CUNI x-ling: Parsing under-resourced languages in CoNLL 2018 UD Shared Task

Rudolf Rosa and **David Mareček**
Charles University, Faculty of Mathematics and Physics
Institute of Formal and Applied Linguistics
Malostranské náměstí 25, 118 00 Praha, Czechia
{rosa,marecek}@ufal.mff.cuni.cz

Abstract

This is a system description paper for the *CUNI x-ling* submission to the *CoNLL 2018 UD Shared Task*. We focused on parsing under-resourced languages, with no or little training data available. We employed a wide range of approaches, including simple word-based treebank translation, combination of delexicalized parsers, and exploitation of available morphological dictionaries, with a dedicated setup tailored to each of the languages. In the official evaluation, our submission was identified as the clear winner of the Low-resource languages category.

1 Introduction

This paper describes our submission to the CoNLL 2018 shared task on Multilingual Parsing from Raw Text to Universal Dependencies (Zeman et al., 2018; Nivre et al., 2016).

Our primary focus was on the 4 languages with no annotated training data (treebanks) available, as we have significant experience with such a setting (Mareček, 2016; Rosa et al., 2017; Rosa, 2018a); in the shared task, these are Naija, Faroese, Thai, and Breton. Apart from Naija, there are at least some non-treebank resources available for each of the languages, such as parallel data, monolingual data, or morphological dictionaries.[1] Furthermore, we also employ treebanks for other languages together with several cross-lingual parsing methods; in our work, we will refer to the language being parsed as the *target language*, and the other languages that we exploit when parsing it as the *source languages*. We used a different setup for each of the languages, based on its characteristics and on the available resources.

Language	Sentences	Tokens
Buryat	19	153
Kurmanji	20	242
Upper Sorbian	23	460
Kazakh	31	529
Armenian	50	804

Table 1: Sizes of available training data.

Our secondary focus was on the 5 languages with only tiny training data available – see Table 1. However, as we had no previous experience with this particular setup, we tried to build upon our successful approaches for languages with no training data, combining the resources available for the target language with treebanks for different (but preferably close) source languages.

In the official evaluation of the shared task, our submission achieved the highest average scores in all of the main evaluation metrics when averaged over the 9 low-resource languages. We scored particularly well for the languages with no training data available, winning for 3 of them. For the languages with small training data available, our submission was usually not the highest scoring one, but still performed very competitively.

2 Approach

Our baseline approach for parsing a target language is to train the UDPipe tokenizer, tagger and parser (Straka et al., 2016) on UD 2.2 training data for the target language (Nivre et al., 2018), using the default settings. For target languages with no treebank, we need to use training data for another language and cross-lingual techniques. For target languages with small training data, we also use

[1] Parallel data actually exist for all of the languages, at least in the form of the New Testament part of the Bible and the Universal Declaration of Human Rights; however, using these datasets was not allowed in the shared task.

cross-lingual techniques, as an enrichment of the baseline approach to achieve better performance.

In this section, we introduce several approaches that we apply to many or most of the target languages; the specific setups used for each of the target languages are described in later sections.

2.1 Treebank translation using parallel data

Tiedemann (2014) introduced the approach of automatically translating the word forms in a source treebank into the target language, and then training a pseudo-target parser (and/or a tagger) on the resulting pseudo-target treebank.

This approach was further investigated by Rosa et al. (2017), Rosa and Žabokrtský (2017) and Rosa (2018a), finding that the sophistication of the Machine Translation (MT) system plays a rather minor role in cross-lingual parsing, while there is a significant benefit in using word-based translation – this forces the translations to be more literal, and enables a trivial approach to annotation transfer.

In this work, we use probably the simplest possible approach, based on extracting a dictionary from word-aligned data, and translating each source word into the target word most frequently aligned to it, ignoring any context or other information. While we had found that using state-of-the-art statistical MT tools leads to slightly better results, it is also much more computationally demanding, which may be a bottleneck when one needs to process a lot of language pairs in a short time.

