CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies

Daniel Zeman[1], **Jan Hajič**[1], **Martin Popel**[1], **Martin Potthast**[2], **Milan Straka**[1], **Filip Ginter**[3], **Joakim Nivre**[4], **and Slav Petrov**[5]

[1]Charles University, Faculty of Mathematics and Physics
[2]Universität Leipzig, [3]University of Turku
[4]Uppsala University, [5]Google AI Language

{zeman|hajic|popel|straka}@ufal.mff.cuni.cz,
martin.potthast@uni-leipzig.de, figint@utu.fi,
joakim.nivre@lingfil.uu.se, slav@google.com

Abstract

Every year, the Conference on Computational Natural Language Learning (CoNLL) features a shared task, in which participants train and test their learning systems on the same data sets. In 2018, one of two tasks was devoted to learning dependency parsers for a large number of languages, in a real-world setting without any gold-standard annotation on the input. All test sets followed the unified annotation scheme of Universal Dependencies (Nivre et al., 2016). This shared task constitutes a 2^{nd} edition—the first one took place in 2017 (Zeman et al., 2017); the main metric from 2017 was kept, allowing for easy comparison, and two new main metrics were introduced. New datasets added to the Universal Dependencies collection between mid-2017 and the spring of 2018 contributed to the increased difficulty of the task this year. In this overview paper, we define the task and the updated evaluation methodology, describe data preparation, report and analyze the main results, and provide a brief categorization of the different approaches of the participating systems.

1 Introduction

The 2017 CoNLL shared task on universal dependency parsing (Zeman et al., 2017) picked up the thread from the influential shared tasks in 2006 and 2007 (Buchholz and Marsi, 2006; Nivre et al., 2007) and evolved it in two ways: (1) the parsing process started from raw text rather than gold standard tokenization and part-of-speech tagging, and (2) the syntactic representations were consistent across languages thanks to the Universal Dependencies framework (Nivre et al., 2016). The 2018 CoNLL shared task on universal dependency parsing starts from the same premises but adds a focus on morphological analysis as well as data from new languages.

Like last year, participating systems minimally had to find labeled syntactic dependencies between words, i.e., a syntactic head for each word, and a label classifying the type of the dependency relation. In addition, this year's task featured new metrics that also scored a system's capacity to predict a morphological analysis of each word, including a part-of-speech tag, morphological features, and a lemma. Regardless of metric, the assumption was that the input should be raw text, with no gold-standard word or sentence segmentation, and no gold-standard morphological annotation. However, for teams who wanted to concentrate on one or more subtasks, segmentation and morphology predicted by the baseline UDPipe system (Straka et al., 2016) was made available just like last year.

There are eight new languages this year: Afrikaans, Armenian, Breton, Faroese, Naija, Old French, Serbian, and Thai; see Section 2 for more details. The two new evaluation metrics are described in Section 3.

2 Data

In general, we wanted the participating systems to be able to use any data that is available free of charge for research and educational purposes (so that follow-up research is not obstructed). We deliberately did not place upper bounds on data sizes (in contrast to e.g. Nivre et al. (2007)), despite the fact that processing large amounts of data may be difficult for some teams. Our primary objective was to determine the capability of current parsers provided with large amounts of freely available data.

In practice, the task was formally closed, i.e., we listed the approved data resources so that all participants were aware of their options. However, the selection was rather broad, ranging from Wikipedia dumps over the OPUS parallel corpora (Tiedemann, 2012) to morphological transducers. Some of the resources were proposed by the participating teams.

We provided dependency-annotated training and test data, and also large quantities of crawled raw texts. Other language resources are available from third-party servers and we only referred to the respective download sites.

2.1 Training Data: UD 2.2

Training and development data came from the Universal Dependencies (UD) 2.2 collection (Nivre et al., 2018). This year, the official UD release immediately followed the test phase of the shared task. The training and development data were available to the participating teams as a pre-release; these treebanks were then released exactly in the state in which they appeared in the task.[1] The participants were instructed to only use the UD data from the package released for the shared task. In theory, they could locate the (yet unreleased) test data in the development repositories on GitHub, but they were trusted that they would not attempt to do so.

82 UD treebanks in 57 languages were included in the shared task;[2] however, nine of the smaller treebanks consisted solely of test data, with no data at all or just a few sentences available for training. 16 languages had two or more treebanks from different sources, often also from different domains.[3] See Table 1 for an overview.

61 treebanks contain designated development data. Participants were asked not to use it for training proper but only for evaluation, development, tuning hyperparameters, doing error analysis etc. Seven treebanks have reasonably-sized training data but no development data; only two of them, Irish and North Sámi, are the sole treebanks of their respective languages. For those treebanks cross-validation had to be used during development, but the entire dataset could be used for training once hyperparameters were determined. Five treebanks consist of extra test sets: they have no training or development data of their own, but large training data exist in other treebanks of the same languages (Czech-PUD, English-PUD, Finnish-PUD, Japanese-Modern and Swedish-PUD, respectively). The remaining nine treebanks are low-resource languages. Their “training data” was either a tiny sample of a few dozen sentences (Armenian, Buryat, Kazakh, Kurmanji, Upper Sorbian), or there was no training data at all (Breton, Faroese, Naija, Thai). Unlike in the 2017 task, these languages were not “surprise languages”, that is, the participants knew well in advance what languages to expect. The last two languages are particularly difficult: Naija is a pidgin spoken in Nigeria; while it can be expected to bear some similarity to English, its spelling is significantly different from standard English, and no resources were available to learn it. Even harder was Thai with a writing system that does not separate words by spaces; the Facebook word vectors were probably the only resource among the approved additional data where participants could learn something about words in Thai (Rosa and Mareček, 2018; Smith et al., 2018). It was also possible to exploit the fact that there is a 1-1 sentence mapping between the Thai test set and the other four PUD test sets.[4]

Participants received the training and development data with gold-standard tokenization, sentence segmentation, POS tags and dependency re-

[1] UD 2.2 also contains other treebanks that were not included in the task for various reasons, and that may have been further developed even during the duration of the task.

[2] Compare with the 81 treebanks and 49 languages in the 2017 task.

[3] We distinguish treebanks of the same language by their short names or acronyms. Hence, the two treebanks of Ancient Greek are identified as Perseus and PROIEL, the three treebanks of Latin are ITTB, Perseus and PROIEL, etc.

[4] While the test datasets were not available to the teams when they developed their systems, the documentation of the treebanks was supplied together with the training data, hence the teams could learn that the PUD treebanks were parallel.

Language	Tbk Code	2017	TrWrds
Afrikaans	af_afribooms	NA	34 K
Ancient Greek	grc_perseus	grc	160 K
Ancient Greek	grc_proiel	grc_proiel	187 K
Arabic	ar_padt	ar	224 K
Armenian	hy_armtdp	NA	1 K
Basque	eu_bdt	eu	73 K
Breton	br_keb	NA	0 K
Bulgarian	bg_btb	bg	124 K
Buryat	bxr_bdt	bxr	0 K
Catalan	ca_ancora	ca	418 K
Chinese	zh_gsd	zh	97 K
Croatian	hr_set	hr	154 K
Czech	cs_cac	cs_cac	473 K
Czech	cs_fictree	NA	134 K
Czech	cs_pdt	cs	1,173 K
Czech	cs_pud	cs_pud	0 K
Danish	da_ddt	da	80 K
Dutch	nl_alpino	nl	186 K
Dutch	nl_lassysmall	nl_lassysmall	75 K
English	en_ewt	en	205 K
English	en_gum	NA	54 K
English	en_lines	en_lines	50 K
English	en_pud	en_pud	0 K
Estonian	et_edt	et	288 K
Faroese	fo_oft	NA	0 K
Finnish	fi_ftb	fi_ftb	128 K
Finnish	fi_pud	fi_pud	0 K
Finnish	fi_tdt	fi	163 K
French	fr_gsd	fr	357 K
French	fr_sequoia	fr_sequoia	51 K
French	fr_spoken	NA	15 K
Galician	gl_ctg	gl	79 K
Galician	gl_treegal	gl_treegal	15 K
German	de_gsd	de	264 K
Gothic	got_proiel	got	35 K
Greek	el_gdt	el	42 K
Hebrew	he_htb	he	138 K
Hindi	hi_hdtb	hi	281 K
Hungarian	hu_szeged	hu	20 K
Indonesian	id_gsd	id	98 K
Irish	ga_idt	ga	14 K

Language	Tbk Code	2017	TrWrds
Italian	it_isdt	it	276 K
Italian	it_postwita	NA	99 K
Japanese	ja_gsd	ja	162 K
Japanese	ja_modern	NA	0 K
Kazakh	kk_ktb	kk	1 K
Korean	ko_gsd	ko	57 K
Korean	ko_kaist	NA	296 K
Kurmanji	kmr_mg	kmr	0 K
Latin	la_ittb	la_ittb	270 K
Latin	la_perseus	la	18 K
Latin	la_proiel	la_proiel	172 K
Latvian	lv_lvtb	lv	81 K
Naija	pcm_nsc	NA	0 K
North Sámi	sme_giella	sme	17 K
Norwegian	no_bokmaal	no_bokmaal	244 K
Norwegian	no_nynorsk	no_nynorsk	245 K
Norwegian	no_nynorsklia	NA	4 K
Old Church Slavonic	cu_proiel	cu	37 K
Old French	fro_srcmf	NA	136 K
Persian	fa_seraji	fa	121 K
Polish	pl_lfg	NA	105 K
Polish	pl_sz	pl	63 K
Portuguese	pt_bosque	pt	207 K
Romanian	ro_rrt	ro	185 K
Russian	ru_syntagrus	ru_syntagrus	872 K
Russian	ru_taiga	NA	10 K
Serbian	sr_set	NA	66 K
Slovak	sk_snk	sk	81 K
Slovenian	sl_ssj	sl	113 K
Slovenian	sl_sst	sl_sst	19 K
Spanish	es_ancora	es_ancora	445 K
Swedish	sv_lines	sv_lines	48 K
Swedish	sv_pud	sv_pud	0 K
Swedish	sv_talbanken	sv	67 K
Thai	th_pud	NA	0 K
Turkish	tr_imst	tr	38 K
Ukrainian	uk_iu	uk	75 K
Upper Sorbian	hsb_ufal	hsb	0 K
Urdu	ur_udtb	ur	109 K
Uyghur	ug_udt	ug	19 K
Vietnamese	vi_vtb	vi	20 K

Table 1: Overview of the 82 test treebanks. **TbkCode** = Treebank identifier, consisting of the ISO 639 language code followed by a treebank-specific code. **2017** = Code of the corresponding treebank in the 2017 task if applicable (“NA” otherwise). **TrWrds** = Size of training data, rounded to the nearest thousand words.

lations; and for most languages also lemmas and morphological features.

Cross-domain and cross-language training was allowed and encouraged. Participants were free to train models on any combination of the training treebanks and apply it to any test set.

2.2 Supporting Data

To enable the induction of custom embeddings and the use of semi-supervised methods in general, the participants were provided with supporting resources primarily consisting of large text corpora for many languages in the task, as well as embeddings pre-trained on these corpora. In total, 5.9 M sentences and 90 G words in 45 languages are available in CoNLL-U format (Ginter et al., 2017); the per-language sizes of the corpus are listed in Table 2.

See Zeman et al. (2017) for more details on how the raw texts and embeddings were processed. Note that the resource was originally prepared for the 2017 task and it was not extended to include the eight new languages; however, some of the new languages are covered by the word vectors provided by Facebook (Bojanowski et al., 2016) and approved for the shared task.

