmall English-EWT English-GUM English-LinES German-GSD Gothic-PROIEL Norwegian-Bokmaal Norwegian-Nynorsk Swedish-LinES Swedish-Talbanken
indo-iranian	Hindi-HDTB Persian-Seraji Urdu-UDTB
latin	Latin-ITTB Latin-PROIEL Latvian-LVTB
romance	Catalan-AnCora French-GSD French-Sequoia French-Spoken Galician-CTG Italian-ISDT Italian-PoSTWITA Old_French-SRCMF Portuguese-Bosque Romanian-RRT Spanish-AnCora
semitic	Arabic-PADT Hebrew-HTB
slavic	Bulgarian-BTB Croatian-SET Czech-CAC Czech-FicTree Czech-PDT Old_Church_Slavonic-PROIEL Polish-LFG Polish-SZ Russian-SynTagRus Serbian-SET Slovak-SNK Slovenian-SSJ
turkish	Turkish-IMST Ukrainian-IU Uyghur-UDT
uralic	Estonian-EDT Finnish-FTB Finnish-TDT

Table 1: Grouping languages according to typology.

arately. We first train a monotonic model for all "big" treebanks. Besides, for each language, there are $N-1$ models fine-tuned from joint-trained (see Figure 2), where N is the number of languages in the same language group.

For small treebanks where training set is less than 50 sentences, we use the delexicalized method the same as Shi et al. (2017)'s approach for the surprise languages. Shi et al. (2017) took delexicalized features (morphology and POS tag) as input and apply 50% dropout rate to the input. In practice, we found that the baseline method performs much better than ours on"fi_pud", "br_keb" "ja_modern" and "th_pud", so we use the baseline method instead for these languages.

Our whole system needs about 90 hours to do the inference of all models on TIRA and requires no more than 560M main memory.

3.1 Architecture

Features We use words, characters as the lexical information, and use morphological features[3] and POS tags as the delexicalized information. We also tried subword embeddings, but it mostly did not help. More precisely, the character-level features are treated as bag-of-characters. Similarly, we use bag-of-morphology for morphological features (one can see **number=single** as a character). We first assign the embedding vectors for characters and morphological features, and then for each word, we apply a Convolutional Network (CNN) to encode variable length embeddings into one fixed length feature.

Biaffine BiLSTM. Similar to Shi et al. (2017); Sato et al. (2017); Vania et al. (2017), we use last year's first-place model (Dozat et al., 2017), the graph-based biaffine bizLSTM model as our backbone. Given a sentence of N words, the input is first fed to a bi-directional LSTM and obtain the feature of each word w_i. A head MLP and a dependent MLP are used to translate the features, which is then fed into a hidden layer to calculate the biaffine attention. Finally, we are able to compute the score of arcs and labels in following way:

[3]we take the *features* column of the UD data as the morphological features, which includes case, number, tense, mood and so on. See http://universaldependencies.org/u/feat/index.html for detailed information.

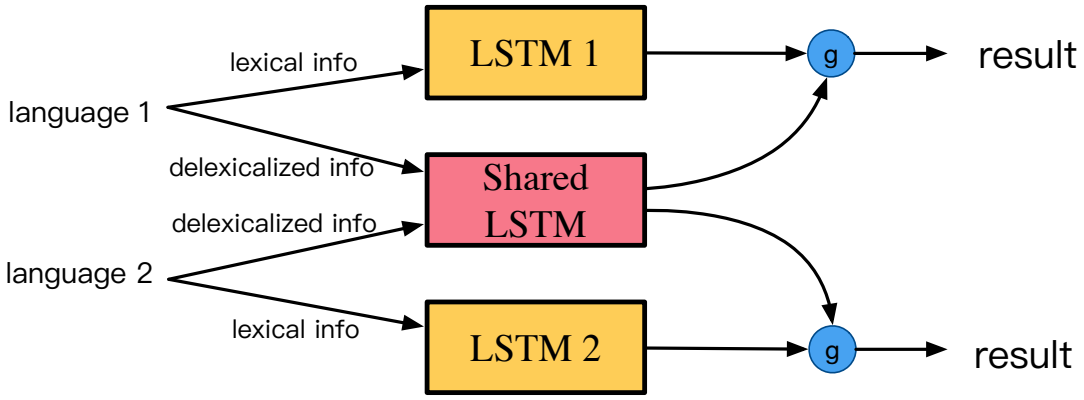

Figure 1: An illustration of the joint training framework for two languages.

$$h_i^h = \text{MLP}_{head}(w_i)$$
$$h_i^d = \text{MLP}_{dep}(w_i)$$
$$s_i = H^h U_1 h_i^d + H^h u_2$$

where $U_1 \in \mathbb{R}^{d \times d}$ and $u_2 \in \mathbb{R}^d$ are trainable parameters.

