NLPre: a revised approach towards language-centric benchmarking of Natural Language Preprocessing systems

Martyna Wiącek, Piotr Rybak, Łukasz Pszenny, Alina Wróblewska
Institute of Computer Science, Polish Academy of Sciences
ul. Jana Kazimierza 5, 01-248 Warsaw, Poland
{m.wiacek, p.rybak, l.pszenny, alina}@ipipan.waw.pl

Abstract

With the advancements of transformer-based architectures, we observe the rise of natural language preprocessing (NLPre) tools capable of solving preliminary NLP tasks (e.g. tokenisation, part-of-speech tagging, dependency parsing, or morphological analysis) without any external linguistic guidance. It is arduous to compare novel solutions to well-entrenched preprocessing toolkits, relying on rule-based morphological analysers or dictionaries. Aware of the shortcomings of existing NLPre evaluation approaches, we investigate a novel method of reliable and fair evaluation and performance reporting. Inspired by the GLUE benchmark, the proposed language-centric benchmarking system enables comprehensive ongoing evaluation of multiple NLPre tools, while credibly tracking their performance. The prototype application is configured for Polish and integrated with the thoroughly assembled NLPre-PL benchmark. Based on this benchmark, we conduct an extensive evaluation of a variety of Polish NLPre systems. To facilitate the construction of benchmarking environments for other languages, e.g. NLPre-GA for Irish or NLPre-ZH for Chinese, we ensure full customization of the publicly released source code of the benchmarking system. The links to all the resources (deployed platforms, source code, trained models, datasets etc.) can be found on the project website: https://sites.google.com/view/nlpre-benchmark.

Keywords: benchmarking, leaderboard, segmentation, POS tagging, dependency parsing, Polish

1. Introduction and related works

Morphosyntactic features predicted by part-of-speech (POS) taggers and dependency parsers underlie various downstream tasks, including but not limited to sentiment analysis (Sun et al., 2019), relation extraction (Zhang et al., 2018; Vashishth et al., 2018; Guo et al., 2019), semantic role labelling (Wang et al., 2019; Kasai et al., 2019), question answering (Khashabi et al., 2018), or machine translation (Chen et al., 2017; Zhang et al., 2019). These underlying tasks may therefore be referred to as *natural language preprocessing* (NLPre) tasks, as they precede the advanced NLP tasks. Since the quality of morphosyntactic predictions has a crucial impact on the performance of downstream tasks (Sachan et al., 2021), it is prudent to employ the best existing NLPre tools to predict the proper linguistic features. We are equipped with various NLPre methods, ranging from rule-based tools with hand-crafted grammars (e.g. Crouch et al., 2011), through statistical systems (e.g. Nivre, 2009; McDonald et al., 2005; Straka et al., 2016), neural systems supported by pre-trained language models (e.g. Qi et al., 2020; Nguyen et al., 2021a) to large language models (LLM Ouyang et al., 2022).