Language	Words
English (en)	9,441 M
German (de)	6,003 M
Portuguese (pt)	5,900 M
Spanish (es)	5,721 M
French (fr)	5,242 M
Polish (pl)	5,208 M
Indonesian (id)	5,205 M
Japanese (ja)	5,179 M
Italian (it)	5,136 M
Vietnamese (vi)	4,066 M
Turkish (tr)	3,477 M
Russian (ru)	3,201 M
Swedish (sv)	2,932 M
Dutch (nl)	2,914 M
Romanian (ro)	2,776 M
Czech (cs)	2,005 M
Hungarian (hu)	1,624 M
Danish (da)	1,564 M
Chinese (zh)	1,530 M
Norwegian-Bokmål (no)	1,305 M
Persian (fa)	1,120 M
Finnish (fi)	1,008 M
Arabic (ar)	963 M
Catalan (ca)	860 M
Slovak (sk)	811 M
Greek (el)	731 M
Hebrew (he)	615 M
Croatian (hr)	583 M
Ukrainian (uk)	538 M
Korean (ko)	527 M
Slovenian (sl)	522 M
Bulgarian (bg)	370 M
Estonian (et)	328 M
Latvian (lv)	276 M
Galician (gl)	262 M
Latin (la)	244 M
Basque (eu)	155 M
Hindi (hi)	91 M
Norwegian-Nynorsk (no)	76 M
Kazakh (kk)	54 M
Urdu (ur)	46 M
Irish (ga)	24 M
Ancient Greek (grc)	7 M
Uyghur (ug)	3 M
Kurdish (kmr)	3 M
Upper Sorbian (hsb)	2 M
Buryat (bxr)	413 K
North Sámi (sme)	331 K
Old Church Slavonic (cu)	28 K
Total	90,669 M

Table 2: Supporting data overview: Number of words (M = million; K = thousand) for each language.

2.3 Test Data: UD 2.2

Each of the 82 treebanks mentioned in Section 2.1 has a test set. Test sets from two different treebanks of one language were evaluated separately as if they were different languages. Every test set contains at least 10,000 words (including punctuation marks). UD 2.2 treebanks that were smaller than 10,000 words were excluded from the shared task. There was no upper limit on the test data; the largest treebank had a test set comprising 170K words. The test sets were officially released as a part of UD 2.2 immediately after the shared task.[5]

3 Evaluation Metrics

There are three main evaluation scores, dubbed **LAS**, **MLAS** and **BLEX.** All three metrics reflect word segmentation and relations between content words. LAS is identical to the main metric of the 2017 task, allowing for easy comparison; the other two metrics include part-of-speech tags, morphological features and lemmas. Participants who wanted to decrease task complexity could concentrate on improvements in just one metric; however, all systems were evaluated with all three metrics, and participants were strongly encouraged to output all relevant annotation, even if they just copy values predicted by the baseline model.

When parsers are applied to raw text, the metric must be adjusted to the possibility that the number of nodes in gold-standard annotation and in the system output vary. Therefore, the evaluation starts with aligning system nodes and gold nodes. A dependency relation cannot be counted as correct if one of the nodes could not be aligned to a gold node. See Section 3.4 and onward for more details on alignment.

The evaluation software is a Python script that computes the three main metrics and a number of additional statistics. It is freely available for download from the shared task website.[6]

3.1 LAS: Labeled Attachment Score

The standard evaluation metric of dependency parsing is the *labeled attachment score* (LAS), i.e., the percentage of nodes with correctly assigned reference to the parent node, including the label (type) of the relation. For scoring purposes, only

[5] http://hdl.handle.net/11234/1-2837

[6] http://universaldependencies.org/conll18/conll18_ud_eval.py

Content	`nsubj, obj, iobj, csubj, ccomp, xcomp, obl, vocative, expl, dislocated, advcl, advmod, discourse, nmod, appos, nummod, acl, amod, conj, fixed, flat, compound, list, parataxis, orphan, goeswith, reparandum, root, dep`
Function	`aux, cop, mark, det, clf, case, cc`
Ignored	`punct`

Table 3: Universal dependency relations considered as pertaining to content words and function words, respectively, in MLAS. Content word relations are evaluated directly; words attached via functional relations are treated as features of their parent nodes.

Features	`PronType, NumType, Poss, Reflex, Foreign, Abbr, Gender, Animacy, Number, Case, Definite, Degree, VerbForm, Mood, Tense, Aspect, Voice, Evident, Polarity, Person, Polite`

Table 4: Universal features whose values are evaluated in MLAS. Any other features are ignored.

universal dependency labels were taken into account, which means that language-specific subtypes such as `expl:pv` (pronoun of a pronominal verb), a subtype of the universal relation `expl` (expletive), were truncated to `expl` both in the gold standard and in the system output before comparing them.

In the end-to-end evaluation of our task, LAS is re-defined as the harmonic mean (F_1) of precision P and recall R, where

$$P = \frac{\#\mathit{correctRelations}}{\#\mathit{systemNodes}} \tag{1}$$

$$R = \frac{\#\mathit{correctRelations}}{\#\mathit{goldNodes}} \tag{2}$$

$$LAS = \frac{2PR}{P+R} \tag{3}$$

Note that attachment of all nodes including punctuation is evaluated. LAS is computed separately for each of the 82 test files and a macro-average of all these scores is used to rank the systems.

3.2 MLAS: Morphology-Aware Labeled Attachment Score

MLAS aims at cross-linguistic comparability of the scores. It is an extension of CLAS (Nivre and Fang, 2017), which was tested experimentally in the 2017 task. CLAS focuses on dependencies between content words and disregards attachment of function words; in MLAS, function words are not ignored, but they are treated as features of content words. In addition, part-of-speech tags and morphological features are evaluated, too.

The idea behind MLAS is that function words often correspond to morphological features in other languages. Furthermore, languages with many function words (e.g., English) have longer sentences than morphologically rich languages (e.g., Finnish), hence a single error in Finnish costs the parser significantly more than an error in English according to LAS.

The core part is identical to LAS (Section 3.1): for aligned system and gold nodes, their respective parent nodes are considered; if the system parent is not aligned with the gold parent, or if the universal relation label differs, the word is not counted as correctly attached. Unlike LAS, certain types of relations (Table 3) are not evaluated directly. Words attached via such relations (in either system or gold data) are not counted as independent words. Instead, they are treated as features of the content words they belong to. Therefore, a system-produced word counts as correct if it is aligned and attached correctly, its universal POS tag and selected morphological features (Table 4) are correct, all its function words are attached correctly, and their POS tags and features are also correct. Punctuation nodes are neither content nor function words; their attachment is ignored in MLAS.

3.3 BLEX: Bilexical Dependency Score

BLEX is similar to MLAS in that it focuses on relations between content words. Instead of morphological features, it incorporates lemmatization in the evaluation. It is thus closer to semantic content and evaluates two aspects of UD annotation that are important for language understanding: dependencies and lexemes. The inclusion of this metric should motivate the competing teams to model lemmas, the last important piece of annotation that is not captured by the other metrics. A system that scores high in all three metrics will thus be a general-purpose language-analysis tool that tackles segmentation, morphology and surface syntax.

Computation of BLEX is analogous to LAS and MLAS. Precision and recall of correct attachments is calculated, attachment of function words and punctuation is ignored (Table 3). An attachment is correct if the parent and child nodes are aligned to the corresponding nodes in gold standard, if the universal dependency label is correct, and if the lemma of the child node is correct.

A few UD treebanks lack lemmatization (or, as in Uyghur, have lemmas only for some words and not for others). A system may still be able to predict the lemmas if it learns them in other treebanks. Such system should not be penalized just because no gold standard is available; therefore, if the gold lemma is a single underscore character ("_"), any system-produced lemma is considered correct.

3.4 Token Alignment

UD defines two levels of token/word segmentation. The lower level corresponds to what is usually understood as tokenization. However, unlike some popular tokenization schemes, it does not include any normalization of the non-whitespace characters. We can safely assume that any two tokenizations of a text differ only in whitespace while the remaining characters are identical. There is thus a 1-1 mapping between gold and system non-whitespace characters, and two tokens are aligned if all their characters match.

3.5 Syntactic Word Alignment

The higher segmentation level is based on the notion of *syntactic word*. Some languages contain *multi-word tokens* (MWT) that are regarded as contractions of multiple syntactic words. For example, the German token *zum* is a contraction of the preposition *zu* "to" and the article *dem* "the".

Syntactic words constitute independent nodes in dependency trees. As shown by the example, it is not required that the MWT is a pure concatenation of the participating words; the simple token alignment thus does not work when MWTs are involved. Fortunately, the CoNLL-U file format used in UD clearly marks all MWTs so we can detect them both in system output and in gold data. Whenever one or more MWTs have overlapping spans of surface character offsets, the longest common subsequence algorithm is used to align syntactic words within these spans.

3.6 Sentence Segmentation

Words are aligned and dependencies are evaluated in the entire file without considering sentence segmentation. Still, the accuracy of sentence boundaries has an indirect impact on attachment scores: any missing or extra sentence boundary necessarily makes one or more dependency relations incorrect.

3.7 Invalid Output

If a system fails to produce one of the 82 files or if the file is not valid CoNLL-U format, the score of that file (counting towards the system's macro-average) is zero.

Formal validity is defined more leniently than for UD-released treebanks. For example, a nonexistent dependency type does not render the whole file invalid, it only costs the system one incorrect relation. However, cycles and multi-root sentences are disallowed. A file is also invalid if there are character mismatches that could make the token-alignment algorithm fail.

3.8 Extrinsic Parser Evaluation

The metrics described above are all *intrinsic* measures: they evaluate the grammatical analysis task per se, with the hope that better scores correspond to output that is more useful for downstream NLP applications. Nevertheless, such correlations are not automatically granted. We thus seek to complement our task with an *extrinsic* evaluation, where the output of parsing systems is exploited by applications like biological event extraction, opinion analysis and negation scope resolution.

This optional track involves English only. It is organized in collaboration with the EPE initiative;[7] for details see Fares et al. (2018).

4 TIRA: The System Submission Platform

Similarly to our 2017 task and to some other recent CoNLL shared tasks, we employed the cloud-

[7] http://epe.nlpl.eu/

based evaluation platform TIRA (Potthast et al., 2014),[8] which implements the *evaluation as a service* paradigm (Hanbury et al., 2015). Instead of processing test data on their own hardware and submitting the outputs, participants submit working software. Naturally, software submissions bring about additional overhead for both organizers and participants, whereas the goal of an evaluation platform like TIRA is to reduce this overhead to a bearable level.

4.1 Blind Evaluation

Traditionally, evaluations in shared tasks are half-blind (the test data are shared with participants while the ground truth is withheld). TIRA enables fully blind evaluation, where the software is locked in a datalock together with the test data, its output is recorded but all communication channels to the outside are closed or tightly moderated. The participants do not even see the input to their software. This feature of TIRA was not too important in the present task, as UD data is not secret, and the participants were simply trusted that they would not exploit any knowledge of the test data they might have access to.

However, closing down all communication channels also has its downsides, since participants cannot check their running software; before the system run completes, even the task moderator does not see whether the system is really producing output and not just sitting in an endless loop. In order to alleviate this extra burden, we made two modifications compared to the previous year: 1. Participants were explicitly advised to invoke shorter runs that process only a subset of the test files. The organizers would then stitch the partial runs into one set of results. 2. Participants were able to see their scores on the test set rounded to the nearest multiple of 5%. This way they could spot anomalies possibly caused by ill-selected models. The exact scores remained hidden because we did not want the participants to fine-tune their systems against the test data.

4.2 Replicability

It is desirable that published experiments can be re-run yielding the same results, and that the algorithms can be tested on alternative test data in the future. Ensuring both requires that a to-be-evaluated software is preserved in working condition for as long as possible. TIRA supplies participants with a virtual machine, offering a range of commonly used operating systems. Once deployed and tested, the virtual machines are archived to preserve the software within.

In addition, some participants agreed to share their code so that we decided to collect the respective projects in an open source repository hosted on GitHub.[9]

5 Baseline System

We prepared a set of baseline models using UDPipe 1.2 (Straka and Straková, 2017).

The baseline models were released together with the UD 2.2 training data. For each of the 73 treebanks with non-empty training data we trained one UDPipe model, utilizing training data for training and development data for hyperparameter tuning. If a treebank had no development data, we cut 10% of the training sentences and considered it as development data for the purpose of tuning hyperparameters of the baseline model (employing only the remainder of the original training data for the actual training in that case).

In addition to the treebank-specific models, we also trained a “mixed model” on samples from all treebanks. Specifically, we utilized the first 200 training sentences of each treebank (or less in case of small treebanks) as training data, and at most 20 sentences from each treebank’s development set as development data.

The baseline models, together with all information needed to replicate them (hyperparameters, the modified train-dev split where applicable, and pre-computed word embeddings for the parser) are available from `http://hdl.handle.net/11234/1-2859`.