3.2 Joint Training

For a joint training model of N languages, we have $N+1$ Biaffne Bi-LSTMs (called LSTMs), see Figure 1. For each language, we have a language-specific LSTM to process the lexical information such as word- or character- level embedding, and the output is $w_{i,j}^l$. For all languages we have a shared LSTM which takes delexicalized information such as morphology and POS tags as input and the output is $w_{i,j}^d$. Inspired by Sato et al. (2017), we use a gating mechanism to combine these two set of features. Formally,

$$x = [w_{i,j}^l; w_{i,j}^d],$$
$$g = G(x), y = x \odot g,$$

where w^l indicates lexical feature, w^d indicates delexicalized feature, and $\odot$ is element-wise multiplication.

The difference between Sato et al. (2017) and ours is that we remove the adversarial training loss, which is because we have already use the universal information in the shared network.

3.3 Fine-tuning

We fine-tunning each joint-training model for 100 steps (see Figure 2).

3.4 Tree Ensemble

We follow the re-parsing method proposed in Sagae and Lavie (2006) to perform model ensemble. Suppose k parsing trees have been obtained, denoted by $T_1, T_2, ...T_k$, a new graph is constructed by setting the score of each edge to

$$S[u \to v] = \sum_{i=1}^{k} [u \to v] \in T_k$$

This graph is feed to a MST algorithm to get the ensemble parsing tree T_e. Then the relation label of edge $[u \to v]$ in T_e is voted by all inputs T_i that contains edge $[u \to v]$.

3.5 Hyper-parameters

We followed the hyper-parameter settings in (Dozat et al., 2017). We train $30,000$ steps for each model and then fine-tune (onot necessary) for 100 steps for the given language. For all the input features, the dimension is 100. For LSTM, we use hidden size equals to 400 and the number of layers is 3. 0.33% dropout rate is applied to the input and LSTM hidden layer. We use Bayesian dropout (Gal and Ghahramani, 2016) in the LSTM layers. We also use word dropout (dropping the whole word with a probability) in the input layer.

4 Results

The results of the test and development set are shown in Table 5 and Table 6, respectively. The first three columns are the baseline results and the second three columns are the results of our submission. Also, we list the performance improvement of Fudan Parser compared to the baseline system in the last three columns.