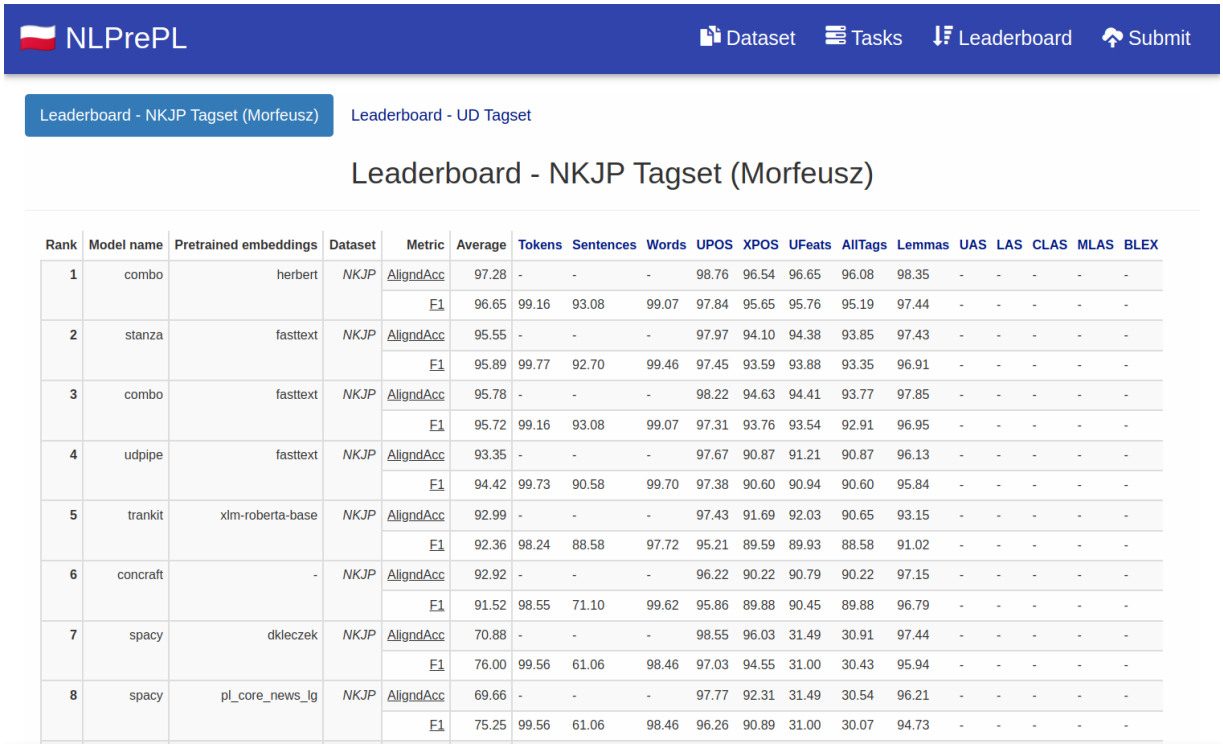
In the context of intrinsically evaluating NLPre tools and reporting their performance, a variety of approaches have been proposed, e.g. shared task, performance table, and progress repository. The main goal of a *shared task* is to comprehensively evaluate participating systems on the released datasets using the carefully defined evaluation methodology. Numerous NLPre shared tasks have been organised so far (e.g. Buchholz and Marsi, 2006; Seddah et al., 2013; Zeman et al., 2017, 2018), and they undoubtedly boosted the development of NLPre. While widely favoured, shared tasks are questionable as a complete and up-to-date source of knowledge about NLPre progress. First, they scrutinise only solutions propounded in the current contest and do not include systems participating in the previous editions or possible future ones. Second, as shared tasks are organised sporadically, their results are not revised and may quickly become outdated. Certainly, the datasets released for shared tasks can be reused in experiments involving novel tools. The results of such experiments can be reported in independent scientific publications. Nonetheless, these publications are widely scattered, lacking a centralised platform for systematically tracking the ongoing NLPre progress with respect to a particular language.

The results of a new or upgraded NLPre tool are typically reported in *performance tables* (e.g. Stanza[1] or Trankit[2]). Such tables provide information about the quality of the tool in preprocessing a set of languages. The performance tables, however, often lack comparison with other systems trained for these particular languages. Additionally, as NL-

[1] https://stanfordnlp.github.io/stanza/performance.html (UD v2.8)

[2] https://trankit.readthedocs.io/en/latest/performance.html#universal-dependencies-v2-5 (UD v2.5)

LREC-COLING 2024, pages 12271–12287
20-25 May, 2024.



Rank	Model name	Pretrained embeddings	Dataset	Metric	Average	Tokens	Sentences	Words	UPOS	XPOS	UFeats	AllTags	Lemmas	UAS	LAS	CLAS	MLAS	BLEX
1	combo	herbert	*NKJP*	AligndAcc	97.28	-	-	-	98.76	96.54	96.65	96.08	98.35	-	-	-	-	-
				F1	96.65	99.16	93.08	99.07	97.84	95.65	95.76	95.19	97.44	-	-	-	-	-
2	stanza	fasttext	*NKJP*	AligndAcc	95.55	-	-	-	97.97	94.10	94.38	93.85	97.43	-	-	-	-	-
				F1	95.89	99.77	92.70	99.46	97.45	93.59	93.88	93.35	96.91	-	-	-	-	-
3	combo	fasttext	*NKJP*	AligndAcc	95.78	-	-	-	98.22	94.63	94.41	93.77	97.85	-	-	-	-	-
				F1	95.72	99.16	93.08	99.07	97.31	93.76	93.54	92.91	96.95	-	-	-	-	-
4	udpipe	fasttext	*NKJP*	AligndAcc	93.35	-	-	-	97.67	90.87	91.21	90.87	96.13	-	-	-	-	-
				F1	94.42	99.73	90.58	99.70	97.38	90.60	90.94	90.60	95.84	-	-	-	-	-
5	trankit	xlm-roberta-base	*NKJP*	AligndAcc	92.99	-	-	-	97.43	91.69	92.03	90.65	93.15	-	-	-	-	-
				F1	92.36	98.24	88.58	97.72	95.21	89.59	89.93	88.58	91.02	-	-	-	-	-
6	concraft	-	*NKJP*	AligndAcc	92.92	-	-	-	96.22	90.22	90.79	90.22	97.15	-	-	-	-	-
				F1	91.52	98.55	71.10	99.62	95.86	89.88	90.45	89.88	96.79	-	-	-	-	-
7	spacy	dkleczek	*NKJP*	AligndAcc	70.88	-	-	-	98.55	96.03	31.49	30.91	97.44	-	-	-	-	-
				F1	76.00	99.56	61.06	98.46	97.03	94.55	31.00	30.43	95.94	-	-	-	-	-
8	spacy	pl_core_news_lg	*NKJP*	AligndAcc	69.66	-	-	-	97.77	92.31	31.49	30.54	96.21	-	-	-	-	-
				F1	75.25	99.56	61.06	98.46	96.26	90.89	31.00	30.07	94.73	-	-	-	-	-