Additionally, the released archive also contains the training and development data with predicted morphology. Morphology in development data was predicted using the baseline models, morphology in training data via “jack-knifing” (split the training set into 10 parts, train a model on 9 parts, use it to predict morphology in the tenth part, repeat for all 10 target parts). The same hyperparameters were used as those used to train the baseline model on the entire training set.

The UDPipe baseline models are able to reconstruct nearly all annotation from CoNLL-U files – they can generate segmentation, tokenization,

[8] `http://www.tira.io/`

[9] `https://github.com/CoNLL-UD-2018`

Treebank without training data	Substitution model
Breton KEB	mixed model
Czech PUD	Czech PDT
English PUD	English EWT
Faroese OFT	mixed model
Finnish PUD	Finnish TDT
Japanese Modern	Japanese GSD
Naija NSC	mixed model
Swedish PUD	Swedish Talbanken
Thai PUD	mixed model

Table 5: Substitution models of the baseline systems for treebanks without training data.

multi-word token splitting, morphological annotation (lemmas, UPOS, XPOS and FEATS) and dependency trees. Participants were free to use any part of the model in their systems – for all test sets, we provided UDPipe processed variants in addition to raw text inputs.

Baseline UDPipe Shared Task System The shared task baseline system employs the UDPipe 1.2 baseline models. For the nine treebanks without their own training data, a substitution model according to Table 5 was used.

6 Results

6.1 Official Parsing Results

Table 6 gives the main ranking of participating systems by the LAS F_1 score macro-averaged over all 82 test files. The table also shows the performance of the baseline UDPipe system; 17 of the 25 systems managed to outperform it. The baseline is comparatively weaker than in the 2017 task (only 12 out of 32 systems beat the baseline there). The ranking of the baseline system by MLAS is similar (Table 7) but in BLEX, the baseline jumps to rank 13 (Table 8). Besides the simple explanation that UDPipe 1.2 is good at lemmatization, we could also hypothesize that some teams put less effort in building lemmatization models (see also the last column in Table 10).

Each ranking has a different winning system, although the other two winners are typically closely following. The same 8–10 systems occupy best positions in all three tables, though with variable mutual ranking. Some teams seem to have deliberately neglected some of the evaluated attributes: Uppsala is rank 7 in LAS and MLAS, but 24 in BLEX; IBM NY is rank 13 in LAS but 24 in MLAS and 23 in BLEX.

Team	LAS
1. HIT-SCIR (Che et al.)	75.84
2. TurkuNLP (Kanerva et al.)	73.28
3. UDPipe Future (Straka)	73.11
LATTICE (Lim et al.)	73.02
ICS PAS (Rybak and Wróblewska)	73.02
6. CEA LIST (Duthoo and Mesnard)	72.56
7. Uppsala (Smith et al.)	72.37
Stanford (Qi et al.)	72.29
9. AntNLP (Ji et al.)	70.90
NLP-Cube (Boroș et al.)	70.82
11. ParisNLP (Jawahar et al.)	70.64
12. SLT-Interactions (Bhat et al.)	69.98
13. IBM NY (Wan et al.)	69.11
14. UniMelb (Nguyen and Verspoor)	68.66
15. LeisureX (Li et al.)	68.31
16. KParse (Kırnap et al.)	66.58
17. Fudan (Chen et al.)	66.34
18. BASELINE UDPipe 1.2	65.80
19. Phoenix (Wu et al.)	65.61
20. CUNI x-ling (Rosa and Mareček)	64.87
21. BOUN (Özateş et al.)	63.54
22. ONLP lab (Seker et al.)	58.35
23. iParse (no paper)	55.83
24. HUJI (Hershcovich et al.)	53.69
25. ArmParser (Arakelyan et al.)	47.02
26. SParse (Önder et al.)	1.95

Table 6: Ranking of the participating systems by the labeled attachment F_1-score **(LAS),** macro-averaged over 82 test sets. Pairs of systems with significantly ($p < 0.05$) different LAS are separated by a line. Citations refer to the corresponding system-description papers in this volume.

While the LAS scores on individual treebanks are comparable to the 2017 task, the macro average is not, because the set of treebanks is different, and the impact of low-resource languages seems to be higher in the present task.

We used bootstrap resampling to compute 95% confidence intervals: they are in the range ±0.11 to ±0.16 (% LAS/MLAS/BLEX) for all systems except SParse (where it is ±0.00).

	Team	MLAS
1.	UDPipe Future (Praha)	61.25
2.	TurkuNLP (Turku)	60.99
	Stanford (Stanford)	60.92
4.	ICS PAS (Warszawa)	60.25
5.	CEA LIST (Paris)	59.92
6.	HIT-SCIR (Harbin)	59.78
7.	Uppsala (Uppsala)	59.20
8.	NLP-Cube (Bucureşti)	57.32
9.	LATTICE (Paris)	57.01
10.	AntNLP (Shanghai)	55.92
11.	ParisNLP (Paris)	55.74
12.	SLT-Interactions (Bengaluru)	54.52
13.	LeisureX (Shanghai)	53.70
	UniMelb (Melbourne)	53.62
15.	KParse (İstanbul)	53.25
16.	Fudan (Shanghai)	52.69
17.	BASELINE UDPipe 1.2	52.42
	Phoenix (Shanghai)	52.26
19.	BOUN (İstanbul)	50.40
	CUNI x-ling (Praha)	50.35
21.	ONLP lab (Ra'anana)	46.09
22.	iParse (Pittsburgh)	45.65
23.	HUJI (Yerushalayim)	44.60
24.	IBM NY (Yorktown Heights)	40.61
25.	ArmParser (Yerevan)	36.28
26.	SParse (İstanbul)	1.68

Table 7: Ranking of the participating systems by **MLAS,** macro-averaged over 82 test sets. Pairs of systems with significantly ($p < 0.05$) different MLAS are separated by a line.

	Team	BLEX
1.	TurkuNLP (Turku)	66.09
2.	HIT-SCIR (Harbin)	65.33
3.	UDPipe Future (Praha)	64.49
	ICS PAS (Warszawa)	64.44
5.	Stanford (Stanford)	64.04
6.	LATTICE (Paris)	62.39
	CEA LIST (Paris)	62.23
8.	AntNLP (Shanghai)	60.91
9.	ParisNLP (Paris)	60.70
10.	SLT-Interactions (Bengaluru)	59.68
11.	UniMelb (Melbourne)	58.67
12.	LeisureX (Shanghai)	58.42
13.	BASELINE UDPipe 1.2	55.80
	Phoenix (Shanghai)	55.71
15.	NLP-Cube (Bucureşti)	55.52
16.	KParse (İstanbul)	55.26
17.	CUNI x-ling (Praha)	54.07
	Fudan (Shanghai)	54.03
19.	BOUN (İstanbul)	53.45
20.	iParse (Pittsburgh)	48.71
21.	HUJI (Yerushalayim)	48.05
22.	ArmParser (Yerevan)	39.18
23.	IBM NY (Yorktown Heights)	32.55
24.	Uppsala (Uppsala)	32.09
25.	ONLP lab (Ra'anana)	28.29
26.	SParse (İstanbul)	1.71

Table 8: Ranking of the participating systems by **BLEX,** macro-averaged over 82 test sets. Pairs of systems with significantly ($p < 0.05$) different BLEX are separated by a line.

We used paired bootstrap resampling to compute whether the difference between two neighboring systems is significant ($p < 0.05$).[10]

6.2 Secondary Metrics

In addition to the main LAS ranking, we evaluated the systems along multiple other axes, which may shed more light on their strengths and weaknesses. This section provides an overview of selected secondary metrics for systems matching or surpassing the baseline; a large number of additional results are available at the shared task website.[11]

The website also features a LAS ranking of unofficial system runs, i.e. those that were not marked by their teams as primary runs, or were even run after the official evaluation phase closed and test data were unblinded. The difference from the official results is much less dramatic than in 2017, with the exception of the team SParse, who managed to fix their software and produce more valid output files.

As an experiment, we also applied the 2017 system submissions to the 2018 test data. This allows us to test how many systems can actually be used to produce new data without a glitch, as well as to see to what extent the results change over one year and two releases of UD. Here it should be noted that not all of the 2018 task languages and treebanks were present in the 2017 task, therefore causing many systems fail due to an unknown language or treebank code. The full results of this

[10]Using Udapi (Popel et al., 2017) eval.Conll18, marked by the presence or absence of horizontal lines in Tables 6–8.

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

Team	Toks	Wrds	Sents
1. Uppsala	97.60	98.18	83.80
2. HIT-SCIR	**98.42**	98.12	**83.87**
3. CEA LIST	98.16	97.78	82.79
4. CUNI x-ling	98.09	97.74	**82.80**
5. TurkuNLP	97.83	97.42	**83.03**
6. SLT-Interactions	97.51	97.09	83.01
7. UDPipe Future	97.46	97.04	**83.64**
8. Phoenix	97.46	97.03	82.91
9. BASELINE UDPipe	97.39	96.97	**83.01**
ParisNLP	97.39	96.97	83.01
AntNLP	97.39	96.97	83.01
UniMelb	97.39	96.97	83.01
BOUN	97.39	96.97	83.01
ICS PAS	97.39	96.97	83.01
LATTICE	97.39	96.97	83.01
LeisureX	97.39	96.97	83.01
KParse	97.39	96.97	83.01
18. Fudan	97.38	96.96	82.85
19. IBM NY	97.30	96.92	**83.51**
20. ONLP lab	97.28	96.86	83.00
21. NLP-Cube	**97.36**	96.80	82.55
22. Stanford	96.19	95.99	76.55
23. HUJI	94.95	94.61	**80.84**
24. ArmParser	79.75	79.41	13.33
25. iParse	78.45	78.11	**68.37**
26. SParse	2.32	2.32	2.34

Table 9: Tokenization, word segmentation and sentence segmentation (ordered by word F_1 scores; out-of-order scores in the other two columns are bold).

Team	UPOS	Feats	Lemm
1. Uppsala	90.91	87.59	58.50
2. HIT-SCIR	90.19	84.24	**88.82**
3. CEA LIST	89.97	**86.83**	**88.90**
4. TurkuNLP	89.81	86.70	**91.24**
5. LATTICE	89.53	83.74	87.84
6. UDPipe Future	89.37	**86.67**	**89.32**
7. Stanford	89.01	85.47	88.32
8. ICS PAS	88.70	85.14	87.99
9. CUNI x-ling	88.68	84.56	**88.96**
10. NLP-Cube	88.50	**85.08**	81.21
11. SLT-Interactions	88.12	83.72	**87.51**
12. IBM NY	88.02	59.11	59.51
13. UniMelb	87.90	**83.74**	**87.84**
14. KParse	87.62	**84.32**	86.26
15. Phoenix	87.49	83.87	**87.69**
16. ParisNLP	87.35	83.74	**87.84**
17. BASELINE UDPipe	87.32	83.74	87.84
AntNLP	87.32	83.74	87.84
19. ONLP lab	87.25	83.67	57.10
20. Fudan	87.25	83.47	**85.91**
21. BOUN	87.19	**83.73**	**87.68**
22. LeisureX	87.15	83.46	**87.77**
23. HUJI	85.06	81.51	85.61
24. ArmParser	72.99	69.91	72.22
25. iParse	71.38	68.64	71.68
26. SParse	2.25	2.29	2.28

Table 10: Universal POS tags, features and lemmas (ordered by UPOS F_1 scores; out-of-order scores in the other two columns are bold).

experiment are available on the shared task website.[12]

Table 9 evaluates detection of tokens, syntactic words and sentences. About a third of the systems trusted the baseline segmentation; this is less than in 2017. For most languages and in aggregate, the segmentation scores are very high and their impact on parsing scores is not easy to prove; but it likely played a role in languages where segmentation is hard. For example, HIT-SCIR's word segmentation in Vietnamese surpasses the second system by a margin of 6 percent points; likewise, the system's advantage in LAS and MLAS (but not in BLEX!) amounts to 7–8 points. Similarly, Uppsala and ParisNLP achieved good segmentation scores (better than their respective macro-averages) on Arabic. They were able to translate it into better LAS, but not MLAS and BLEX, where there were too many other chances to make an error.