Our treebank translation pipeline is:

1. obtain *OpenSubtitles2018*[2] (Lison and Tiedemann, 2016) sentence-aligned source-target parallel data from Opus[3] (Tiedemann, 2012)
2. tokenize the parallel data with source and target UDPipe tokenizers
3. obtain intersection word-alignment with FastAlign[4] (Dyer et al., 2013)
4. extract the translation table: for each source word, take the target word most frequently aligned to it, and store it as its translation
5. translate the source training treebank into the target language, replacing each word form and each lemma[5] by its translation from the translation table (keep the word untranslated if it does not appear in the translation table)
6. now UDPipe can be trained in a standard way on the resulting pseudo-target treebank and applied to target texts

2.2 UniMorph morphology post-corrections

One of the available resources is UniMorph (Sylak-Glassman, 2016),[6] a project on universal morphology annotation that covers a majority of the low-resource languages in this shared task. It provides a list of words associated with lemmas and morphological features. The annotation of features is unfortunately different from that used in Universal Dependencies, however, almost all the features can be mapped to them. The data available for low-resource languages is as follows:

- large data (10,000 words): Armenian, Breton, Faroese, and Kurmanji
- small data (257 words): Kazakh
- no data: Buryat, Naija, Thai, and Upper Sorbian; for Upper Sorbian, we use the large data for the similar Lower Sorbian

The POS tag of the word can be found also among the features, however, sometimes it does not match the UPOS; e.g. the copula verbs are *AUX* in UD but *V* (verb) in UniMorph.

We use the UniMorph lexicon for correcting the morphological features, lemmas, and tags. If a token is found in the lexicon, we change its tag (unless it is *AUX*), lemma, and morphological features according to the lexicon. Each feature that was mapped from UniMorph style to UD style is added to the features obtained by the tagger. In case it was there but with a different value, the value is changed.

We chose to post-correct the morphological annotation only after parsing. This way, the parser cannot benefit from the potentially better morphological annotation; however, the target parser seems to benefit from being applied to an annotation more similar to what it was trained on.[7]

[2] http://www.opensubtitles.org/
[3] http://opus.nlpl.eu/
[4] https://github.com/clab/fast_align

[5] We translate *lemmas* using the dictionary extracted on *forms*, as we typically do not have another choice anyway. We assume that the lemma is a prominent word form and is thus likely to be translated correctly even in this way.
[6] https://unimorph.github.io/
[7] We have not evaluated the influence on delexicalized

2.3 Combining multiple parsers

For cross-lingual parsing of target languages without any training data, McDonald et al. (2013) showed that combining syntactic information from multiple source languages can lead to a more accurate parsing than when using only one source language. Moreover, this idea can be easily extended to target languages with small training data, combining the target language resources with larger resources for other close languages (Zhang and Barzilay, 2015).

To combine the multilingual resources, we use the weighted parse tree combination method of Rosa and Žabokrtský (2015), which is based on the work of Sagae and Lavie (2006). It consists of training separate parsers on the source language treebanks (and also the target language treebank if it is available), applying them independently to the input sentence, and then combining the resulting dependency trees into a directed graph, with each edge weighted by a sum of weights of the parsers which produced this edge. The final parse tree is then obtained by applying the directed maximum spanning tree algorithm of Chu and Liu (1965) and Edmonds (1967) to the weighted graph.

To make the source parser applicable to the target language sentences, we either use a translation approach (translating the source training treebank into the target language, or translating the target input data into the source language), or we train a delexicalized parser, which only uses the part-of-speech as its input, disregarding the word forms.

The parsers need to be weighted according to their expected performance on the target language data. Moreover, for efficiency reasons, we want to only select a few most promising source languages to use; we usually combine only 3 sources.[8] For target languages with small training data available, we simply evaluate the source parsers on the target treebank, select the ones that perform best, and weight them according to their LAS;[9] the target parser is weighted by a hand-crafted weight slightly above the highest-scoring source parser.[10]

For languages with no treebank data available, we used the typological similarity score of Agić (2017) computed on the WALS dataset (Dryer and Haspelmath, 2013).

We use a similar approach to combine multiple predictors for the dependency relation label, part of speech tag, morphological features, and morphological lemma. However, as opposed to dependency trees, there are no strict structural constraints, which means that instead of the spanning tree algorithm, we can use a simple weighted voting.