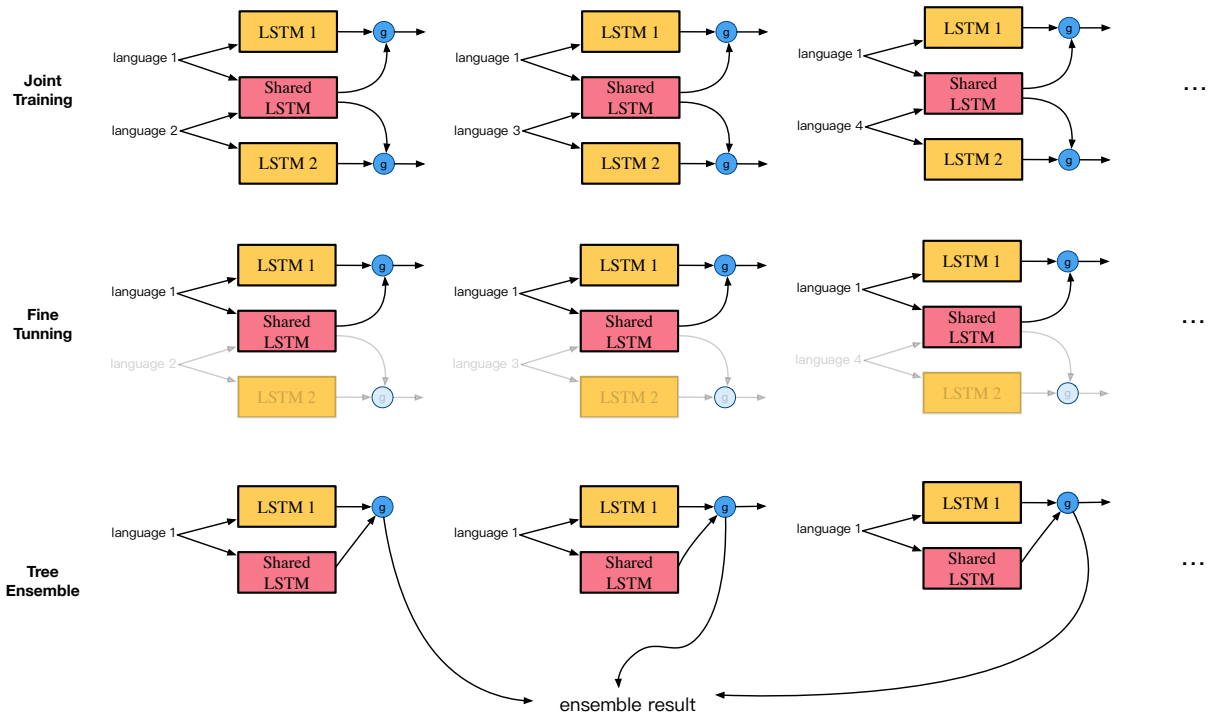

Figure 2: Take four languages as an example. We aim at testing sentence in language 1. We first jointly train languages 1 and other three languages in three separate network. And then we only keep LSTM 1 and the shared LSTM part to fine tune the models for language 1. Finally we re-parse it as an ensemble to obtain the final parsing tree for a given sentence in language 1.

As shown in both Table 6 and 6, we find that our system achieves higher improvements on the datasets with large size of training data. It is reasonable since our model contains enormous parameters, which is easy to get overfitting if the training set is too small. More analysis are included in Section 5.

5 Analysis

5.1 Language similarity

The accuracy of the joint training model actually reveals the syntactic similarity between two languages. The accuracy of three language groups, Slavic (Table 2), Romance (Table 3) and Germanic (Table 4). A number in row i, column j means the accuracy of language i testing on the model jointly training on language i and language j. The **bold** font indicates it is the best model for language i. We can see that for every language, jointly trained models consistently beat single models (the number on the diagonal) which shows the efficacy of the proposed approach.

5.2 Morphology

Morphology is extremely helpful when predicting the dependency between words, especially for those morphology rich languages. However, the UD Parsing task is not done in an end-to-end fashion (i.e. the input morphological features are not the ground-true labels) and thus the morphology information is noisy. The performance is hurt greatly because of the noisy predicted morphology features. A significant accuracy gain should be obtained if a better morphology prediction model is used.

6 Conclusion

Our system provided a simple yet effective method –sharing the universal features to the same part of neural network– to boost the accuracy of syntactic parsing. We also demonstrated that morphological feature plays an important role in syntactic parsing, which is a promising direction to work on.

In the future, we can investigate a better way to do the ensemble or apply a multi-model compression method (e.g. knowledge distillation) to reduce the computational cost. Also, we can explore

Table 2: Slavic languages joint training result.

Acc.(%)	bg	hr	cs	pl	ru	sk	sl	max improvement
bg	92.6	92.5	**92.8**	92.7	92.7	92.3	92.4	0.2
hr	85.7	86	**86.1**	85.2	85.5	85.8	85.5	0.1
cs	91.2	91.2	91.2	91.1	**91.3**	91.3	91.2	0.1
pl	**90.4**	89.8	90.2	90.1	90.2	90.4	90.8	0.3
ru	84.4	84.7	**85.2**	84.4	83.8	84.1	84.6	1.4
sk	86.4	86.2	**87.8**	85.9	86.4	86.7	86.1	1.1
sl	91.4	**91.8**	91.7	91.4	91.4	91	91.2	0.6
Avg.								0.54
# samples	8908	7690	68496	6101	3851	8484	6479	

Table 3: Romance languages joint training result.

Acc. (%)	ca	fr	gl	it	pt	ro	es	max improvement
ca	92.6	92.4	92.6	**92.7**	92.6	92.4	92.5	0.1
fr	93.1	92.9	93	**93.4**	93.2	93.1	93.2	0.5
gl	**86.9**	86.3	86.1	86.7	86.4	86.3	86.4	0.8
it	93	92.6	92.7	92.3	93	92.8	**93.1**	0.8
pt	**93**	92.8	92.7	92.8	92.6	92.9	92.8	0.4
ro	88.8	88.9	88.9	**89**	88.7	88.4	88.7	0.6
es	90.9	90.8	90.5	**91.1**	91	90.6	90.7	0.4
Avg.								0.56
# samples	13124	14554	2277	12839	8332	8044	14188	

Table 4: Germanic languages joint training result.