The complexity of the new metrics, especially MLAS, is further underlined by Table 10: Uppsala is the clear winner in both UPOS tags and morphological features, but 6 other teams had better dependency relations and better MLAS. Note that as with segmentation, morphology predicted by the baseline system was available, though only a few systems seem to have used it without attempting to improve it.

6.3 Partial Results

Table 11 gives the three main scores averaged over the 61 "big" treebanks (training data larger than

[12] http://universaldependencies.org/conll18/results-2017-systems.html

Team	LAS	MLAS	BLEX
1. HIT-SCIR	84.37	70.12	75.05
2. Stanford	83.03	**72.67**	**75.46**
3. TurkuNLP	81.85	71.27	**75.83**
4. UDPipe Future	81.83	**71.71**	74.67
5. ICS PAS	81.72	70.30	74.42
6. CEA LIST	81.66	**70.89**	72.32
7. LATTICE	80.97	66.27	71.50
8. NLP-Cube	80.48	**67.79**	64.76
9. ParisNLP	80.29	65.88	**70.95**
10. Uppsala	80.25	**68.81**	36.02
11. SLT-Interactions	79.67	64.95	**69.77**
12. AntNLP	79.61	**65.43**	**70.34**
13. LeisureX	77.98	63.79	68.55
14. UniMelb	77.69	63.17	68.25
15. IBM NY	77.55	47.34	36.68
16. Fudan	75.42	**62.28**	**62.90**
17. KParse	74.84	**62.40**	**63.84**
18. BASELINE UDPipe	74.14	61.27	**64.67**
19. Phoenix	73.93	61.12	64.47
20. BOUN	72.85	60.00	62.99
21. CUNI x-ling	71.54	58.33	61.63
22. ONLP lab	67.08	55.20	33.08
23. iParse	66.55	**55.37**	**58.80**
24. HUJI	62.07	53.20	56.90
25. ArmParser	58.14	45.87	49.25
26. SParse	2.63	2.26	2.30

Table 11: Average LAS on the 61 “big” treebanks (ordered by LAS F_1 scores; out-of-order scores in the other two columns are bold).

Team	LAS	MLAS	BLEX
1. CUNI x-ling	27.89	6.13	13.98
2. Uppsala	25.87	5.16	9.03
3. CEA LIST	23.90	3.75	**10.99**
4. HIT-SCIR	23.88	2.88	10.50
5. LATTICE	23.39	**4.38**	10.01
6. TurkuNLP	22.91	3.59	**11.40**
7. IBM NY	21.88	2.62	7.17
8. UDPipe Future	21.75	**2.82**	**8.80**
9. ICS PAS	19.26	1.89	6.17
10. AntNLP	18.59	**3.43**	**8.61**
11. KParse	17.84	3.32	6.58
12. SLT-Interactions	17.47	1.79	**6.95**
13. Stanford	17.45	**2.76**	**7.63**
14. BASELINE UDPipe	17.17	**3.44**	7.63
UniMelb	17.17	3.44	7.63
16. LeisureX	17.16	3.43	7.63
17. Phoenix	16.99	3.02	**8.00**
18. NLP-Cube	16.85	**3.39**	7.05
19. ParisNLP	16.52	2.53	6.75
20. ONLP lab	15.98	**3.58**	4.96
21. Fudan	15.45	2.98	**6.61**
22. BOUN	14.78	2.59	6.43
23. HUJI	8.53	0.92	2.77
24. ArmParser	7.47	**1.86**	**3.54**
25. iParse	2.82	0.23	0.97
26. SParse	0.00	0.00	0.00

Table 12: Average LAS, MLAS and BLEX on the 9 low-resource languages: Armenian (hy), Breton (br), Buryat (bxr), Faroese (fo), Kazakh (kk), Kurmanji (kmr), Naija (pcm), Thai (th) and Upper Sorbian (hsb) (ordered by LAS F_1 scores; out-of-order scores in the other two columns are bold).

test data, development data available). Higher scores reflect the fact that models for these test sets are easier to learn: enough data is available, no cross-lingual or cross-domain learning is necessary (the extra test sets are not included here). Regarding ranking, the Stanford system makes a remarkable jump when it does not have to carry the load of underresourced languages: from rank 8 to 2 in LAS, from 3 to 1 in MLAS and from 5 to 2 in BLEX.

Table 12 gives the LAS F_1 score on the nine low-resource languages only. Here we have a true specialist: The team CUNI x-ling lives up to its name and wins in all three scores, although in the overall ranking they fall even slightly behind the baseline. On the other hand, the scores are extremely low and the outputs are hardly useful for any downstream application. Especially morphology is almost impossible to learn from foreign languages, hence the much lower values of MLAS and BLEX. BLEX is a bit better than MLAS, which could be explained by cases where a word form is identical to its lemma. However, there are significant language-by-language differences; the best LAS on Faroese and Upper Sorbian surpassing 45%. This probably owes to the presence of many Germanic and Slavic treebanks in training data, including some of the largest datasets in UD. Three languages, Buryat, Kurmanji and Upper Sorbian, were introduced in the 2017 task as

	Team	LAS	MLAS	BLEX
1.	HIT-SCIR	69.53	45.94	53.30
2.	LATTICE	68.12	45.03	51.71
3.	ICS PAS	66.90	**49.24**	**54.89**
4.	TurkuNLP	64.48	47.63	53.54
5.	UDPipe Future	64.21	47.53	49.53
6.	AntNLP	63.73	42.24	48.31
7.	Uppsala	63.60	**46.00**	29.25
8.	ParisNLP	60.84	40.71	**46.08**
9.	CEA LIST	57.34	39.97	43.39
10.	KParse	57.32	39.20	**43.61**
11.	NLP-Cube	56.78	37.13	38.30
12.	SLT-Interactions	56.74	35.73	**42.90**
13.	IBM NY	56.13	26.51	25.23
14.	UniMelb	56.12	**36.09**	**42.09**
15.	BASELINE UDPipe	55.01	**38.80**	41.06
	LeisureX	55.01	38.80	41.06
17.	Phoenix	54.63	38.38	40.72
	Fudan	54.63	38.15	40.07
19.	CUNI x-ling	54.33	38.10	**40.70**
20.	BOUN	50.18	34.29	36.75
21.	Stanford	48.56	**34.86**	**38.55**
22.	ONLP lab	47.49	32.74	22.39
23.	iParse	38.79	28.03	**29.62**
24.	HUJI	36.74	24.47	27.70
25.	ArmParser	34.54	22.94	25.26
26.	SParse	0.00	0.00	0.00

Table 13: Average attachment score on the 7 small treebanks: Galician TreeGal, Irish, Latin Perseus, North Sámi, Norwegian Nynorsk LIA, Russian Taiga and Slovenian SST (ordered by LAS F_1 scores; out-of-order scores in the other two columns are bold).

	Team	LAS	MLAS	BLEX
1.	HIT-SCIR	74.20	55.52	62.34
2.	Stanford	73.14	**58.75**	61.96
3.	LATTICE	72.34	55.60	60.42
4.	Uppsala	72.27	**57.80**	29.73
5.	ICS PAS	72.18	**58.07**	**60.97**
6.	TurkuNLP	71.78	57.54	**63.25**
7.	UDPipe Future	71.57	**57.93**	61.52
8.	CEA LIST	70.45	54.99	57.83
9.	NLP-Cube	69.83	**55.01**	54.15
10.	IBM NY	69.40	46.59	38.12
11.	AntNLP	68.87	**53.47**	**57.71**
12.	UniMelb	68.72	52.05	56.77
13.	Phoenix	66.97	**52.26**	55.69
14.	BASELINE UDPipe	66.63	51.75	54.87
15.	KParse	66.55	51.29	54.45
16.	SLT-Interactions	64.73	48.47	**54.90**
17.	CUNI x-ling	64.70	**49.71**	52.72
18.	ParisNLP	64.09	48.79	**53.16**
19.	Fudan	63.54	45.54	50.73
20.	LeisureX	61.05	41.95	50.60
21.	BOUN	56.46	41.91	45.12
22.	HUJI	56.35	**46.52**	**50.10**
23.	iParse	44.20	33.43	38.18
24.	ONLP lab	43.33	30.20	20.08
25.	ArmParser	0.00	0.00	0.00
	SParse	0.00	0.00	0.00

Table 14: Average attachment score on the 5 additional test sets for high-resource languages: Czech PUD, English PUD, Finnish PUD, Japanese Modern and Swedish PUD (ordered by LAS F_1 scores; out-of-order scores in the other two columns are bold).

surprise languages and had higher scores there.[13] This is because in 2017, the segmentation, POS tags and morphology UDPipe models were trained on the test data, applied to it via cross-validation, and made available to the systems. Such an approach makes the conditions unrealistic, therefore it was not repeated this year. Consequently, parsing these languages is now much harder.

In contrast, the results on the 7 treebanks with "small" training data and no development data (Table 13) are higher on average, but again the variance is significant. The smallest treebank in the group, Norwegian Nynorsk LIA, has only 3583 training words. There are two larger Norwegian treebanks that could be used as additional training sources. However, the LIA treebank consists of spoken dialects and is probably quite dissimilar to the other treebanks. The same can be said about Slovenian SST and the other Slovenian treebank; SST is the most difficult dataset in the group, despite of having almost 20K of its own training words. Other treebanks, like Russian Taiga and Galician TreeGal, have much better scores (74% LAS, about 61% MLAS and 64% BLEX). There are also two treebanks that are the sole representatives of their languages: Irish and North Sámi. Their best LAS is around 70%: com-

[13] The fourth surprise language, North Sámi, has now additional training data and does not fall in the low-resource category.

parable to Nynorsk LIA but much better than SST. ICS PAS is the most successful system in the domain of small treebanks, especially when judged by MLAS and BLEX.

Table 14 gives the average LAS on the 5 extra test sets (no own training data, but other treebanks of the same language exist). Four of them come from the Parallel UD (PUD) collection introduced in the 2017 task (Zeman et al., 2017). The fifth, Japanese Modern, turned out to be one of the toughest test sets in this shared task. There is another Japanese treebank, GSD, with over 160K training tokens, but the Modern dataset seems almost inapproachable with models trained on GSD. A closer inspection reveals why: despite its name, it is actually a corpus of historical Japanese, although from the relatively recent Meiji and Taishō periods (1868–1926). An average sentence in GSD is about 1.3× longer than in Modern. GSD has significantly more tokens tagged as auxiliaries, but more importantly, the top ten `AUX` lemmas in the two treebanks are completely disjoint sets. Some other words are out-of-vocabulary because their preferred spelling changed. For instance, the demonstrative pronoun *sore* is written using hiragana in GSD, but a kanji character is used in Modern. Striking differences can be observed also in dependency relations: in GSD, 3.7% relations are `nsubj` (subject), and 1.2% are `cop` (copula). In Modern, there is just 0.13% of subjects, and not a single occurrence of a copula.

See Tables 15, 16 and 17 for a ranking of all test sets by the best scores achieved on them by any parser. Note that this cannot be directly interpreted as a ranking of languages by their parsing difficulty: many treebanks have high ranks simply because the corresponding training data is large. Table 18 compares average LAS and MLAS for each treebank.

Finally, Tables 19 and 20 show the treebanks where word and sentence segmentation was extremely difficult (judged by the average parser score). Not surprisingly, word segmentation is difficult for the low-resource languages and for languages like Chinese, Vietnamese, Japanese and Thai, where spaces do not separate words. Notably the Japanese GSD set is not as difficult, but whoever trusted it, crashed on the "Modern" set. Sentence segmentation was particularly hard for treebanks without punctuation, i.e., most of the classical languages and spoken data.