Figure 1: Screenshot of the NLPre-PL leaderboard.

Pre systems may be trained on different dataset releases (e.g. of Universal Dependencies), comparing their performance tables is not conclusive.

Information about trends and progress in NLP research is usually collected in public repositories such as *Papers with Code*[3] or *NLP-progress*[4]. These repositories contain a repertoire of datasets for common NLP tasks, e.g. dependency parsing and POS tagging, and rankings of models trained and tested on these datasets. They are open to contributing new datasets and results, which, to ensure their credibility, originate from published and linked scientific papers. However, cutting-edge yet unpublished results of a new or upgraded NLPre system are not eligible to report. NLPre tasks are accompanied by datasets mostly in English, raising the problem of language unrepresentation of the repositories. Last but not least, the Papers with Code repository is prone to abuse. After logging in, one can add new results and link them with irrelevant papers as well as edit existing results. The fraudulent results are publicised immediately.

Despite yielding valuable information about the progress in NLPre, the mentioned evaluation approaches also reveal shortcomings, e.g. outdated and incomplete outcomes, lack of cross-system comparison, disregarding some systems, risk of result manipulation and absence of a language-centring perspective.

[3] https://paperswithcode.com
[4] http://nlpprogress.com

Following standard procedures in NLP research, we propose to robustly and fairly evaluate NLPre tools using the benchmarking method that allows for the evaluation of NLP models' performance and progress. NLP benchmarks are coupled with leaderboards that report and update model performance on the benchmark tasks, e.g. GLUE (Wang et al., 2018), XTREME (Hu et al., 2020), GEM (Gehrmann et al., 2021). The conventional benchmarking approach may be dynamically enhanced, exemplified by the *Dynabench* platform (Kiela et al., 2021), which enables users to augment the benchmark data by inputting custom examples. This human-and-model-in-the-loop benchmarking scenario appears promising for NLU tasks. Nevertheless, it may not be effective in the case of NLPre, as annotating credible examples of syntactic trees or morphological features requires expert knowledge. Finding multiple experts among casual users can be a serious obstacle, we thus implement our system in tune with the standard benchmarking method.

To our knowledge, benchmarking hasn't been used to rank NLPre systems, even if it is valuable and desired by the community creating treebanks or designing advanced NLP pipelines. Our NLPre benchmarking approach fills this gap. The proposed online benchmarking system automatically assesses submitted predictions of NLPre systems and publishes their performance ranking on a public scoreboard (see Section 2.2). The system is language-centric and tagset-agnostic, enables comprehensive and credible evaluation and consti-

tutes an up-to-date source of information on NLPre progress for a particular language. Unlike similar platforms, e.g. Codalab (Pavao et al., 2022), the NLPre benchmarking system is fully configurable and easy to set up, allowing users to establish an evaluation environment for any language. Additionally, it can be self-hosted, making it convenient for developers and researchers working with a particular language to have it accessible on a local server.

To justify the use of the benchmarking technique for NLPre tasks, we conduct empirical research in a challenging scenario with Polish as an example language. In the case of Polish, one dominant hurdle arises – the discrepancies between different tagsets, annotation schemes and datasets utilised for training disparate systems preclude their direct comparison. We thus standardise the training and evaluation of NLPre systems on a new performance benchmark for Polish, hereafter NLPre-PL (see Section 3). It consists of a predefined set of NLPre tasks and reformulated versions of existing Polish datasets. Section 4 outlines our robust and reliable evaluation of the selected NLPre systems on the NLPre-PL benchmark. According to our knowledge, no evaluation experiments have been carried out in Polish to compare the performance of off-the-shelf LLMs, neural NLPre systems and established tagging disambiguators due to the lack of a coherent evaluation environment.