2.4 Using pre-trained word embeddings

The UDPipe parser uses vector representations of input word forms, which it by default trains jointly with training the parser, i.e. using only the words that appear in the training treebank. However, the parsing accuracy can typically be improved by pre-training the word embeddings on larger monolingual data, and using these fixed embeddings in the parser instead.[11] For low-resource languages, this becomes even more promising, as the training treebanks are tiny or non-existent, and pre-training the word embeddings on much larger data can both improve performance on words unseen in the training data (which are most words) as well as indirectly provide the parser with some more knowledge of the structure of the target language (Rosa et al., 2017). This is especially useful when using translation approaches, where the parser is not actually exposed to genuine target texts during training; in such cases, the word embeddings bring in such exposure at least indirectly.

We use the word embeddings of Bojanowski et al. (2016) pre-trained on Wikipedia texts, which are available online[12] for nearly all of our focus languages (with the only exception of Naija).

3 Languages with low training data

For all the languages with some small training data available, we use this data for training a base UDPipe model, and combine it with additional models trained on data for other close source languages.

source parsers. However, the delexicalized parsers do not use morphological features, and UniMorph-based post-processing does not seem to change the UPOS tags very often, so we do not expect a strong influence.

[8] Note that combining only two parsers does not make much sense due to the combination/voting mechanism.

[9] Labelled Attachment Score; see Section 5.

[10] We want to enable combining the information from the parsers, but we also want to give most power to the target parser. Therefore, we manually choose a target weight higher than the highest source weight, but lower than the sum of the two lowest source weights.

[11] https://ufal.mff.cuni.cz/udpipe/users-manual#udpipe_training_parser_embeddings

[12] https://github.com/facebookresearch/fastText/blob/master/pretrained-vectors.md

Target weight	Additional sources' weights	
0.57 Armenian	0.56 Latvian	0.51 Estonian
0.45 Buryat	0.41 Hindi	0.38 Uyghur
0.44 Kazakh	0.33 Turkish	0.29 Uyghur
0.52 Kurmanji	0.47 Latin	0.45 Greek

Table 2: Weights of the target parser and additional delexicalized source parsers used in parser combination. Weights are based on LAS achieved by the parsers on the target training treebank.

3.1 Armenian, Buryat, Kazakh, Kurmanji

Our setup is identical for four of the target languages – we train UDPipe on the target training data, and combine it with delexicalized parsers for two close source languages, selected and weighted based on the LAS they achieve on the target training treebank:

1. train a UDPipe tokenizer, tagger and parser on the small target training data; use the pretrained word embeddings for the parser
2. train delexicalized parsers for two other close source languages
3. tokenize and tag the input with the target model
4. parse it with the target parser and the delexicalized source language parsers
5. do a weighted combination of the parse trees, using LAS on target treebank as weights
6. post-fix the morphology using data from UniMorph, rewriting UPOS and lemmas and merging morphological features (except for Buryat for which UniMorph is not available)

In Table 2, we list the additional source languages and the weights used for the parser combination.

3.2 Upper Sorbian

For Upper Sorbian, our setup is a bit more complex, combining the target model with source models both for tagging and parsing:

1. apply Polish tokenizer
2. combine Upper Sorbian tagger with Polish tagger and pseudo-Upper Sorbian tagger trained on MonoTranslated Czech treebank
3. apply UniMorph morphology post-correction based on Lower Sorbian UniMorph data
4. combine Upper Sorbian parser with delexicalized Czech, Croatian, Russian, Slovak and Slovene parsers

Predicting lemmas:

0.40 U.Sorb.	0.60 Polish	0.51 Czech

Predicting UPOS tags:

1.00 U.Sorb.	0.69 Polish	0.65 Czech

Predicting morphological features:

0.30 U.Sorb.	0.10 Polish	0.24 Czech

Parsing:

0.53 U.Sorb.	0.70 Croatian	0.73 Czech
	0.66 Russian	0.69 Slovak
	0.68 Slovene	

Table 3: Weights used for combining the individual UDPipe predictors for the Upper Sorbian target and for the additional source languages.