Treebank	LAS	Best system	Avg	StDev
1. pl_lfg	94.86	HIT-SCIR	85.89	± 6.97
2. ru_syntagrus	92.48	HIT-SCIR	79.68	± 9.09
3. hi_hdtb	92.41	HIT-SCIR	**85.16**	± 5.32
4. pl_sz	92.23	HIT-SCIR	81.47	± 7.27
5. cs_fictree	92.02	HIT-SCIR	**82.10**	± 7.26
6. it_isdt	92.00	HIT-SCIR	**87.61**	± 4.12
7. cs_pdt	91.68	HIT-SCIR	82.18	± 6.91
8. ca_ancora	91.61	HIT-SCIR	**83.61**	± 6.01
9. cs_cac	91.61	HIT-SCIR	82.69	± 6.93
10. sl_ssj	91.47	HIT-SCIR	75.00	± 9.13
11. no_bokmaal	91.23	HIT-SCIR	**79.80**	± 7.29
12. bg_btb	91.22	HIT-SCIR	**82.52**	± 5.88
13. no_nynorsk	90.99	HIT-SCIR	78.55	± 7.88
14. es_ancora	90.93	HIT-SCIR	**82.84**	± 6.17
15. fi_pud	90.23	HIT-SCIR	68.87	±15.61
16. fr_sequoia	89.89	LATTICE	**80.55**	± 5.91
17. el_gdt	89.65	HIT-SCIR	**80.65**	± 6.05
18. nl_alpino	89.56	HIT-SCIR	77.76	± 7.42
19. sk_snk	88.85	HIT-SCIR	76.53	± 7.24
20. fi_tdt	88.73	HIT-SCIR	73.55	± 9.39
21. sr_set	88.66	Stanford	**79.84**	± 6.57
22. sv_talbanken	88.63	HIT-SCIR	77.71	± 6.50
23. fi_ftb	88.53	HIT-SCIR	76.89	± 7.60
24. uk_iu	88.43	HIT-SCIR	72.47	± 8.25
25. fa_seraji	88.11	HIT-SCIR	**78.71**	± 6.04
26. en_pud	87.89	LATTICE	74.51	± 8.28
27. pt_bosque	87.81	Stanford	**80.49**	± 5.46
28. hr_set	87.36	HIT-SCIR	78.37	± 6.42
29. fro_srcmf	87.12	UDPipe Future	74.38	±16.74
30. la_ittb	87.08	HIT-SCIR	**77.00**	± 7.42
31. ko_kaist	86.91	HIT-SCIR	**77.10**	± 8.72
32. fr_gsd	86.89	HIT-SCIR	**79.43**	± 5.47
33. ro_rrt	86.87	HIT-SCIR	75.77	± 7.66
34. nl_lassysmall	86.84	HIT-SCIR	75.08	± 6.59
35. da_ddt	86.28	HIT-SCIR	75.02	± 6.47
36. cs_pud	86.13	HIT-SCIR	73.24	± 9.97
37. af_afribooms	85.47	HIT-SCIR	**76.61**	± 6.17
38. et_edt	85.35	HIT-SCIR	72.08	± 8.71
39. ko_gsd	85.14	HIT-SCIR	71.88	±10.53
40. en_gum	85.05	LATTICE	**74.20**	± 6.27
41. en_ewt	84.57	HIT-SCIR	**75.99**	± 5.40
42. eu_bdt	84.22	HIT-SCIR	72.08	± 8.83
43. sv_lines	84.08	HIT-SCIR	**73.76**	± 5.98
44. lv_lvtb	83.97	HIT-SCIR	67.76	± 9.01
45. ur_udtb	83.39	HIT-SCIR	**75.89**	± 4.69
46. ja_gsd	83.11	HIT-SCIR	73.68	± 4.55
47. gl_ctg	82.76	Stanford	72.46	± 7.13
48. hu_szeged	82.66	HIT-SCIR	67.05	± 8.63
49. en_lines	81.97	HIT-SCIR	**72.28**	± 5.59
50. de_gsd	80.36	HIT-SCIR	70.13	± 7.14
51. sv_pud	80.35	HIT-SCIR	67.02	± 9.23
52. id_gsd	80.05	HIT-SCIR	**73.05**	± 4.69
53. it_postwita	79.39	HIT-SCIR	64.95	± 6.88
54. grc_perseus	79.39	HIT-SCIR	59.01	±15.56
55. grc_proiel	79.25	HIT-SCIR	**65.02**	±14.58
56. ar_padt	77.06	Stanford	64.07	± 6.41
57. zh_gsd	76.77	HIT-SCIR	60.32	± 6.14
58. he_htb	76.09	Stanford	58.73	± 5.29
59. fr_spoken	75.78	HIT-SCIR	**64.66**	± 5.35
60. cu_proiel	75.73	Stanford	62.64	± 6.98
61. gl_treegal	74.25	UDPipe Future	**64.65**	± 5.61
62. ru_taiga	74.24	ICS PAS	56.27	± 9.16
63. la_proiel	73.61	HIT-SCIR	**61.25**	± 6.87
64. la_perseus	72.63	HIT-SCIR	46.91	±11.12
65. ga_idt	70.88	TurkuNLP	**58.37**	± 7.05
66. no_nynorsklia	70.34	HIT-SCIR	50.33	± 9.28
67. sme_giella	69.87	LATTICE	**51.10**	±14.32
68. got_proiel	69.55	Stanford	**60.55**	± 4.93
69. ug_udt	67.05	HIT-SCIR	54.27	± 6.90
70. tr_imst	66.44	HIT-SCIR	**55.61**	± 6.49
71. sl_sst	61.39	HIT-SCIR	47.07	± 5.84
72. vi_vtb	55.22	HIT-SCIR	40.40	± 4.43
73. fo_oft	49.43	CUNI x-ling	27.87	± 9.75
74. hsb_ufal	46.42	SLT-Interactions	26.48	± 8.90
75. br_keb	38.64	CEA LIST	13.27	± 8.77
76. hy_armtdp	37.01	LATTICE	**22.39**	± 7.91
77. kk_ktb	31.93	Uppsala	19.11	± 6.34
78. kmr_mg	30.41	IBM NY	**20.27**	± 6.14
79. pcm_nsc	30.07	CUNI x-ling	13.19	± 5.76
80. ja_modern	28.33	Stanford	**18.92**	± 5.14
81. bxr_bdt	19.53	AntNLP	11.45	± 4.28
82. th_pud	13.70	CUNI x-ling	1.38	± 2.83

Table 15: Treebank ranking by best parser LAS (Avg=average LAS over all systems, out-of-order scores in bold).

	Treebank	MLAS	Best system	Avg	StDev
1.	pl_lfg	86.93	UDPipe Future	73.73	± 7.29
2.	ru_syntagrus	86.76	UDPipe Future	71.63	± 9.36
3.	cs_pdt	85.10	UDPipe Future	**73.61**	± 6.32
4.	cs_fictree	84.23	ICS PAS	69.91	± 7.77
5.	ca_ancora	84.07	UDPipe Future	**74.62**	± 7.69
6.	es_ancora	83.93	Stanford	74.61	± 7.43
7.	it_isdt	83.89	Stanford	**77.14**	± 8.89
8.	fi_pud	83.78	Stanford	62.38	±14.83
9.	no_bokmaal	83.68	UDPipe Future	**70.75**	± 8.92
10.	cs_cac	83.42	UDPipe Future	**71.39**	± 6.89
11.	bg_btb	83.12	UDPipe Future	**73.18**	± 7.15
12.	fr_sequoia	82.55	Stanford	70.42	± 9.04
13.	sl_ssj	82.38	Stanford	62.41	± 9.18
14.	no_nynorsk	81.86	UDPipe Future	**68.62**	± 9.45
15.	ko_kaist	81.29	HIT-SCIR	**70.18**	± 9.36
16.	ko_gsd	80.85	HIT-SCIR	63.73	±16.02
17.	fi_tdt	80.84	Stanford	**65.27**	± 9.22
18.	fa_seraji	80.83	UDPipe Future	**71.23**	± 7.77
19.	pl_sz	80.77	Stanford	64.80	± 8.49
20.	fro_srcmf	80.28	UDPipe Future	**65.19**	±16.58
21.	la_ittb	79.84	ICS PAS	**67.77**	± 8.37
22.	fi_ftb	79.65	TurkuNLP	66.11	± 8.86
23.	sv_talbanken	79.32	Stanford	**68.05**	± 8.49
24.	ro_rrt	78.68	TurkuNLP	67.43	± 7.24
25.	el_gdt	78.66	Stanford	64.29	± 8.28
26.	fr_gsd	78.44	Stanford	**69.33**	± 8.59
27.	hi_hdtb	78.30	UDPipe Future	68.48	± 5.88
28.	sr_set	77.73	UDPipe Future	67.33	± 5.96
29.	da_ddt	77.31	Stanford	65.00	± 6.89
30.	et_edt	76.97	TurkuNLP	63.59	± 8.34
31.	nl_alpino	76.52	Stanford	62.82	± 9.81
32.	en_ewt	76.33	Stanford	**66.84**	± 5.86
33.	pt_bosque	75.94	Stanford	66.22	± 6.76
34.	cs_pud	75.81	UDPipe Future	60.47	±11.36
35.	af_afribooms	75.67	UDPipe Future	**63.76**	± 7.06
36.	sk_snk	75.01	Stanford	56.82	± 8.32
37.	en_pud	74.86	Stanford	**63.05**	± 7.89
38.	nl_lassysmall	74.11	Stanford	61.95	± 9.12
39.	hr_set	73.44	Stanford	60.08	± 7.07
40.	en_gum	73.24	ICS PAS	**61.72**	± 7.69
41.	ja_gsd	72.62	HIT-SCIR	59.52	± 6.20
42.	uk_iu	72.27	UDPipe Future	55.45	± 8.08
43.	en_lines	72.25	ICS PAS	**62.35**	± 8.04
44.	eu_bdt	71.73	UDPipe Future	58.49	± 8.62
45.	gl_ctg	70.92	Stanford	57.92	±14.10
46.	ar_padt	68.54	Stanford	53.28	± 6.12
47.	it_postwita	68.50	Stanford	51.72	± 8.80
48.	id_gsd	68.36	Stanford	**61.03**	± 6.49
49.	lv_lvtb	67.89	Stanford	53.31	± 7.96
50.	hu_szeged	67.13	UDPipe Future	53.08	± 8.01
51.	zh_gsd	66.62	HIT-SCIR	50.42	± 5.87
52.	sv_lines	66.58	Stanford	**57.40**	± 7.43
53.	fr_spoken	64.67	HIT-SCIR	53.17	± 5.61
54.	he_htb	63.38	Stanford	45.22	± 4.94
55.	cu_proiel	63.31	Stanford	**50.28**	± 6.69
56.	ru_taiga	61.59	ICS PAS	37.16	± 7.53
57.	gl_treegal	60.63	UDPipe Future	**47.35**	± 5.93
58.	grc_proiel	60.27	Stanford	**47.62**	±11.82
59.	la_proiel	59.36	Stanford	**47.79**	± 6.90
60.	de_gsd	58.04	TurkuNLP	39.13	±10.35
61.	ur_udtb	57.98	TurkuNLP	**49.64**	± 4.21
62.	no_nynorsklia	57.51	ICS PAS	37.08	± 7.78
63.	sme_giella	57.47	TurkuNLP	**38.29**	±12.37
64.	got_proiel	56.45	UDPipe Future	**46.18**	± 5.36
65.	tr_imst	55.73	Stanford	45.26	± 6.15
66.	grc_perseus	54.98	HIT-SCIR	35.65	±12.31
67.	sv_pud	51.74	TurkuNLP	**39.41**	± 7.78
68.	la_perseus	49.77	ICS PAS	28.67	± 8.06
69.	vi_vtb	47.61	HIT-SCIR	**32.45**	± 7.28
70.	sl_sst	45.93	ICS PAS	**33.12**	± 5.33
71.	ga_idt	45.79	TurkuNLP	**33.70**	± 5.18
72.	ug_udt	45.78	UDPipe Future	**35.08**	± 5.96
73.	br_keb	13.91	Uppsala	1.52	± 3.34
74.	hy_armtdp	13.36	CUNI x-ling	**5.94**	± 2.92
75.	ja_modern	11.82	Uppsala	**6.45**	± 2.59
76.	hsb_ufal	9.09	LATTICE	4.66	± 2.37
77.	kk_ktb	8.93	CUNI x-ling	**5.04**	± 2.34
78.	kmr_mg	7.98	IBM NY	4.01	± 1.96
79.	th_pud	6.29	CUNI x-ling	0.42	± 1.27
80.	pcm_nsc	5.30	KParse	**3.00**	± 1.30
81.	bxr_bdt	2.98	AntNLP	1.33	± 0.72
82.	fo_oft	1.07	CUNI x-ling	0.37	± 0.21

Table 16: Treebank ranking by best parser MLAS.