This work makes a tripartite contribution encompassing novelty, research, and development underpinned by an open-source ethos. (1) We propose a novel language-oriented benchmarking approach to evaluate and rank NLPre systems. (2) We conduct a scientific evaluation of the proposed approach in the non-trivial Polish language scenario on the assembled NLPre-PL benchmark. (3) We publish online benchmarking platforms for three distinct languages: Polish[5], Chinese[6], and Irish[7], and release the benchmarking system's source code as open-source.

2. NLPre benchmarking

2.1. Research concept

In this study, we introduce a novel adaptation of the benchmarking approach to NLPre. The primary objective is to establish an automated and credible method for evaluating NLPre systems against a provided benchmark and continuously updating their performance ranking on a publicly accessible scoreboard. More specifically, predictions for the benchmark test sets output by NLPre systems and submitted to the benchmarking system are automatically compared against the publicly undisclosed reference dataset. This method effectively prevents result manipulation and ensures fairness of the final assessment. The second important methodological assumption is to enable the ongoing evaluation of new or upgraded NLPre systems to guarantee up-to-date and complete ranking. Consequently, the leaderboard can serve as a reliable point of reference for NLPre system developers.

Based on these assumptions, we design and implement the language-centric and tagset-agnostic benchmarking system that enables comprehensive and credible evaluation, constitutes an up-to-date source of information on NLPre progress, and is fully configurable to facilitate building benchmarking systems for multiple languages.

2.2. Online benchmarking system

The benchmarking system comprises three main parts: a data repository, a submission and evaluation system, and a leaderboard. The data repository provides descriptions of NLPre tasks, datasets, and evaluation metrics, as well as links to the datasets.

The model submission and evaluation system allows the researchers to evaluate a new model by submitting its predictions for the test sets of raw sentences. It is mandatory to upload predictions for all provided test sets for a given tagset; however, it is possible to participate in an evaluation for only one tagset and only for a selected range of tasks.

The leaderboard is a tabular display of the performance of all submissions with their results for each dataset and tagset. The results for the evaluated model and its rank are displayed in the leaderboard provided the submitter confirms their publication.

The benchmarking system is implemented as a web-based application in Python using Django framework. This framework allows quite an easy implementation of MVC design pattern. Moreover, it offers access to the administrator panel, which can be very useful in the custom configuration of the benchmark. The submission scores are stored in a local SQLite database and the submissions are stored in `.zip` files in a designated directory. The results from the leaderboard are conveniently accessible via an API.

2.3. Configuration

We acknowledge the need to configure similar evaluation environments for other languages to promote linguistic diversity within the worldwide NLP community and to support local NLP communities working on a particular language. To ensure that, we publish a `.yaml` file that enables easy management of datasets, tagset, and metrics included in the benchmark. The content of all subpages can be

[5] https://nlpre-pl.clarin-pl.eu
[6] https://nlpre-zh.clarin-pl.eu
[7] https://nlpre-ga.clarin-pl.eu

modified using a WYSIWYG editor within the application. This setting ensures quite a low entry level for setting up the platform, with minimal changes required.

As a standard feature, we include pre-defined descriptions for the prevalent NLPre tasks. Those can be modified via either configuration files or the administrator panel. Additionally, we supply a default evaluation script, but users are free to provide their own customised code.

To show the capabilities of the benchmarking system, we set up a prototype for Polish (Figure 1). NLPre-PL is described in detail in Section 3. To support our claim that the system is language agnostic, we set up NLPre-GA for Irish and NLPre-ZH for Chinese. The choice of those languages is not arbitrary; our objective is to demonstrate the capability of the platform in evaluating diverse languages, including those based on non-Latin scripts. In setting up said benchmarking systems we use existing UDv2.9 treebanks: UD_Chinese-GSD (Shen et al., 2019) and UD_Irish-IDT (Lynn et al., 2015) and available up-to-date models, trained on these treebanks. The selection of models mirrors the criteria applied in this work regarding the evaluation of Polish, that is: COMBO, Stanza, SpaCy, UDPipe, and Trankit. If the specific model is not available for UDv2.9, we train it from scratch on the datasets linked above.