The combination weights are listed in Table 3.

Upper Sorbian is very similar to Czech and Polish, even lexically, which we tried to exploit by using Czech and Polish treebanks also to train taggers for Upper Sorbian.

We found the similarity with Polish to be sufficient for the Polish tagger to be directly applicable to Upper Sorbian texts without any translation.

For Czech, we decided to try to translate the Czech treebank into a pseudo-Upper Sorbian treebank, as the orthographic variance is higher for this language pair. However, as there are no parallel data available, we resorted to an approximate translation approach. First, we preprocessed the Czech treebank by changing '-v-' to '-w-' and 'o-' to 'wo-', as this seems to be a regular difference between Czech and Upper Sorbian. We then applied the MonoTrans system (Rosa, 2017), which tries to map the source words onto similar target words; the similarity is computed based on an edit distance of the word forms, and on their frequencies in monolingual corpora.[13]

Moreover, evaluation of the Upper Sorbian tools on the Upper Sorbian training data indicated a very low performance of the tools (even without cross-validation); therefore, we often give more power to the cross-lingual tools than to the Upper Sorbian

[13] We used the Czech treebank and the Upper Sorbian texts from Wikipedia (Rosa, 2018b) as monolingual corpora.

tools in the combinations. The weights we use are the accuracies of the tools on the Upper Sorbian training data (LAS for parsing, UPOS accuracy for UPOS, etc.); this time, the target weights are obtained in the same way as the source weights, i.e. they are not hand-crafted.

4 Languages with no training data

We list the processing pipeline for each of the languages, together with detailed descriptions of processing steps specific for the language.

4.1 Naija

1. apply English tokenizer
2. "translate" words to English
3. apply English tagger and parser
4. copy lowercased form to lemma, remove final '-s' if there is one

For Naija (Nigerian Pidgin) we have no data available at all. Some basic information about this language can be found on Wikipedia,[14] where there are also links to other resources. A couple of sentences can be found also on the web of Jehovah Witnesses[15] and we also looked into the translation of the Declaration of Human Rights.[16] Based on these resources, we conclude that Naija is very similar to English, but differs mainly in the most common words and function words. We also learned that its written form is not standardized, since different resources showed different level of similarity with English spelling. While the texts of Jehovah Witnesses were almost English (*We come from different different place and we dey speak different different language.*), the example from the web of the University of Hawai[17] shows more differences:

Naija: *A bai shu giv mai broda.*
English: *I bought shoes that I gave to my brother.*

Since we did not know what spelling is used in the testing treebank, we decided to use the tools trained on English, applied to Naija inputs processed with a couple of translation rules which we devised based on the Naija texts which we read.

[14] https://en.wikipedia.org/wiki/Nigerian_Pidgin
[15] https://www.jw.org/pcm/
[16] https://unicode.org/udhr/translations.html
[17] http://www.hawaii.edu/satocenter/langnet/definitions/naija.html

First, some Naija words are directly translated into English using the following small dictionary:

sey	→	that	de	→	is
na	→	is	don	→	has
wey	→	which	am	→	him
im	→	his	go	→	will
wetin	→	what	no	→	not
dey	→	is	di	→	the
deh	→	is	pikin	→	small
foh	→	in	sebi	→	right
e	→	he	abi	→	right
dem	→	they	nna	→	man
dis	→	this	sabi	→	know

It seems that Naija language is very simple and many words are homonymous when translating into English. It is of course possible that not all translations are correct, since the dictionary was developed mainly by choosing the most probable English word based on the example context.

Second, we used a couple of regular expressions to translate remaining non-English words.[18]

i	→	y	k	→	c
d	→	th	^	→	h
t	→	th	$	→	t
a$	→	er	o	→	ou

We perform the above substitutions on each unknown Naija word one after another, until it becomes a known English word. If no English word is reached after all the substitutions are done, the original word is used.