	Treebank	BLEX	Best system	Avg	StDev
1.	pl_lfg	90.42	TurkuNLP	72.81	±16.96
2.	ru_syntagrus	88.65	TurkuNLP	68.57	±18.07
3.	cs_pdt	87.91	HIT-SCIR	**74.41**	±14.88
4.	cs_fictree	87.81	ICS PAS	71.10	±16.26
5.	cs_cac	86.79	TurkuNLP	**71.61**	±18.18
6.	hi_hdtb	86.74	HIT-SCIR	**75.80**	± 9.28
7.	pl_sz	86.29	TurkuNLP	67.33	±17.15
8.	no_bokmaal	85.82	UDPipe Future	**69.52**	±13.54
9.	ca_ancora	85.47	UDPipe Future	**72.60**	±12.31
10.	es_ancora	84.92	HIT-SCIR	72.10	±12.71
11.	it_isdt	84.76	ICS PAS	**75.42**	±10.72
12.	fr_sequoia	84.67	ICS PAS	70.63	±11.66
13.	no_nynorsk	84.44	TurkuNLP	67.43	±14.10
14.	la_ittb	84.37	TurkuNLP	**68.10**	±17.85
15.	bg_btb	84.31	TurkuNLP	**68.13**	±15.02
16.	fro_srcmf	84.11	UDPipe Future	**70.46**	±16.40
17.	sr_set	83.28	TurkuNLP	65.62	±17.61
18.	sl_ssj	83.23	Stanford	62.54	±17.20
19.	fi_ftb	82.44	TurkuNLP	59.66	±16.50
20.	fi_pud	82.44	TurkuNLP	52.25	±18.50
21.	sv_talbanken	81.44	TurkuNLP	**66.45**	±13.18
22.	fi_tdt	81.24	TurkuNLP	54.70	±17.25
23.	fr_gsd	81.18	HIT-SCIR	**69.61**	±10.58
24.	ro_rrt	80.97	TurkuNLP	63.53	±15.84
25.	sk_snk	80.74	TurkuNLP	58.35	±15.07
26.	pt_bosque	80.62	TurkuNLP	**68.71**	±11.27
27.	en_pud	80.53	LATTICE	64.73	±10.88
28.	cs_pud	80.53	ICS PAS	64.62	±16.03
29.	hr_set	80.50	TurkuNLP	**64.64**	±17.13
30.	fa_seraji	80.44	Stanford	**68.38**	± 7.39
31.	el_gdt	80.09	TurkuNLP	63.26	±15.60
32.	ko_kaist	79.55	TurkuNLP	57.32	±20.78
33.	et_edt	79.37	TurkuNLP	57.06	±16.14
34.	nl_alpino	79.15	HIT-SCIR	**64.29**	±10.83
35.	en_ewt	78.44	HIT-SCIR	**67.53**	± 8.47
36.	uk_iu	78.38	TurkuNLP	57.78	±15.95
37.	eu_bdt	78.15	TurkuNLP	**60.52**	±15.24
38.	da_ddt	78.07	TurkuNLP	**63.16**	±11.41
39.	sv_lines	77.01	ICS PAS	63.13	±11.72
40.	id_gsd	76.56	Stanford	62.52	± 7.89
41.	nl_lassysmall	76.54	HIT-SCIR	60.92	±11.93
42.	af_afribooms	76.44	TurkuNLP	**63.87**	± 9.62
43.	ko_gsd	76.31	TurkuNLP	54.13	±17.78
44.	en_lines	75.29	HIT-SCIR	**62.29**	± 9.27
45.	gl_ctg	75.14	Stanford	60.86	±10.82
46.	ur_udtb	73.79	TurkuNLP	**62.93**	± 6.42
47.	ja_gsd	73.79	HIT-SCIR	60.87	± 6.04
48.	en_gum	73.57	ICS PAS	**61.02**	± 8.59
49.	hu_szeged	73.17	TurkuNLP	55.42	±10.95
50.	zh_gsd	72.97	HIT-SCIR	**55.66**	± 6.26
51.	lv_lvtb	72.40	TurkuNLP	53.42	±14.56
52.	de_gsd	71.40	HIT-SCIR	**54.86**	±14.99
53.	cu_proiel	71.31	Stanford	51.27	±15.35
54.	ar_padt	70.06	Stanford	49.13	±18.98
55.	it_postwita	69.34	HIT-SCIR	**50.97**	± 8.76
56.	grc_proiel	69.03	TurkuNLP	48.58	±19.91
57.	la_proiel	67.60	TurkuNLP	**51.03**	±14.56
58.	sv_pud	66.12	TurkuNLP	50.20	±11.30
59.	fr_spoken	65.63	HIT-SCIR	**52.57**	± 7.29
60.	he_htb	65.04	Stanford	47.22	± 6.60
61.	ru_taiga	64.36	ICS PAS	39.32	±10.49
62.	gl_treegal	64.29	UDPipe Future	**49.38**	± 8.18
63.	got_proiel	63.98	Stanford	48.79	±13.77
64.	no_nynorsklia	60.98	ICS PAS	41.20	± 8.64
65.	tr_imst	60.13	TurkuNLP	**45.39**	±10.38
66.	sme_giella	60.10	TurkuNLP	35.76	±12.68
67.	grc_perseus	58.68	TurkuNLP	**36.48**	±16.03
68.	ug_udt	55.42	HIT-SCIR	**41.64**	± 8.09
69.	ga_idt	55.18	TurkuNLP	37.83	± 7.61
70.	la_perseus	52.75	ICS PAS	30.16	±11.05
71.	sl_sst	50.94	ICS PAS	**37.20**	± 6.87
72.	vi_vtb	44.02	Stanford	35.50	± 3.74
73.	pcm_nsc	26.04	CUNI x-ling	12.07	± 5.63
74.	hsb_ufal	21.09	LATTICE	11.26	± 4.97
75.	br_keb	20.70	TurkuNLP	4.19	± 4.93
76.	hy_armtdp	19.04	CUNI x-ling	**10.68**	± 4.37
77.	fo_oft	14.40	CUNI x-ling	7.32	± 3.33
78.	ja_modern	13.79	Stanford	**7.70**	± 2.86
79.	kmr_mg	13.66	LATTICE	**8.44**	± 3.11
80.	kk_ktb	11.33	CUNI x-ling	6.75	± 2.95
81.	th_pud	10.77	CUNI x-ling	0.91	± 2.11
82.	bxr_bdt	6.65	AntNLP	**3.39**	± 1.61

Table 17: Treebank ranking by best parser BLEX.

	Treebank	LAS	MLAS	Diff	Language
1.	de_gsd	70.13	39.13	31.01	German
2.	sv_pud	67.02	39.41	27.61	Swedish
3.	fo_oft	27.87	0.37	27.50	Faroese
4.	ur_udtb	75.89	49.64	26.25	Urdu
5.	ga_idt	58.37	33.70	24.66	Irish
6.	grc_perseus	59.01	35.65	23.36	Ancient Greek
7.	hsb_ufal	26.48	4.66	21.82	Upper Sorbian
8.	sk_snk	76.53	56.82	19.71	Slovak
9.	ug_udt	54.27	35.08	19.20	Uyghur
10.	ru_taiga	56.27	37.16	19.12	Russian
11.	hr_set	78.37	60.08	18.29	Croatian
12.	la_perseus	46.91	28.67	18.24	Latin
13.	grc_proiel	65.02	47.62	17.40	Ancient Greek
14.	gl_treegal	64.65	47.35	17.30	Galician
15.	uk_iu	72.47	55.45	17.01	Ukrainian
16.	hi_hdtb	85.16	68.48	16.68	Hindi
17.	pl_sz	81.47	64.80	16.67	Polish
18.	hy_armtdp	22.39	5.94	16.45	Armenian
19.	el_gdt	80.65	64.29	16.36	Greek
20.	sv_lines	73.76	57.40	16.36	Swedish
21.	kmr_mg	20.27	4.01	16.26	Kurmanji
22.	nl_alpino	77.76	62.82	14.95	Dutch
23.	gl_ctg	72.46	57.92	14.55	Galician
24.	lv_lvtb	67.76	53.31	14.45	Latvian
25.	got_proiel	60.55	46.18	14.37	Gothic
26.	pt_bosque	80.49	66.22	14.27	Portuguese
27.	ja_gsd	73.68	59.52	14.16	Japanese
28.	kk_ktb	19.11	5.04	14.07	Kazakh
29.	hu_szeged	67.05	53.08	13.96	Hungarian
30.	sl_sst	47.07	33.12	13.95	Slovenian
31.	eu_bdt	72.08	58.49	13.59	Basque
32.	he_htb	58.73	45.22	13.51	Hebrew
33.	la_proiel	61.25	47.79	13.46	Latin
34.	no_nynorsklia	50.33	37.08	13.25	Norwegian
35.	it_postwita	64.95	51.72	13.22	Italian
36.	nl_lassysmall	75.08	61.95	13.14	Dutch
37.	af_afribooms	76.61	63.76	12.84	Afrikaans
38.	sme_giella	51.10	38.29	12.82	North Sámi
39.	cs_pud	73.24	60.47	12.77	Czech
40.	sl_ssj	75.00	62.41	12.59	Slovenian
41.	sr_set	79.84	67.33	12.50	Serbian
42.	en_gum	74.20	61.72	12.48	English
43.	ja_modern	18.92	6.45	12.47	Japanese
44.	cu_proiel	62.64	50.28	12.36	Old Church Slavonic
45.	cs_fictree	82.10	69.91	12.19	Czech
46.	pl_lfg	85.89	73.73	12.17	Polish
47.	id_gsd	73.05	61.03	12.02	Indonesian
48.	br_keb	13.27	1.52	11.75	Breton
49.	fr_spoken	64.66	53.17	11.49	French
50.	en_pud	74.51	63.05	11.46	English
51.	cs_cac	82.69	71.39	11.29	Czech
52.	ar_padt	64.07	53.28	10.79	Arabic
53.	fi_ftb	76.89	66.11	10.78	Finnish
54.	it_isdt	87.61	77.14	10.47	Italian
55.	tr_imst	55.61	45.26	10.34	Turkish
56.	pcm_nsc	13.19	3.00	10.19	Naija
57.	fr_sequoia	80.55	70.42	10.13	French
58.	bxr_bdt	11.45	1.33	10.12	Buryat
59.	fr_gsd	79.43	69.33	10.10	French
60.	da_ddt	75.02	65.00	10.02	Danish
61.	no_nynorsk	78.55	68.62	9.93	Norwegian
62.	en_lines	72.28	62.35	9.93	English
63.	zh_gsd	60.32	50.42	9.90	Chinese
64.	sv_talbanken	77.71	68.05	9.66	Swedish
65.	bg_btb	82.52	73.18	9.34	Bulgarian
66.	la_ittb	77.00	67.77	9.23	Latin
67.	fro_srcmf	74.38	65.19	9.18	Old French
68.	en_ewt	75.99	66.84	9.15	English
69.	no_bokmaal	79.80	70.75	9.05	Norwegian
70.	ca_ancora	83.61	74.62	8.99	Catalan
71.	cs_pdt	82.18	73.61	8.57	Czech
72.	et_edt	72.08	63.59	8.50	Estonian
73.	ro_rrt	75.77	67.43	8.33	Romanian
74.	fi_tdt	73.55	65.27	8.28	Finnish
75.	es_ancora	82.84	74.61	8.23	Spanish
76.	ko_gsd	71.88	63.73	8.15	Korean
77.	ru_syntagrus	79.68	71.63	8.05	Russian
78.	vi_vtb	40.40	32.45	7.95	Vietnamese
79.	fa_seraji	78.71	71.23	7.48	Persian
80.	ko_kaist	77.10	70.18	6.92	Korean
81.	fi_pud	68.87	62.38	6.49	Finnish
82.	th_pud	1.38	0.42	0.96	Thai

Table 18: Treebank ranking by difference between average parser LAS and MLAS.