3. NLPre-PL benchmark

3.1. Datasets

	NKJP1M		**PDB-UD**
POS	Morfeusz	Morfeusz / UD	UD
DEP	n/a	n/a	UD
Format	TEI / DAG	CoNLL-X / -U	CoNLL-U
# tokens	1.2M		350K
# sentences	85.7K		22K
Avg. t/s	14.2		15.8
NLPre-PL			
Split	*byName*	*byType*	*original*
# train	984K	978K	282K
# dev	110K	112K	35K
# test	122K	125K	34K

Table 1: Summary of source datasets (NKJP1M and PDB-UD) and NLPre-PL Datasets (in tokens). Explanations: *POS* – the part-of-speech tagset; *DEP* – the dependency schema; *Avg. t/s* – the average number of tokens per sentence.

NKJP1M (Przepiórkowski et al., 2018) The NKJP1M subcorpus of the Polish National Corpus (Przepiórkowski et al., 2012) is manually annotated according to the NKJP tagset (Szałkiewicz and Przepiórkowski, 2012) and afterwards modified in line with the Morfeusz tagset (Woliński, 2019). This balanced subset of thematic- and genre-diverse texts and transcriptions is used to train Polish POS taggers. NKJP1M is maintained in two formats: TEI[8] and DAG.[9] These two formats are accepted by older NLPre tools but not modern ones. We thus convert NKJP1M to the CoNLL-X format (Buchholz and Marsi, 2006) preserving the original segmentation, POS tags and morphological features (i.e. the Morfeusz tagset), and to the CoNLL-U format[10] with UD tags, Morfeusz tags (*XPOS*) and UD morphological features.

Since there is no generally accepted split of NKJP1M into training, development and testing subsets, we uniformly divide NKJP1M in all formats (i.e. DAG, TEI, CoNLL-X and CoNLL-U) pursuant to the formulated splitting heuristics. Each document in the subcorpus contains multiple paragraphs of continuous textual data. To avoid possible information leakage, we treat each such paragraph as an indivisible unit. To ensure that the subsets include paragraphs of varying length, we investigate the distribution over the number of segments in each paragraph. Since it is akin to Gaussian distribution, we decide to not exclude any data, and we divide the paragraphs into K = 10 buckets of roughly similar size and then sample from them with respective ratios of 0.8:0.1:0.1 (corresponding to train, dev, and test subsets). This data selection technique assures similar distribution of segments number per paragraph in three subsets, hereafter *byName*. For creating our second split, hereafter *byType*, we consider the type of document a paragraph belongs to. We first group paragraphs into categories equal to the document types, and then we repeat the above-mentioned procedure per category (see the summary of NKJP1M and data splits in Table 1).

PDB-UD (Wróblewska, 2018) Polish Dependency Bank is the largest collection of Polish sentences manually annotated with dependency trees and afterwards converted into UD representations in line with the UD annotation schema (de Marneffe et al., 2021). PDB-UD slightly correlates with NKJP1M, i.e., a subset of the PDB-UD sentences comes from NKJP1M, and the language-specific tags (*XPOS*) in PDB-UD match the Morfeusz tagset. PDB-UD is typically used to train NLPre systems for Polish. In NLPre-PL, we use the original PDB-UD data without any modifications and its standard split (see the statistical summary of PDB-UD in Table 1).

3.2. Tasks

The complete set of NLPre tasks was originally curated for evaluating language systems in the CoNLL shared task 2018 (Zeman et al., 2018). These tasks

[8] http://nlp.ipipan.waw.pl/TEI4NKJP.

[9] https://github.com/kawu/concraft-pl#data-format

[10] https://universaldependencies.org/format.html

mainly focus on preliminary text processing, such as tokenisation or divulging morphosyntactic features. We follow the CoNLL task choice and include all these tasks in NLPre-PL.

Segmentation A segmentation task consists in splitting texts into sentences (*Sentences*), orthographic tokens (*Tokens*), and syntactic words (*Words*), the latter being the basic units of morphosyntactic analysis. Segmentation is not a trivial task. In some languages, an orthographic token may be recognised as a *multi-word token* (*multi-word* for short) combining multiple syntactic words, e.g. in Polish, the token *spalibyśmy* (Eng. *we would sleep*) consists of the past participle *spali* (Eng. *slept*), the conditional marker *by* (Eng. *would*) and the mobile inflection *śmy*. Since the consistent model of segmentation into words and sentences was used in NKJP1M and PDB-UD, we maintain this data segmentation in NLPre-PL. It is also worth mentioning that the CoNLL format (but not TEI and DAG) allows for annotating orthographic tokens; thus, they are included in the NLPre-PL benchmark.

Tagging A tagging task is the process of identifying parts of speech (i.e. POS tagging) and possibly morphological features (i.e. morphological analysis) of words. It follows a predefined POS tagset. As mentioned in Section 3.1, two tagsets are used in the NLPre-PL datasets: Morfeusz and UD.