It is evident that any information found about such a highly low-resource language is crucial. We read a couple of web pages with examples of Naija, and based on that we built the small dictionary. If we were limited to read only the English Wikipedia article about the Naija language, the dictionary would be of course smaller and the results would be worse. In Table 4, we show the results when no translation rules are used and Naija is parsed by English parser, and the results when only the information from the Wikipedia article about Naija is used.

4.2 Thai

1. obtain a Thai tokenizer
2. translate Indonesian, Chinese and Vietnamese treebanks into Thai, using *OpenSub-*

[18] For this purposes, we define that a word is English if it has more than five occurrences in the first 3 million words of the English Wikipedia dump.

Naija	LAS	MLAS	BLEX
no translation	16.1	2.7	14.9
using Wikipedia	22.3	2.6	19.4
using all sources	30.1	4.6	26.0

Table 4: Comparison of Naija results with no translation, only with Wikipedia examples, and the full setup which also uses information we learned from other websites.

titles2018 parallel data;[19]

3. train pseudo-Thai taggers and parsers on the translated treebanks; use pre-trained Thai word embeddings for the parsers

4. combine the taggers and parsers (with weights 0.75, 0.55, 0.40 based on LAS of the source parsers on source development data)

The crucial part of Thai analysis is tokenization, since there are no tokenized texts available; however, we need tokenization both in the main processing pipeline, as well as to tokenize the parallel data for the translation step. The only data comprising separated Thai tokens are the word vectors trained on Wikipedia[20] (Bojanowski et al., 2016). The tokens are ordered according to their frequency and are associated with the vectors.

We used a very simple approach. We generated a synthetic Thai text by sampling Thai tokens from the list of tokens available. Since we do not know the token distribution, we decided that the probability $Prob(t)$ of a token is inversely proportional to the square root of its order $Ord(t)$:

$$Prob(t) \propto \frac{1}{\sqrt{Ord(t)}}$$

The lexicon itself contains a lot of foreign words (English, Japanese, Chinese), which caused that approximately every third generated word was not Thai. We therefore filtered out all the tokens containing English, Japanese, or Chinese characters.[21] After a token is sampled, the end of sentence is generated with a probability of 5%.

By this procedure, we generated a text of one million tokens in total. We generated two variants, one with tokens separated by spaces and one without spaces. Using these two files, we trained the UDPipe tokenizer for Thai.[22] We assume that since the tokens were sampled randomly, the only information the tokenizer can learn are the tokens itself and therefore the tokenization of real Thai texts should be reasonable.

The parameters of the sampling procedure could be tuned if we had even a tiny example of tokenized text in Thai.

4.3 Faroese

1. train devowelled Nynorsk tagger and parser

2. apply Nynorsk tokenizer

3. apply devowelled Nynorsk tagger and parser

4. copy lowercased form to lemma

5. apply UniMorph morphology post-correction

Faroese is quite close to the Nynorsk variant of Norwegian; even applying the Nynorsk models directly to Faroese texts yields competitive results. Unfortunately, there is no parallel data available to perform standard treebank translation.

However, as shown by Rosa et al. (2017), lexically similar languages can be brought even closer by *devowelling* the words, i.e. by removing all vowels, which acts as a sort of a poor man's translation into an intermediary pivot language. We thus devowel the Nynorsk treebank to obtain a devowelled Nynorsk tagger and parser, and apply it to devowelled Faroese texts.

4.4 Breton

1. translate French treebank into Breton, using *OpenSubtitles2018* parallel data

2. train pseudo-Breton tagger and parser on the translated treebank (by mistake, we did not use pre-trained Breton word embeddings)

3. apply French tokenizer

4. apply pseudo-Breton tagger and parser

5. apply UniMorph morphology post-correction

[19] Vietnamese uses a lot of tokens with internal spaces; for the translation, we replaced the spaces with underscores.

[20] https://github.com/facebookresearch/fastText/blob/master/pretrained-vectors.md

[21] For filtering, we used the regular expression `[a-z A-Z \u4E00-\u9FFF \u3040-\u309F \u30A0-\u30FF \u4e00-\u9fff]`

[22] As recommended by the UDPipe manual, we use the `dimension=64` setting.