	Treebank	Best	Best system	Avg	StDev
70.	bxr_bdt	99.24	IBM NY	88.64	± 8.09
71.	fi_pud	99.69	Uppsala	88.13	±10.81
72.	zh_gsd	96.71	HIT-SCIR	86.91	± 3.83
73.	fo_oft	99.47	CUNI x-ling	86.76	±10.68
74.	ar_padt	96.81	Stanford	86.62	± 7.00
75.	kmr_mg	96.97	Uppsala	86.61	± 7.16
76.	kk_ktb	97.40	Uppsala	85.55	± 7.45
77.	br_keb	92.45	TurkuNLP	83.76	± 7.37
78.	he_htb	93.98	Stanford	82.45	± 3.80
79.	vi_vtb	93.46	HIT-SCIR	81.71	± 3.73
80.	pcm_nsc	99.71	CEA LIST	79.94	±10.69
81.	ja_modern	75.69	HIT-SCIR	59.40	± 7.70
82.	th_pud	69.93	Uppsala	17.16	±20.57

Table 19: Treebanks with most difficult word segmentation (by average parser F_1).

	Treebank	Best	Best system	Avg	StDev
73.	grc_proiel	51.84	HIT-SCIR	42.46	± 7.33
74.	cu_proiel	48.67	Stanford	35.54	± 4.02
75.	la_proiel	39.61	Stanford	33.40	± 5.39
76.	got_proiel	38.23	Stanford	27.22	± 4.47
77.	it_postwita	65.90	Stanford	25.25	±14.30
78.	sl_sst	24.43	NLP-Cube	20.92	± 4.70
79.	fr_spoken	24.17	Stanford	20.43	± 2.89
80.	th_pud	12.37	TurkuNLP	1.75	± 3.68
81.	pcm_nsc	0.93	Stanford	0.06	± 0.19
82.	ja_modern	0.23	Stanford	0.01	± 0.04

Table 20: Treebanks with most difficult sentence segmentation (by average parser F_1).

7 Analysis of Submitted Systems

Table 21 gives an overview of 24 of the systems evaluated in the shared task. The overview is based on a post-evaluation questionnaire to which 24 of 25 teams responded. Systems are ordered alphabetically by name and their LAS rank is indicated in the second column.

Looking first at word and sentence segmentation, we see that, while a clear majority of systems (19/24) rely on the baseline system for segmentation, slightly more than half (13/24) have developed their own segmenter, or tuned the baseline segmenter, for at least a subset of languages. This is a development from 2017, where only 7 out of 29 systems used anything other than the baseline segmenter.

When it comes to morphological analysis, including universal POS tags, features and lemmas, all systems this year include some such component, and only 6 systems rely entirely on the base-

System	R	Segment	Morph	Syntax	WEmb	Additional Data	MultiLing
AntNLP	9	Base	Base	Single-G	FB	None	Own$_S$
ArmParser	25	Base	Own	Single	FB	None	None
BOUN	21	Base	Base	Single-T	Base	None	None
CEA LIST	6	Base	B$_L$/Own	Single-G/T	B/FB	OPUS/Wikt	Own$_L$
CUNI x-ling	20	B/Own	B/Own	Single/Ens	FB/None	O/UM/WALS/Wiki	Own$_{L,S}$
Fudan	17	Base	Base	Ensemble	None	None	Own$_{L,S}$
HIT-SCIR	1	B/Own	Base	Ensemble	B/FB/Crawl	None	Own$_{L,S}$
HUJI	24	Base	Base	Single-T	FB	None	Own$_L$
IBM NY	13	B/Own	B/Joint	Ensemble-T	B/FB	Wiki	Own$_{L,S}$
ICS PAS	3	Base	Own	Single-G	FB/None	None	None
KParse	16	B/Own	Own	Single	Other	None	Own$_L$
LATTICE	3	Base	Own$_U$	Single-G/Ens	B/FB/Crawl	OPUS/Wiki	Own$_{L,S}$
LeisureX	15	Base	Own	Single	Base	None	Own$_L$
NLP-Cube	9	Own	Own	Single	FB	None	Own$_L$
ONLP lab	22	Base	Base	Single-T	None	UML	None
ParisNLP	11	B/Own	B/Own	Single-G	FB	UML	Own$_L$
Phoenix	19	Own	Own$_U$	Single	Train	None	Own$_L$
SLT-Interactions	12	B/Own	Own	Single	Crawl	None	Own$_L$
SParse	26	B/Own	Own	Single-G	Crawl	None	Own$_L$
Stanford	7	Own	Own	Single-G	B/FB	None	None
TurkuNLP	2	B/Own	Own	Single-G	B/FB	OPUS/Aper	Own$_L$
UDPipe Future	3	Own	Joint	Single-G	B/FB	None	None
UniMelb	14	Base	Joint	Single	Base	None	Base
Uppsala	7	Own	Own$_{U,F}$	Single-T	B/FB/Wiki	OPUS/Wiki/Aper	Own$_{L,S}$

Table 21: Classification of participating systems. **R** = LAS ranking. **Segment** = word/sentence segmentation. **Morph** = morphological analysis, including universal POS tags [U], features [F] and lemmas [L], with subscripts for subsets [Joint = morphological component trained jointly with syntactic parser]. **Syntax** = syntactic parsing [Single = single parser; Ensemble (or Ens) = parser ensemble; G = graph-based; T = transition-based]. **WEmb** = pre-trained word embeddings [FB = Facebook; Crawl = trained on web crawl data provided by the organizers; Wiki = trained on Wikipedia data; Train = trained on treebank training data]. **Additional Data** = data used in addition to treebank training sets [OPUS (or O) = OPUS, Aper = Apertium morphological analysers, Wikt = Wiktionary, Wiki = Wikipedia, UM = UniMorph, UML = Universal Morphological Lattices, WALS = World Atlas of Language Structures]. **MultiLing** = multilingal models used for low-resource (L) or small (S) languages. In all columns, Base (or B) refers to the Baseline UDPipe system or the baseline word embeddings provided by the organizers, while None means that there is no corresponding component in the system.

line UDPipe system. This is again quite different from 2017, where more than half the systems either just relied on the baseline tagger (13 systems) or did not predict any morphology at all (3 systems). We take this to be primarily a reflection of the fact that two out of three official metrics included (some) morphological analysis this year, although 3 systems did not predict the lemmas required for the BLEX metric (and 2 systems only predicted universal POS tags, no features). As far as we can tell from the questionnaire responses, only 3 systems used a model where morphology and syntax were predicted jointly.[14]

For syntactic parsing, most teams (19) use a single parsing model, while 5 teams, including the winning HIT-SCIR system, build ensemble models, either for all languages or a subset of them. When it comes to the type of parsing model, we observe that graph-based models are more popular than transition-based models this year, while the opposite was true in 2017. We hypothesize that

[14]The ONLP lab system also has a joint model but in the end used the baseline morphology as it gave better results.

this is due to the superior performance of the Stanford graph-based parser in last year's shared task, and many of the high-performing systems this year either incorporate that parser or a reimplementation of it.[15]

The majority of parsers make use of pre-trained word embeddings. Most popular are the Facebook embeddings, which are used by 17 systems, followed by the baseline embeddings provided by the organizers (11), and embeddings trained on web crawl data (4).[16] When it comes to additional data, over and above the treebank training sets and pre-trained word embeddings, the most striking observation is that a majority of systems (16) did not use any at all. Those that did primarily used OPUS (5), Wikipedia dumps (3), Apertium morphological analyzers (2), and Universal Morphological Lattices (2). The CUNI x-ling system, which focused on low-resource languages, also exploited UniMorph and WALS (in addition to OPUS and Wikipedia).

Finally, we note that a majority of systems make use of models trained on multiple languages to improve parsing for languages with little or no training data. According to the questionnaire responses, 15 systems use multilingual models for the languages classified as "low-resource", while 7 systems use them for the languages classified as "small".[17] Only one system relied on the baseline delexicalized parser trained on data from all languages.

8 Conclusion

The CoNLL 2018 Shared Task on UD parsing, the second in the series, was novel in several respects. Besides using cross-linguistically consistent linguistic representations, emphasizing end-to-end processing of text, and in using a multiply parallel test set, as in 2017, it was unusual also in featuring an unprecedented number of languages and treebanks and in integrating cross-lingual learning for resource-poor languages. Compared to the first edition of the task in 2017, this year several languages were provided with little-to-no resources, whereas in 2017, predicted morphology trained on the language in question was available for all of the languages. The most extreme example of these is Thai, where the only accessible resource was the Facebook Research Thai embeddings model and the OPUS parallel corpora. This year's task also introduced two additional metrics that take into account morphology and lemmatization. This encouraged the development of truly end-to-end full parsers, producing complete parses including morphological features and lemmas in addition to the syntactic tree. This also aimed to improve the utility of the systems developed in the shared task for later downstream applications. For most UD languages, these parsers represent a new state of the art for end-to-end dependency parsing.

The analysis of the shared task results has so far only scratched the surface, and we refer to the system description papers for more in-depth analysis of individual systems and their performance. For many previous CoNLL shared tasks, the task itself has only been the starting point of a long and fruitful research strand, enabled by the resources created for the task. We hope and believe that the 2017 and 2018 UD parsing tasks will join this tradition.

[15]This is true of at least 3 of the 5 best performing systems.

[16]The baseline embeddings were the same as in 2017 and therefore did not cover new languages, which may partly explain the greater popularity of the Facebook embeddings this year.

[17]We know that some teams used them also for clusters involving high-resource languages, but we have no detailed statistics on this usage.

Acknowledgments

We are grateful to all the contributors to Universal Dependencies; without their effort a task like this simply wouldn't be possible.

The work described herein, including data preparation for the *CoNLL 2018 UD Shared Task*, has been supported by the following grants and projects: OP PPR No. CZ.07.1.02/0.0/0.0/16_023/0000108 of the City of Prague, "CRACKER," H2020 Project No. 645357 of the European Commission; "MANYLA," Grant No. GA15-10472S of the Grant Agency of the Czech Republic; FIN-CLARIN; Grant No. 2016-01817 of the Swedish Research Council; and the LINDAT/CLARIN research infrastructure project funded by the Ministry of Education, Youth and Sports of the Czech Republic, Project. No. LM2015071. The data for the *CoNLL 2018 UD Shared Task* are available also via the LINDAT/CLARIN repository.

References

Gor Arakelyan, Karen Hambardzumyan, and Hrant Khachatrian. 2018. Towards JointUD: Part-of-speech tagging and lemmatization using recurrent neural networks. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*. Association for Computational Linguistics.

Riyaz Ahmad Bhat, Irshad Ahmad Bhat, and Srinivas Bangalore. 2018. The SLT-Interactions parsing system at the CoNLL 2018 shared task. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*. Association for Computational Linguistics.

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

Tiberiu Boroș, Stefan Daniel Dumitrescu, and Ruxandra Burtica. 2018. NLP-Cube: End-to-end raw text processing with neural networks. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*. Association for Computational Linguistics.

Sabine Buchholz and Erwin Marsi. 2006. CoNLL-X shared task on multilingual dependency parsing. In *Proceedings of the 10th Conference on Computational Natural Language Learning (CoNLL-X)*. Association for Computational Linguistics, pages 149–164. http://anthology.aclweb.org/W/W06/W06-29.pdf#page=165.

Wanxiang Che, Yijia Liu, Yuxuan Wang, Bo Zheng, and Ting Liu. 2018. Towards better UD parsing: Deep contextualized word embeddings, ensemble, and treebank concatenation. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*. Association for Computational Linguistics.

Danlu Chen, Mengxiao Lin, Zhifeng Hu, and Xipeng Qiu. 2018. A simple yet effective joint training method for cross-lingual universal dependency parsing. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*. Association for Computational Linguistics.

Elie Duthoo and Olivier Mesnard. 2018. CEA LIST : Processing low-resource languages for CoNLL 2018. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*. Association for Computational Linguistics.

Murhaf Fares, Stephan Oepen, Lilja Øvrelid, Jari Björne, and Richard Johansson. 2018. The 2018 shared task on extrinsic parser evaluation: On the downstream utility of English universal dependency parsers. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*. Association for Computational Linguistics, Brussels, Belgium.