Lemmatisation Lemmatisation involves predicting canonical forms of syntactic words. Canonical forms are conventionally established identifiers of lexemes (i.e. sets of inflectionally related syntactic words). Since Polish is a fusional language with a large number of inflected words, lemmatisation is an important task, albeit not trivial, e.g. the lemma of *kluczy* can be either the infinitive KLUCZYĆ (Eng. *to weave*) or the noun KLUCZ (Eng. *a key*).

Dependency parsing Dependency parsing is the process of automatically predicting the syntactic structure of an input sentence. A dependency structure is a labelled directed tree with nodes corresponding to syntactic words and edges between these words specifying dependency relations.

4. Evaluation

4.1. Evaluation methodology

To maintain the de facto standard to NLPre evaluation, we apply the evaluation measures defined for the CoNLL 2018 shared task and implemented in the official evaluation script.[11] In particular, we focus on F1 and *AlignedAccuracy*, which is similar to F1 but does not consider possible misalignments in tokens, words, or sentences.

In our evaluation process, we follow default training procedures suggested by the authors of the evaluated systems, i.e. we do not conduct any optimal hyperparameter search in favour of leaving the recommended model configuration as-is. We also do not further fine-tune selected models.

4.2. Evaluated systems

Based on the NLPre-PL benchmark, we evaluate both well-rooted rule-based disambiguation methods and modern systems based on neural network architectures to enable an informative and thorough comparison of different approaches. We use the most up-to-date versions of available tools at the time of conducting experiments: (1) pipelines of separate tools (Concraft-pl, UDPipe), (2) systems integrating separate models for NLPre tasks (spaCy, Stanza, Trankit), (3) end-to-end systems with a model for all NLPre tasks (COMBO), and large language model GPT-3.5.

Concraft-pl (Waszczuk, 2012; Waszczuk et al., 2018) [12] is a system for joint morphosyntactic disambiguation and segmentation.[13] It uses Morfeusz morphological analyser (Woliński, 2014; Kieraś and Woliński, 2017) to extract morphological and segmentation equivocates and then disambiguates them using the conditional random fields model. We train the Concraft-pl models with default parameters.

UDPipe (Straka and Straková, 2017) is a language-agnostic trainable NLPre pipeline.[14] Depending on the task, it uses recurrent neural networks (Graves and Schmidhuber, 2005) in segmentation and tokenization, the average perceptron in tagging and lemmatization, a rule-based approach in multi-word splitting, and a transition-based neural dependency parser. We train the UDPipe models with the default parameters. The dependency parser is trained with the Polish *fastText* embeddings (Grave et al., 2018).

SpaCy (Montani and Honnibal, 2022) is an NLP Python library shipped with pretrained pipelines and word vectors for multiple languages.[15] It also supports training the models for tagging and parsing, inter alia. We use spaCy to train pipelines for morphosyntactic analysis with: feed-forward network-

[11] https://universaldependencies.org/conll18/conll18_ud_eval.py

[12] Polish is a fusional language for which a two-stage tagging procedure is typically applied: first, a rule-based morphological analyser outputs all morphological interpretations of individual tokens, and then a tagging disambiguator selects the most likely one for each token. The tools implemented in accordance with this procedure are still imminent.

[13] https://github.com/kawu/concraft-pl (v2.0)

[14] https://ufal.mff.cuni.cz/udpipe (v1)

[15] https://github.com/explosion/spaCy (v3.4.1)

Model / Task	Average	Tokens	Sentences	Words	UPOS	XPOS	UFeats	AllTags	Lemmas	Tok/s CPU	Tok/s GPU
concraft	91.61	98.56	71.33	99.64	95.88	90.04	90.59	90.04	96.79	111	–
udpipe + fT	94.43	99.75	90.51	**99.73**	97.36	90.64	90.97	90.64	95.86	2365	2181
combo + fT	95.75	99.12	**93.33**	99.04	97.25	93.82	93.61	92.98	96.90	458	822
combo + H	**96.67**	99.12	**93.33**	99.04	**97.80**	**95.66**	**95.75**	**95.20**	**97.42**	241	722
stanza + fT	95.89	**99.76**	92.70	99.45	97.43	93.57	93.90	93.36	96.94	933	2379
spacy + pl	75.38	99.56	61.85	98.46	96.30	90.97	31.03	30.14	94.77	**3252**	**8407**
spacy + fT	75.15	99.56	61.85	98.46	95.89	89.93	31.03	30.08	94.43	3134	8063
spacy + P	76.12	99.56	61.85	98.46	97.02	94.60	31.03	30.46	95.98	1571	5367
trankit + R	92.59	98.37	89.39	97.84	95.36	89.74	90.05	88.73	91.19	287	541

Table 2: Results (F1 scores) and inference time (the number of tokens processed per second) of benchmarking the selected NLPre systems on the Morfeusz tagset averaged by the datasets (*byName* and *byType*). The systems are grouped into non-neural and neural by a double horizontal line. Embeddings used in the models are: *R* – xlm-RoBERTa-base, *fT* – fastText, *P* – Polbert, *pl* – pl-core-news-lg, *H* – HerBERT.