Acc. (%)	da	nl	en	de	sv	max improvement
da	85	85.1	85	85.2	**85.6**	0.6
nl	89	88.8	88.7	**89.3**	88.6	0.5
en	**89.4**	89.1	88.9	88.9	89.1	0.5
de	88.5	**88.9**	88.7	88.6	88.5	0.3
sv	85.7	85	**86.5**	85.9	85	1.5
Avg.						0.68
# samples	4384	12331	12544	14119	4303	

a more sophisticated model (e.g., Neural Architecture Search (Zoph and Le, 2016)) for joint training on multiple languages.

	Baseline			Fudan			Improvement		
Language code	**LAS**	**MLAS**	**BLEX**	**LAS**	**MLAS**	**BLEX**	**LAS**	**MLAS**	**BLEX**
af_afribooms	77.88%	64.48%	66.60%	80.02%	67.34%	66.04%	2.14%	2.86%	-0.56%
grc_perseus	57.75%	31.05%	38.74%	63.31%	34.58%	38.22%	5.56%	3.53%	-0.52%
grc_proiel	67.57%	49.51%	55.85%	69.54%	51.35%	53.03%	1.97%	1.84%	-2.82%
ar_padt	66.41%	55.01%	57.60%	67.33%	55.88%	58.63%	0.92%	0.87%	1.03%
hy_armtdp	21.79%	5.00%	11.94%	26.24%	10.00%	13.85%	4.45%	5.00%	1.91%
eu_bdt	70.13%	57.65%	63.50%	72.74%	58.98%	57.41%	2.61%	1.33%	-6.09%
br_keb	10.25%	0.00%	0.00%	10.25%	0.00%	0.00%	0.00%	0.00%	0.00%
bg_btb	84.91%	75.30%	73.78%	86.47%	77.04%	77.70%	1.56%	1.74%	3.92%
bxr_bdt	12.61%	0.00%	5.00%	12.61%	0.00%	5.00%	0.00%	0.00%	0.00%
ca_ancora	85.61%	76.74%	77.27%	88.37%	80.17%	66.01%	2.76%	3.43%	-11.26%
hr_set	78.61%	58.72%	70.26%	81.01%	60.92%	66.61%	2.40%	2.20%	-3.65%
cs_cac	83.72%	70.89%	77.65%	86.28%	73.90%	79.54%	2.56%	3.01%	1.89%
cs_fictree	82.49%	69.26%	74.96%	85.22%	72.01%	77.18%	2.73%	2.75%	2.22%
cs_pdt	83.94%	74.32%	79.39%	85.35%	75.89%	74.68%	1.41%	1.57%	-4.71%
cs_pud	80.08%	66.53%	73.79%	81.05%	67.97%	68.99%	0.97%	1.44%	-4.80%
da_ddt	75.43%	65.41%	66.04%	78.38%	68.20%	62.51%	2.95%	2.79%	-3.53%
nl_alpino	77.60%	61.55%	64.76%	79.02%	63.89%	60.53%	1.42%	2.34%	-4.23%
nl_lassysmall	74.56%	61.85%	63.14%	77.41%	64.64%	49.81%	2.85%	2.79%	-13.33%
en_ewt	77.56%	68.70%	71.02%	78.44%	68.99%	62.92%	0.88%	0.29%	-8.10%
en_gum	74.20%	62.66%	62.14%	75.29%	63.36%	57.45%	1.09%	0.70%	-4.69%
en_lines	73.10%	64.03%	65.42%	74.83%	65.78%	62.80%	1.73%	1.75%	-2.62%
en_pud	79.56%	67.59%	71.14%	78.80%	67.20%	64.26%	-0.76%	-0.39%	-6.88%
et_edt	75.02%	67.12%	63.85%	77.80%	69.82%	61.53%	2.78%	2.70%	-2.32%
fo_oft	25.19%	0.00%	5.00%	26.95%	0.00%	5.00%	1.76%	0.00%	0.00%
fi_ftb	75.64%	65.22%	61.76%	78.27%	68.03%	66.99%	2.63%	2.81%	5.23%
fi_pud	80.15%	73.16%	65.46%	80.15%	73.16%	65.46%	0.00%	0.00%	0.00%
fi_tdt	76.45%	68.58%	62.19%	79.18%	70.74%	59.79%	2.73%	2.16%	-2.40%
fr_gsd	81.05%	72.16%	74.22%	83.19%	74.01%	68.58%	2.14%	1.85%	-5.64%
fr_sequoia	81.12%	71.34%	74.41%	83.39%	73.59%	69.28%	2.27%	2.25%	-5.13%
fr_spoken	65.56%	53.46%	54.67%	65.63%	52.96%	52.82%	0.07%	-0.50%	-1.85%
gl_ctg	76.10%	62.11%	65.29%	80.38%	67.42%	71.64%	4.28%	5.31%	6.35%
gl_treegal	66.16%	49.13%	51.60%	68.08%	50.06%	52.80%	1.92%	0.93%	1.20%
de_gsd	70.85%	34.09%	60.56%	71.88%	35.12%	34.30%	1.03%	1.03%	-26.26%
got_proiel	62.16%	48.57%	55.02%	65.49%	51.72%	54.63%	3.33%	3.15%	-0.39%
el_gdt	82.11%	65.33%	68.67%	82.56%	65.58%	64.68%	0.45%	0.25%	-3.99%
he_htb	57.86%	44.09%	46.51%	58.87%	44.89%	47.37%	1.01%	0.80%	0.86%
hi_hdtb	87.15%	69.09%	79.93%	88.43%	70.48%	81.52%	1.28%	1.39%	1.59%