Filip Ginter, Jan Hajič, Juhani Luotolahti, Milan Straka, and Daniel Zeman. 2017. CoNLL 2017 shared task - automatically annotated raw texts and word embeddings. LINDAT/CLARIN digital library at the Institute of Formal and Applied Linguistics, Charles University. http://hdl.handle.net/11234/1-1989.

Allan Hanbury, Henning Müller, Krisztian Balog, Torben Brodt, Gordon V. Cormack, Ivan Eggel, Tim Gollub, Frank Hopfgartner, Jayashree Kalpathy-Cramer, Noriko Kando, Anastasia Krithara, Jimmy Lin, Simon Mercer, and Martin Potthast. 2015. Evaluation-as-a-Service: Overview and Outlook. *ArXiv e-prints* http://arxiv.org/abs/1512.07454.

Daniel Hershcovich, Omri Abend, and Ari Rappoport. 2018. Universal dependency parsing with a general transition-based DAG parser. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*. Association for Computational Linguistics.

Ganesh Jawahar, Benjamin Muller, Amal Fethi, Louis Martin, Éric de La Clergerie, Benoît Sagot, and Djamé Seddah. 2018. ELMoLex: Connecting ELMo and lexicon features for dependency parsing. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*. Association for Computational Linguistics.

Tao Ji, Yufang Liu, Yijun Wang, Yuanbin Wu, and Man Lan. 2018. AntNLP at CoNLL 2018 shared task: A graph-based parser for universal dependency parsing. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*. Association for Computational Linguistics.

Jenna Kanerva, Filip Ginter, Niko Miekka, Akseli Leino, and Tapio Salakoski. 2018. Turku neural parser pipeline: An end-to-end system for the CoNLL 2018 shared task. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*. Association for Computational Linguistics.

Ömer Kırnap, Erenay Dayanık, and Deniz Yuret. 2018. Tree-stack LSTM in transition based dependency parsing. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*. Association for Computational Linguistics.

Zuchao Li, Shexia He, Zhuosheng Zhang, and Hai Zhao. 2018. Joint learning of pos and dependencies for multilingual universal dependency parsing. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*. Association for Computational Linguistics.

KyungTae Lim, Cheoneum Park, Changki Lee, and Thierry Poibeau. 2018. SEx BiST: A multi-source trainable parser with deep contextualized lexical representations. In *Proceedings of the CoNLL 2018*

Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies. Association for Computational Linguistics.

Dat Quoc Nguyen and Karin Verspoor. 2018. An improved neural network model for joint pos tagging and dependency parsing. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*. Association for Computational Linguistics.

Joakim Nivre, Mitchell Abrams, Željko Agić, Lars Ahrenberg, Lene Antonsen, Maria Jesus Aranzabe, Gashaw Arutie, Masayuki Asahara, Luma Ateyah, Mohammed Attia, Aitziber Atutxa, Liesbeth Augustinus, Elena Badmaeva, Miguel Ballesteros, Esha Banerjee, Sebastian Bank, Verginica Barbu Mititelu, John Bauer, Sandra Bellato, Kepa Bengoetxea, Riyaz Ahmad Bhat, Erica Biagetti, Eckhard Bick, Rogier Blokland, Victoria Bobicev, Carl Börstell, Cristina Bosco, Gosse Bouma, Sam Bowman, Adriane Boyd, Aljoscha Burchardt, Marie Candito, Bernard Caron, Gauthier Caron, Gülşen Cebiroğlu Eryiğit, Giuseppe G. A. Celano, Savas Cetin, Fabricio Chalub, Jinho Choi, Yongseok Cho, Jayeol Chun, Silvie Cinková, Aurélie Collomb, Çağrı Çöltekin, Miriam Connor, Marine Courtin, Elizabeth Davidson, Marie-Catherine de Marneffe, Valeria de Paiva, Arantza Diaz de Ilarraza, Carly Dickerson, Peter Dirix, Kaja Dobrovoljc, Timothy Dozat, Kira Droganova, Puneet Dwivedi, Marhaba Eli, Ali Elkahky, Binyam Ephrem, Tomaž Erjavec, Aline Etienne, Richárd Farkas, Hector Fernandez Alcalde, Jennifer Foster, Cláudia Freitas, Katarína Gajdošová, Daniel Galbraith, Marcos Garcia, Moa Gärdenfors, Kim Gerdes, Filip Ginter, Iakes Goenaga, Koldo Gojenola, Memduh Gökırmak, Yoav Goldberg, Xavier Gómez Guinovart, Berta Gonzáles Saavedra, Matias Grioni, Normunds Grūzītis, Bruno Guillaume, Céline Guillot-Barbance, Nizar Habash, Jan Hajič, Jan Hajič jr., Linh Hà Mỹ, Na-Rae Han, Kim Harris, Dag Haug, Barbora Hladká, Jaroslava Hlaváčová, Florinel Hociung, Petter Hohle, Jena Hwang, Radu Ion, Elena Irimia, Tomáš Jelínek, Anders Johannsen, Fredrik Jørgensen, Hüner Kaşıkara, Sylvain Kahane, Hiroshi Kanayama, Jenna Kanerva, Tolga Kayadelen, Václava Kettnerová, Jesse Kirchner, Natalia Kotsyba, Simon Krek, Sookyoung Kwak, Veronika Laippala, Lorenzo Lambertino, Tatiana Lando, Septina Dian Larasati, Alexei Lavrentiev, John Lee, Phương Lê Hồng, Alessandro Lenci, Saran Lertpradit, Herman Leung, Cheuk Ying Li, Josie Li, Keying Li, KyungTae Lim, Nikola Ljubešić, Olga Loginova, Olga Lyashevskaya, Teresa Lynn, Vivien Macketanz, Aibek Makazhanov, Michael Mandl, Christopher Manning, Ruli Manurung, Cătălina Mărănduc, David Mareček, Katrin Marheinecke, Héctor Martínez Alonso, André Martins, Jan Mašek, Yuji Matsumoto, Ryan McDonald, Gustavo Mendonça, Niko Miekka, Anna Missilä, Cătălin Mititelu, Yusuke Miyao, Simonetta Montemagni, Amir More, Laura Moreno Romero, Shinsuke Mori, Bjartur Mortensen, Bohdan Moskalevskyi, Kadri Muischnek, Yugo Murawaki, Kaili Müürisep, Pinkey Nainwani, Juan Ignacio Navarro Horñiacek, Anna Nedoluzhko, Gunta Nešpore-Bērzkalne, Lương Nguyễn Thị, Huyền Nguyễn Thị Minh, Vitaly Nikolaev, Rattima Nitisaroj, Hanna Nurmi, Stina Ojala, Adédayọ̀ Olúòkun, Mai Omura, Petya Osenova, Robert Östling, Lilja Øvrelid, Niko Partanen, Elena Pascual, Marco Passarotti, Agnieszka Patejuk, Siyao Peng, Cenel-Augusto Perez, Guy Perrier, Slav Petrov, Jussi Piitulainen, Emily Pitler, Barbara Plank, Thierry Poibeau, Martin Popel, Lauma Pretkalniņa, Sophie Prévost, Prokopis Prokopidis, Adam Przepiórkowski, Tiina Puolakainen, Sampo Pyysalo, Andriela Rääbis, Alexandre Rademaker, Loganathan Ramasamy, Taraka Rama, Carlos Ramisch, Vinit Ravishankar, Livy Real, Siva Reddy, Georg Rehm, Michael Rießler, Larissa Rinaldi, Laura Rituma, Luisa Rocha, Mykhailo Romanenko, Rudolf Rosa, Davide Rovati, Valentin Roșca, Olga Rudina, Shoval Sadde, Shadi Saleh, Tanja Samardžić, Stephanie Samson, Manuela Sanguinetti, Baiba Saulīte, Yanin Sawanakunanon, Nathan Schneider, Sebastian Schuster, Djamé Seddah, Wolfgang Seeker, Mojgan Seraji, Mo Shen, Atsuko Shimada, Muh Shohibussirri, Dmitry Sichinava, Natalia Silveira, Maria Simi, Radu Simionescu, Katalin Simkó, Mária Šimková, Kiril Simov, Aaron Smith, Isabela Soares-Bastos, Antonio Stella, Milan Straka, Jana Strnadová, Alane Suhr, Umut Sulubacak, Zsolt Szántó, Dima Taji, Yuta Takahashi, Takaaki Tanaka, Isabelle Tellier, Trond Trosterud, Anna Trukhina, Reut Tsarfaty, Francis Tyers, Sumire Uematsu, Zdeňka Urešová, Larraitz Uria, Hans Uszkoreit, Sowmya Vajjala, Daniel van Niekerk, Gertjan van Noord, Viktor Varga, Veronika Vincze, Lars Wallin, Jonathan North Washington, Seyi Williams, Mats Wirén, Tsegay Woldemariam, Tak-sum Wong, Chunxiao Yan, Marat M. Yavrumyan, Zhuoran Yu, Zdeněk Žabokrtský, Amir Zeldes, Daniel Zeman, Manying Zhang, and Hanzhi Zhu. 2018. Universal dependencies 2.2. LINDAT/CLARIN digital library at the Institute of Formal and Applied Linguistics (ÚFAL), Faculty of Mathematics and Physics, Charles University. http://hdl.handle.net/11234/1-2837.

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. http://www.lrec-conf.org/proceedings/lrec2016/summaries/348.html.

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, Sandra Kübler, Ryan McDonald, Jens Nilsson, Sebastian Riedel, and Deniz Yuret. 2007. The CoNLL 2007 shared task on dependency parsing. In *Proceedings of the CoNLL Shared Task Session of EMNLP-CoNLL 2007*. Association for Computational Linguistics, pages 915–932. http://www.aclweb.org/anthology/D/D07/D07-1.pdf#page=949.

Berkay Furkan Önder, Can Gümeli, and Deniz Yuret. 2018. SParse: Koç University graph-based parsing system for the CoNLL 2018 shared task. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*. Association for Computational Linguistics.

Şaziye Betül Özateş, Arzucan Özgür, Tunga Güngör, and Balkız Öztürk. 2018. A morphology-based representation model for LSTM-based dependency parsing of agglutinative languages. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*. Association for Computational Linguistics.

Martin Popel, Zdeněk Žabokrtský, and Martin Vojtek. 2017. Udapi: Universal API for universal dependencies. In *NoDaLiDa 2017 Workshop on Universal Dependencies*. Göteborgs universitet, Göteborg, Sweden, pages 96–101. http://aclweb.org/anthology/W/W17/W17-0412.pdf.

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.

Peng Qi, Timothy Dozat, Yuhao Zhang, and Christopher D. Manning. 2018. Universal dependency parsing from scratch. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*. Association for Computational Linguistics.

Rudolf Rosa and David Mareček. 2018. CUNI x-ling: Parsing under-resourced languages in CoNLL 2018 UD shared task. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*. Association for Computational Linguistics.

Piotr Rybak and Alina Wróblewska. 2018. Semi-supervised neural system for tagging, parsing and lematization. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*. Association for Computational Linguistics.

Amit Seker, Amir More, and Reut Tsarfaty. 2018. Universal morpho-syntactic parsing and the contribution of lexica: Analyzing the ONLP submission to the CoNLL 2018 shared task. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*. Association for Computational Linguistics.

Aaron Smith, Bernd Bohnet, Miryam de Lhoneux, Joakim Nivre, Yan Shao, and Sara Stymne. 2018. 82 treebanks, 34 models: Universal dependency parsing with multi-treebank models. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*. Association for Computational Linguistics.

Milan Straka. 2018. UDPipe 2.0 prototype at CoNLL 2018 UD shared task. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*. Association for Computational Linguistics.

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

Milan Straka 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.

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.

Hui Wan, Tahira Naseem, Young-Suk Lee, Vittorio Castelli, and Miguel Ballesteros. 2018. IBM Research at the CoNLL 2018 shared task on multilingual parsing. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*. Association for Computational Linguistics.

Yingting Wu, Hai Zhao, and Jia-Jun Tong. 2018. Multilingual universal dependency parsing from raw text with low-resource language enhancement. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*. Association for Computational Linguistics.

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, Héctor Martínez Alonso, Çağrı Çöltekin, Umut Sulubacak, 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, Vancouver, Canada, pages 1–19. http://www.aclweb.org/anthology/K17-3001.