Model / Task	Average	Tokens	Sentences	Words	UPOS	XPOS	UFeats	AllTags	Lemmas	Tok/s CPU	Tok/s GPU
udpipe + fT	92.30	**99.79**	92.44	**99.78**	97.33	89.97	90.37	89.35	95.23	1977	1848
combo + fT	94.04	99.18	**94.29**	98.77	96.64	93.30	93.48	91.97	96.53	471	844
combo + H	**95.51**	99.21	**94.29**	98.77	**97.57**	**95.33**	**95.61**	**94.54**	**97.13**	254	733
stanza + fT	94.25	99.77	93.92	99.43	97.33	92.88	92.90	91.63	96.60	910	2262
spacy + pl	88.39	99.58	65.05	98.47	96.36	90.95	91.22	89.65	93.62	**2495**	**5403**
spacy + fT	87.68	99.58	65.05	98.47	95.79	89.77	90.05	88.37	93.37	2484	4533
spacy + P	90.70	99.58	65.05	98.47	97.26	94.68	94.84	94.09	94.89	1376	4207
trankit + R	92.91	98.88	92.44	98.52	96.50	91.74	91.91	90.21	90.47	319	593

Table 3: Results (F1 scores) and inference time (tokens per second) of benchmarking the selected NLPre systems on the UD tagset averaged by the datasets (*byName*, *byType*, and *PDB-UD*). The systems are grouped into non-neural and neural by a double horizontal line (Concraft is not included because it does not allow data in the UD tagset) Embeddings used in the models are: *R* – xlm-RoBERTa-base, *fT* – fastText, *P* – Polbert, *pl* – pl-core-news-lg, *H* – HerBERT.

Task / model	*udpipe + fT*	*combo + fT*	*combo + H*	*stanza + fT*	*spacy + fT*	*spacy + pl*	*spacy + P*	*trankit + R*	*GPT-3.5*
Avg. F1 on PDB-UD	88.16	90.46	93.37	92.10	83.03	84.21	87.98	**94.03**	50.95
Tokens	99.86	99.35	99.40	99.86	99.65	99.65	99.65	**99.90**	98.08
Sentences	95.90	96.22	96.22	96.83	71.46	71.46	71.46	**98.51**	89.81
Words	99.84	98.22	98.22	99.42	98.51	98.51	98.51	**99.89**	96.96
UPOS	97.28	95.34	97.31	97.64	95.62	96.49	97.54	**99.07**	64.07
XPOS	88.57	92.03	94.92	93.17	88.57	90.14	94.35	**96.18**	41.32
UFeats	89.07	92.21	95.23	93.22	88.79	90.42	94.52	**96.34**	41.88
AllTags	88.02	90.41	94.29	92.15	87.00	88.71	93.85	**95.57**	35.65
Lemmas	94.29	95.37	**96.38**	95.77	91.72	91.69	93.77	88.98	64.77
UAS	86.68	88.49	91.31	91.09	80.91	82.15	88.08	**95.79**	35.57
LAS	83.01	86.19	89.98	88.83	72.24	73.60	80.33	**94.24**	26.58
CLAS	79.53	84.14	89.03	86.90	73.72	75.71	80.50	**93.00**	29.06
MLAS	69.53	76.64	84.77	79.90	63.57	66.90	75.75	**87.79**	11.81
BLEX	74.49	81.34	**86.77**	82.53	67.64	69.29	75.48	77.18	26.86
Avg. F1 on NKJP1M	94.37	95.84	**96.59**	95.33	90.01	90.49	92.06	92.36	NA

Table 4: Results of benchmarking the selected NLPre systems on the smaller PDB-UD dataset. The last row with the mean F1 scores of the models trained on larger NKJP1M data is for reference. Embeddings used in the models are: *R* – xlm-RoBERTa-base, *fT* – fastText, *P* – Polbert, *pl* – pl-core-news-lg, *H* – HerBERT. The results of GPT-3.5 are greyed out due to their exclusion from display on the leaderboard.

based text encoders with static embeddings (*fastText* and *pl-core-news-lg*) or transformer-based encoders with the Polbert embeddings (Kłeczek, 2021), taggers (linear layers with softmax activation on top of the encoders), and transition-based parsers.