Submission	LAS	MLAS	BLEX	UPOS
CUNI xling	**27.9**	**6.1**	**14.0**	57.6
Uppsala	25.9	5.2	9.0	**61.1**
TurkuNLP	22.9	3.6	11.4	52.5
Baseline	17.2	3.4	7.6	45.2

Table 5: Macro-average LAS, MLAS, BLEX and UPOS on the 9 low-resource languages. Best result in bold, second-best result underlined.

Breton is a Celtic language; however, we do not have much treebank or parallel data for Celtic languages. Therefore, we decided to only use French as a single source, since due to the long-term contact, Breton is similar to French in some aspects, and there is at least some parallel data available.

5 Evaluation

The evaluation of the submissions to the shared task was performed by the organizers via the TIRA evaluation platform (Potthast et al., 2014), running the submitted systems on secret test data and reporting their performance in LAS (labeled attachment score), MLAS (morphology-aware labeled attachment score), and BLEX (bi-lexical dependency score). For a full description of the metrics, see (Zeman et al., 2018) or the shared task website;[23] here, we only note that while LAS only evaluates parsing accuracy, MLAS also includes evaluation of tagging (UPOS and morphological features), while BLEX also includes lemmatization. We also list UPOS tagging accuracies.

Table 5 shows the average scores over the 9 low-resource languages. Our submission achieved the best average result in all the 3 main scoring metrics; for comparison, we also list the submissions that scored second-best in the metrics, and the baseline setup.

Table 6 reports the results individually for each low-resource language, together with the ranking of our submission among all of the 26 participants.

All scores are adapted from official results.[24]

5.1 Languages with no training data

For the languages with no training data, which were our primary focus, our submission typically scores best in all of the metrics, with the exception of Breton. Our results are particularly strong for Thai, 2x-3x higher than the second best system.

By analyzing our setup for Breton and comparing it to the setups used by other participants of the shared task, we found that we had unfortunately taken several clearly suboptimal steps:

- We overlooked the availability of *Ofis Publik ar Brezhoneg*,[25] a Breton-French parallel corpus of 60,000 sentences, considerably larger and probably cleaner than the 17,000 *OpenSubtitles2018* sentences we used.
- We failed to note the peculiar Breton spelling with a lot of intra-word apostrophes, which calls for an adaptation of the tokenizer.
- We forgot to use the available pre-trained word embeddings.

Another case where our solution performs poorly is MLAS score for Naija, which does not even surpass the baseline. We made the mistake of keeping the morphological features predicted by the English tagger, even though the pidgin language exhibits little or no inflection, and a better approach would thus be not to predict any morphological features at all (i.e. to always return '_'). Indeed, in the now-released test data, no morphological features are annotated in the Naija treebank.

We also did not do well in UPOS tagging for Faroese, probably because of the devowelling.

5.2 Languages with low training data

For the languages with some small training data available, we score a bit worse. Our submission is usually among the top 5 submissions and always above the baseline, but it is rarely the best. Nevertheless, as this setting was only our secondary focus and as we had no prior experience with it, we are still happy about our results.

In general, our submission performs particularly well in MLAS, which is probably thanks to our exploitation of the UniMorph dictionary. For Armenian and Kazakh, we managed to win in MLAS and BLEX, although we are not sure why, as our setup was similar for all of the languages. We note, however, that the available training data are largest for these two languages; as we train UDPipe on the target data with the default settings, not adapted to the small size of the training data in any way, our approach is probably better suited for