hu_szeged	66.76%	52.82%	56.92%	68.74%	54.66%	53.52%	1.98%	1.84%	-3.40%
zh_gsd	57.91%	48.49%	52.92%	60.13%	49.17%	54.29%	2.22%	0.68%	1.37%
id_gsd	74.37%	63.42%	62.50%	75.51%	63.54%	71.50%	1.14%	0.12%	9.00%
ga_idt	62.93%	37.66%	42.06%	64.87%	39.22%	42.44%	1.94%	1.56%	0.38%
it_isdt	86.26%	77.06%	77.12%	88.28%	79.48%	72.47%	2.02%	2.42%	-4.65%
it_postwita	66.81%	53.64%	53.99%	67.58%	53.93%	44.53%	0.77%	0.29%	-9.46%
ja_gsd	72.32%	58.35%	60.17%	73.16%	59.39%	60.92%	0.84%	1.04%	0.75%
ja_modern	22.71%	10.00%	10.00%	22.71%	10.00%	10.00%	0.00%	0.00%	0.00%
kk_ktb	24.21%	10.00%	10.00%	24.21%	10.00%	10.00%	0.00%	0.00%	0.00%
ko_gsd	61.40%	54.10%	50.50%	74.94%	68.34%	62.21%	13.54%	14.24%	11.71%
ko_kaist	70.25%	61.49%	57.68%	82.74%	75.55%	69.47%	12.49%	14.06%	11.79%
kmr_mg	23.92%	5.00%	11.86%	23.92%	5.00%	11.86%	0.00%	0.00%	0.00%
la_ittb	75.95%	66.08%	71.87%	80.07%	71.95%	76.29%	4.12%	5.87%	4.42%
la_perseus	47.61%	30.16%	32.19%	49.99%	31.35%	33.75%	2.38%	1.19%	1.56%
la_proiel	59.66%	47.05%	53.65%	63.93%	51.19%	54.64%	4.27%	4.14%	0.99%
lv_lvtb	69.43%	54.96%	58.25%	70.89%	56.14%	57.30%	1.46%	1.18%	-0.95%
pcm_nsc	12.18%	5.00%	10.87%	10.00%	5.00%	5.00%	-2.18%	0.00%	-5.87%
sme_giella	56.98%	46.05%	42.35%	61.58%	49.88%	44.19%	4.60%	3.83%	1.84%
no_bokmaal	83.47%	74.65%	76.32%	85.29%	76.97%	70.82%	1.82%	2.32%	-5.50%
no_nynorsk	82.13%	72.40%	74.22%	84.09%	74.71%	69.97%	1.96%	2.31%	-4.25%
no_nynorsklia	48.95%	37.60%	40.69%	52.84%	40.67%	43.70%	3.89%	3.07%	3.01%
fro_srcmf	79.27%	70.70%	74.45%	82.70%	75.06%	78.96%	3.43%	4.36%	4.51%
cu_proiel	65.46%	53.96%	58.39%	70.03%	58.51%	63.28%	4.57%	4.55%	4.89%
fa_seraji	79.10%	72.20%	69.43%	79.57%	71.96%	69.42%	0.47%	-0.24%	-0.01%
pl_lfg	87.53%	74.54%	78.58%	88.78%	75.92%	77.55%	1.25%	1.38%	-1.03%
pl_sz	81.90%	63.84%	71.98%	83.54%	65.25%	72.25%	1.64%	1.41%	0.27%
pt_bosque	82.07%	67.40%	72.04%	84.59%	70.21%	62.91%	2.52%	2.81%	-9.13%
ro_rrt	80.27%	71.48%	71.87%	82.67%	74.11%	71.11%	2.40%	2.63%	-0.76%
ru_syntagrus	84.59%	76.87%	78.01%	87.70%	79.58%	82.35%	3.11%	2.71%	4.34%
ru_taiga	55.51%	36.79%	39.79%	57.94%	38.59%	42.12%	2.43%	1.80%	2.33%
sr_set	82.07%	70.04%	74.12%	83.54%	70.86%	66.69%	1.47%	0.82%	-7.43%
sk_snk	75.41%	54.38%	60.35%	78.45%	56.57%	67.75%	3.04%	2.19%	7.40%
sl_ssj	77.33%	63.47%	68.93%	79.15%	65.05%	69.10%	1.82%	1.58%	0.17%
sl_sst	46.95%	34.19%	38.73%	46.19%	33.61%	38.00%	-0.76%	-0.58%	-0.73%
es_ancora	84.43%	76.01%	76.43%	87.66%	80.08%	68.03%	3.23%	4.07%	-8.40%
sv_lines	74.06%	58.62%	66.39%	75.87%	59.96%	64.81%	1.81%	1.34%	-1.58%
sv_pud	70.63%	43.38%	54.47%	72.26%	44.27%	52.52%	1.63%	0.89%	-1.95%
sv_talbanken	77.91%	69.22%	70.01%	80.00%	70.66%	71.49%	2.09%	1.44%	1.48%
th_pud	0.70%	0.03%	0.42%	0.70%	0.03%	0.42%	0.00%	0.00%	0.00%
tr_imst	54.04%	44.50%	45.91%	57.57%	46.59%	46.27%	3.53%	2.09%	0.36%
uk_iu	74.91%	56.78%	63.72%	76.27%	57.66%	63.11%	1.36%	0.88%	-0.61%
hsb_ufal	23.64%	5.00%	11.72%	29.92%	10.00%	15.16%	6.28%	5.00%	3.44%
ur_udtb	77.29%	50.31%	63.74%	77.91%	50.78%	64.30%	0.62%	0.47%	0.56%
ug_udt	56.26%	36.82%	43.53%	55.88%	35.84%	43.16%	-0.38%	-0.98%	-0.37%
vi_vtb	39.63%	33.49%	35.72%	39.53%	32.33%	32.07%	-0.10%	-1.16%	-3.65%

Table 5: All results on test set.

	Baseline			Fudan			Improvement		
Language code	LAS	MLAS	BLEX	LAS	MLAS	BLEX	LAS	MLAS	BLEX
el_gdt	81.37%	63.92%	65.21%	82.31%	64.62%	61.66%	0.94%	0.70%	-3.55%
tr_imst	54.83%	44.25%	45.81%	58.65%	46.16%	45.81%	3.82%	1.91%	0.00%
id_gsd	74.40%	63.51%	63.29%	74.82%	63.16%	71.21%	0.42%	-0.35%	7.92%
da_ddt	75.16%	65.29%	66.07%	78.20%	68.34%	63.80%	3.04%	3.05%	-2.27%
et_edt	76.50%	68.27%	64.17%	79.86%	71.38%	62.30%	3.36%	3.11%	-1.87%
got_proiel	62.03%	48.16%	54.39%	75.09%	61.20%	63.54%	13.06%	13.04%	9.15%
sl_ssj	77.72%	63.96%	68.97%	83.77%	69.66%	72.37%	6.05%	5.70%	3.40%
en_gum	76.63%	65.57%	67.20%	78.81%	67.61%	62.09%	2.18%	2.04%	-5.11%
cu_proiel	66.12%	54.48%	59.16%	79.39%	67.52%	70.80%	13.27%	13.04%	11.64%
ur_udtb	77.44%	49.91%	63.55%	77.92%	50.79%	64.01%	0.48%	0.88%	0.46%
fro_srcmf	79.15%	70.43%	74.27%	81.90%	74.01%	77.87%	2.75%	3.58%	3.60%
hi_hdtb	87.26%	69.78%	80.59%	88.55%	71.17%	82.16%	1.29%	1.39%	1.57%
ko_gsd	57.25%	49.06%	44.24%	72.41%	65.18%	57.04%	15.16%	16.12%	12.80%
cs_fictree	83.16%	70.72%	75.80%	85.99%	73.49%	77.82%	2.83%	2.77%	2.02%
gl_ctg	76.32%	62.58%	65.57%	81.75%	68.93%	73.42%	5.43%	6.35%	7.85%
lv_lvtb	70.67%	57.79%	60.96%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
fr_gsd	85.81%	77.80%	79.16%	88.83%	81.24%	70.55%	3.02%	3.44%	-8.61%
ru_syntagrus	83.87%	75.78%	77.27%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
hu_szeged	68.41%	56.47%	60.17%	70.67%	58.33%	57.56%	2.26%	1.86%	-2.61%
sv_lines	76.23%	62.16%	67.63%	77.91%	63.39%	65.60%	1.68%	1.23%	-2.03%
no_bokmaal	84.56%	75.95%	78.04%	86.54%	78.64%	72.31%	1.98%	2.69%	-5.73%
sv_talbanken	75.39%	66.87%	68.25%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
es_ancora	85.08%	76.81%	77.48%	88.21%	80.75%	68.74%	3.13%	3.94%	-8.74%
he_htb	61.95%	49.28%	51.45%	79.89%	65.68%	67.50%	17.94%	16.40%	16.05%
uk_iu	77.94%	59.66%	68.07%	78.75%	60.11%	66.23%	0.81%	0.45%	-1.84%
grc_proiel	69.13%	52.42%	57.92%	77.67%	61.50%	59.95%	8.54%	9.08%	2.03%
eu_bdt	70.06%	57.46%	63.39%	72.94%	59.11%	56.95%	2.88%	1.65%	-6.44%
fi_ftb	75.76%	65.72%	62.68%	79.90%	70.61%	69.82%	4.14%	4.89%	7.14%
cs_pdt	84.85%	75.35%	80.55%	86.83%	77.45%	75.86%	1.98%	2.10%	-4.69%
sk_snk	75.73%	54.34%	59.71%	80.35%	57.64%	69.80%	4.62%	3.30%	10.09%