[23] http://universaldependencies.org/conll18/evaluation.html

[24] http://universaldependencies.org/conll18/results.html

[25] http://opus.nlpl.eu/OfisPublik.php

Target language	LAS ours		LAS comp.		MLAS ours		MLAS comp.		BLEX ours		BLEX comp.		UPOS ours		UPOS comp.	
Breton	26.9	4	38.6	1	3.0	4	13.9	1	11.4	4	20.7	1	52.7	4	85.0	1
Faroese	**49.4**	**1**	47.2	2	**1.1**	**1**	0.8	2	**14.4**	**1**	14.4	2	58.7	6	65.5	1
Naija	**30.1**	**1**	24.5	2	4.6	6	5.3	1	**26.0**	**1**	22.9	2	**67.9**	**1**	57.2	2
Thai	**13.7**	**1**	6.9	2	**6.3**	**1**	2.2	2	**10.8**	**1**	3.5	2	**39.4**	**1**	33.8	2
Buryat	17.1	5	19.5	1	2.5	2	3.0	1	5.6	4	6.7	1	42.3	7	50.8	1
U.Sorb.	33.4	5	46.4	1	8.5	2	9.1	1	14.6	10	21.1	1	69.9	4	79.5	1
Armen.	30.1	4	37.0	1	**13.4**	**1**	10.4	2	**19.0**	**1**	18.3	2	71.4	2	75.4	1
Kazakh	26.3	2	31.9	1	**8.9**	**1**	8.6	2	**11.3**	**1**	10.2	2	54.6	5	61.7	1
Kurm.	24.0	8	30.4	1	6.9	3	8.0	1	12.6	3	13.7	1	**61.5**	**1**	61.3	2

Table 6: LAS, MLAS, BLEX and UPOS of our submission (*ours*), as well as the best result achieved among the other participants (*comp.*). The ranks are also listed.

Submission	LAS	MLAS	BLEX	UPOS
Best	75.8	61.3	66.1	90.9
Baseline	65.8	52.4	55.8	87.3
CUNI xling	64.9	50.4	54.1	88.7

Table 7: Macro-average LAS, MLAS, BLEX and UPOS on all 82 test sets for 57 languages.

languages with somewhat larger training data. We hypothesize that the training procedure should be modified when the training data are small, e.g. by lowering the number of training iterations over the data, or by reducing the complexity of the model; however, we have not performed any experiments in this direction.

5.3 All languages

For completeness, we also include the macro-average evaluation of our submission on all 82 test sets in Table 7; for all but the 9 low-resourced ones, we simply submitted a standard UDPipe system trained with default parameters.

We usually rank slightly below the official baseline (typically around the 20th position), with a huge loss to the winner. This shows that the parser we use is not very strong in itself, in contrast with most of our competitors' parsers. Nevertheless, by applying various specialized cross-lingual techniques, we managed to surpass even the stronger parsers on the low-resource languages.

6 Conclusion

In this paper, we described our submission to the *CoNLL 2018 UD Shared Task*, in which we focused on under-resourced languages.

We have devised a separate processing pipeline tailored to each low-resource language, based on what resources are available for it and how similar to other resource-rich languages it is. Our approach mostly revolves around simple dictionary-based machine translation, employment of pre-trained word embeddings, combination of delexicalized parsers for close languages, and exploitation of a morphological dictionary.

Our submission achieved the best average result in all the three main evaluation metrics on the low-resource languages. For the languages with no training data, our submission usually outperformed all other submissions. For the languages with small training data, our submission was usually among the top 5 out of all the 26 submissions.

Our approach demonstrates that even quite simple methods can work well, as in the context-independent word-based dictionary-lookup translation. On the other hand, we did not surpass a LAS of 50 for any of the under-resourced languages, only reaching 28 on average. This shows that, even though the various techniques we used can bring huge improvements over the baselines, the resulting parsing accuracies are probably still too low for most practical purposes.

Acknowledgments

This work was supported by the grant 18-02196S of the Grant Agency of the Czech Republic and the grant CZ.02.1.01/0.0/0.0/16_013/0001781 of the Ministry of Education, Youth and Sports of the Czech Republic. This work 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).

References

Željko Agić. 2017. Cross-lingual parser selection for low-resource languages. In *Proceedings of the NoDaLiDa 2017 Workshop on Universal Dependencies (UDW 2017)*. Association for Computational Linguistics, Gothenburg, Sweden, pages 1–10. http://www.aclweb.org/anthology/W17-0401.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2016. Enriching word vectors with subword information. *CoRR* abs/1607.04606. http://arxiv.org/abs/1607.04606.

Yoeng-Jin Chu and Tseng-Hong Liu. 1965. On shortest arborescence of a directed graph. *Scientia Sinica* 14(10):1396.

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

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.