hr_set	77.84%	59.60%	69.99%	80.63%	61.89%	66.75%	2.79%	2.29%	-3.24%
no_nynorsk	82.75%	73.88%	75.76%	85.07%	76.64%	72.06%	2.32%	2.76%	-3.70%
grc_perseus	57.89%	30.80%	40.49%	63.21%	34.03%	39.99%	5.32%	3.23%	-0.50%
fr_spoken	65.09%	54.00%	55.42%	73.46%	64.24%	63.19%	8.37%	10.24%	7.77%
pl_sz	82.65%	63.92%	72.59%	84.02%	65.11%	73.04%	1.37%	1.19%	0.45%
fi_tdt	76.39%	68.60%	62.33%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
ca_ancora	85.63%	77.04%	77.56%	88.30%	80.31%	67.64%	2.67%	3.27%	-9.92%
ar_padt	66.81%	55.67%	57.90%	76.11%	63.98%	65.94%	9.30%	8.31%	8.04%
sr_set	82.12%	69.12%	73.06%	84.49%	70.73%	67.66%	2.37%	1.61%	-5.40%
bg_btb	84.67%	74.54%	73.78%	86.73%	76.96%	78.24%	2.06%	2.42%	4.46%
vi_vtb	43.65%	37.39%	39.18%	57.34%	49.57%	47.08%	13.69%	12.18%	7.90%
de_gsd	75.55%	38.52%	65.39%	77.00%	39.91%	40.77%	1.45%	1.39%	-24.62%
fr_seguoia	82.72%	74.13%	76.34%	85.71%	76.53%	71.16%	2.99%	2.40%	-5.18%
cs_cac	84.42%	72.17%	78.29%	86.46%	74.65%	77.70%	2.04%	2.48%	-0.59%
pl_lfg	88.79%	75.15%	79.18%	89.98%	76.88%	79.01%	1.19%	1.73%	-0.17%
en_lines	75.78%	66.29%	68.57%	78.17%	67.59%	67.22%	2.39%	1.30%	-1.35%
zh_gsd	57.39%	48.19%	52.84%	70.09%	58.37%	64.74%	12.70%	10.18%	11.90%
it_postwita	65.85%	52.14%	52.90%	77.23%	66.00%	53.42%	11.38%	13.86%	0.52%
la_proiel	61.33%	48.40%	55.10%	74.41%	61.54%	64.39%	13.08%	13.14%	9.29%
fa_seraji	79.78%	73.03%	73.35%	80.41%	72.61%	73.63%	0.63%	-0.42%	0.28%
af_afribooms	80.19%	65.98%	70.40%	80.95%	67.26%	68.10%	0.76%	1.28%	-2.30%
ko_kaist	71.00%	63.32%	59.19%	83.17%	77.37%	71.51%	12.17%	14.05%	12.32%
la_ittb	73.23%	59.94%	67.43%	78.06%	66.21%	72.02%	4.83%	6.27%	4.59%
en_ewt	77.62%	68.58%	70.98%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
ug_udt	56.88%	37.43%	43.34%	57.78%	37.54%	44.08%	0.90%	0.11%	0.74%
pt_bosque	84.93%	73.22%	76.02%	88.19%	76.65%	66.58%	3.26%	3.43%	-9.44%
ro_rrt	80.32%	71.21%	71.82%	83.42%	74.27%	71.73%	3.10%	3.06%	-0.09%
nl_lassysmall	73.61%	59.99%	61.71%	77.63%	64.44%	52.55%	4.02%	4.45%	-9.16%
it_isdt	85.95%	77.20%	77.37%	87.81%	79.30%	71.82%	1.86%	2.10%	-5.55%
nl_alpino	80.21%	67.14%	69.77%	81.66%	69.04%	58.68%	1.45%	1.90%	-11.09%
ja_gsd	75.48%	62.39%	64.58%	92.51%	83.13%	85.11%	17.03%	20.74%	20.53%

Table 6: All results on development set.

References

Mathieu Dehouck and Pascal Denis. 2017. Delexicalized word embeddings for cross-lingual dependency parsing. In *EACL*. hal.inria.fr, volume 1, pages 241–250.

Timothy Dozat, Peng Qi, and Christopher D Manning. 2017. Stanford's graph-based neural dependency parser at the conll 2017 shared task. *Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies* pages 20–30.

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

Yarin Gal and Zoubin Ghahramani. 2016. A theoretically grounded application of dropout in recurrent neural networks. In *Advances in neural information processing systems*. pages 1019–1027.

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)*. volume 1, pages 1234–1244.

Joakim Nivre et al. 2018. Universal Dependencies 2.2. LINDAT/CLARIN digital library at the Institute of Formal and Applied Linguistics, Charles University, Prague, `http://hdl.handle.net/11234/1-1983xxx`. http://hdl.handle.net/11234/1-1983xxx.

Kenji Sagae and Alon Lavie. 2006. Parser combination by reparsing. In *Proceedings of the Human Language Technology Conference of the NAACL, Companion Volume: Short Papers*. Association for Computational Linguistics, pages 129–132.

Motoki Sato, Hitoshi Manabe, Hiroshi Noji, and Yuji Matsumoto. 2017. Adversarial training for cross-domain universal dependency parsing. In *Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*. Association for Computational Linguistics, pages 71–79. https://doi.org/10.18653/v1/K17-3007.

Tianze Shi, Felix G. Wu, Xilun Chen, and Yao Cheng. 2017. Combining global models for parsing universal dependencies. In *Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*. Association for Computational Linguistics, pages 31–39. https://doi.org/10.18653/v1/K17-3003.

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.

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.

Clara Vania, Xingxing Zhang, and Adam Lopez. 2017. Uparse: the edinburgh system for the conll 2017 ud shared task. In *Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*. Association for Computational Linguistics, pages 100–110. https://doi.org/10.18653/v1/K17-3010.

Dan Zeman et al. 2018a. Universal Dependencies 2.2 CoNLL 2018 shared task development and test data. LINDAT/CLARIN digital library at the Institute of Formal and Applied Linguistics, Charles University, Prague, `http://hdl.handle.net/11234/1-2184`. http://hdl.handle.net/11234/1-2184.

Daniel Zeman, Filip Ginter, Jan Hajič, Joakim Nivre, Martin Popel, Milan Straka, and et al. 2018b. 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*. Association for Computational Linguistics, pages 1–20.

Barret Zoph and Quoc V Le. 2016. Neural architecture search with reinforcement learning. *arXiv preprint arXiv:1611.01578* .