Pierre Lison and Jörg Tiedemann. 2016. Opensubtitles2016: Extracting large parallel corpora from movie and TV subtitles. In *Proceedings of the 10th International Conference on Language Resources and Evaluation (LREC 2016)*. European Language Resources Association.

David Mareček. 2016. Twelve years of unsupervised dependency parsing. In Broňa Brejová, editor, *Proceedings of the 16th ITAT: Slovenskočeský NLP workshop (SloNLP 2016)*. Comenius University in Bratislava, Faculty of Mathematics, Physics and Informatics, CreateSpace Independent Publishing Platform, Bratislava, Slovakia, volume 1649 of *CEUR Workshop Proceedings*, pages 56–62.

Ryan McDonald, Joakim Nivre, Yvonne Quirmbach-Brundage, Yoav Goldberg, Dipanjan Das, Kuzman Ganchev, Keith Hall, Slav Petrov, Hao Zhang, Oscar Täckström, Claudia Bedini, Núria Bertomeu Castelló, and Jungmee Lee. 2013. Universal dependency annotation for multilingual parsing. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*. pages 92–97.

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.

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.

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.

Rudolf Rosa. 2017. Monotrans: Statistical machine translation from monolingual data. In Jaroslava Hlaváčová, editor, *Proceedings of the 17th conference ITAT 2017: Slovenskočeský NLP workshop (SloNLP 2017)*. ÚFAL MFF UK, CreateSpace Independent Publishing Platform, Praha, Czechia, volume 1885 of *CEUR Workshop Proceedings*, pages 201–208.

Rudolf Rosa. 2018a. *Discovering the structure of natural language sentences by semi-supervised methods*. Ph.D. thesis, Charles University, Faculty of Mathematics and Physics, Praha, Czechia.

Rudolf Rosa. 2018b. Plaintext Wikipedia dump 2018. LINDAT/CLARIN digital library at the Institute of Formal and Applied Linguistics (ÚFAL), Faculty of Mathematics and Physics, Charles University. http://hdl.handle.net/11234/1-2735.

Rudolf Rosa and Zdeněk Žabokrtský. 2015. KL_{cpos^3} – a language similarity measure for delexicalized parser transfer. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, Short Papers*. Association for Computational Linguistics, Stroudsburg, PA, USA.

Rudolf Rosa and Zdeněk Žabokrtský. 2017. Error analysis of cross-lingual tagging and parsing. In Jan Hajič, editor, *Proceedings of the 16th International Workshop on Treebanks and Linguistic Theories*. Univerzita Karlova, Univerzita Karlova, Praha, Czechia, pages 106–118.

Rudolf Rosa, Daniel Zeman, David Mareček, and Zdeněk Žabokrtský. 2017. Slavic forest, Norwegian wood. In Preslav Nakov, Marcos Zampieri,

Nikola Ljubešić, Jörg Tiedemann, Shervin Malmasi, and Ahmed Ali, editors, *Proceedings of the Fourth Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial4)*. Association for Computational Linguistics, Association for Computational Linguistics, Stroudsburg, PA, USA, pages 210–219.

Kenji Sagae and Alon Lavie. 2006. Parser combination by reparsing. In *Proceedings of HLT-NAACL*. ACL, pages 129–132.

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.

John Sylak-Glassman. 2016. The composition and use of the universal morphological feature schema (unimorph schema). Technical report, Technical report, Department of Computer Science, Johns Hopkins University.

Jörg Tiedemann. 2012. Parallel data, tools and interfaces in OPUS. In *LREC*. volume 2012, pages 2214–2218.

Jörg Tiedemann. 2014. Rediscovering annotation projection for cross-lingual parser induction. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*. pages 1854–1864.

Daniel Zeman, Jan Hajič, Martin Popel, Martin Potthast, Milan Straka, Filip Ginter, Joakim Nivre, and Slav Petrov. 2018. CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*. Association for Computational Linguistics, Brussels, Belgium, pages 1–20.

Yuan Zhang and Regina Barzilay. 2015. Hierarchical low-rank tensors for multilingual transfer parsing. Association for Computational Linguistics.