Stanza (Qi et al., 2020) is a language-agnostic, fully neural toolkit offering a modular pipeline for tokenization, multi-word token expansion, lemmatization, tagging, and dependency parsing.[16] It mainly uses recurrent neural networks (Graves and Schmidhuber, 2005) as a base architecture and external word embeddings (*fastText*). Each module reuses the basic architecture.

Trankit (Nguyen et al., 2021b) uses a multilingual pre-trained transformer-based language model, XLM-Roberta (Conneau et al., 2019) as the text encoder which is then shared across pipelines for different languages.[17] The resulting model is jointly trained on 90 UD treebanks with a separate adapter (Pfeiffer et al., 2020a,b) for each treebank. Trankit uses a wordpiece-based splitter to exploit contextual information.

COMBO (Rybak and Wróblewska, 2018; Klimaszewski and Wróblewska, 2021) is a fully neural language-independent NLPre system[18] integrated with the LAMBO tokeniser (Przybyła, 2022). It is an end-to-end system with jointly trained modules for tagging, parsing, and lemmatisation. We train the COMBO models with the pre-trained word embeddings – *fastText* and *HerBERT* (Mroczkowski et al., 2021).

GPT-3.5 (Brown et al., 2020) is a large language model, notable for its outstanding performance in NLU tasks. It is a fined-tuned version of the GPT-3 model. GPT-3.5's architecture is based on a transformer neural network with 12 stacks of decoders blocks with multi-head attention blocks.

For segmentation tasks, we train modules integrated with the tested NLPre systems. The only aberration is in spaCy, where poor segmentation results of the dependency module[19] forced us to use an out-of-the-box sentenciser available in spaCy.

For each model, we initialise training with possibly the most prominent and congruent embedding model available. Virtually all models are capable of fully capitalising from that addition, apart from Concraft and UDPipe. The first does not use embeddings at all, and the latter uses them only for dependency parsing training. If embeddings based on BERT architecture are feasible to use, we select their *base* versions. This ensures fairness of comparison between NLPre systems, as not all of them support BERT-*large* embeddings.

[16] https://github.com/stanfordnlp/stanza (v1.4.0)

[17] https://github.com/nlp-uoregon/trankit (v1.1.1)

[18] https://gitlab.clarin-pl.eu/syntactic-tools/combo (v1.0.5)

[19] Dependency parsing module is responsible for sentence segmentation in the spaCy implementation.

4.3. Results

Impact of system architecture We assess the quality of the selected NLPre systems contingent on the NLPre-PL benchmark. In Polish (and most other languages), non-neural NLPre tools are currently not widely developed. We evaluate two of them: Concraft and UDPipe. Although they do not use neural network algorithms to train models, their quality does not significantly differ from the best tested neural systems, especially in terms of segmentation, which UDPipe performs best (*Words*) or second-best (*Sentences*) (see Tables 2 and 3). We cannot unequivocally say that the system architecture has a decisive influence on the results, as spaCy models, even transformer-based, output the lowest quality.

Impact of tagset selection We compare systems trained and tested on data adjusted to two tagsets – the Morfeusz tagset (see Table 2) and the UD tagset (see Table 3). The average scores indicate that only COMBO performs better on Morfeusz-annotated data than on UD data. The performance of Trankit, UDPipe, and Stanza slightly decreases on Morfeusz data. Notably, all spaCy models trained on this dataset record a significant quality drop mainly due to poorly performed morphological analysis, i.e. *UFeats* values (and thus also the low *AllTags* values, i.e., matching between *UPOS*, *XPOS*, and *UFeats*). Regarding segmentation, *UPOS* and *XPOS* tagging, and lemmatisation, the tagset selection does not negatively affect the results, and the systems perform comparably.

Impact of the size of training data Intuitively, the size of the training data affects the prediction quality. Considering the data size factor, we compare the average F1 scores of the NLPre systems trained on NKJP1M (see the last row in Table 4) and on PDB-UD (see Table 4), which is two orders of magnitude smaller. The results confirm our intuitive assumptions – there is a difference of 6.21 between the mean F1 scores obtained by the systems trained on the smaller PDB-UD (avg. F1 of 88.16) and those trained on the larger NKJP1M (avg. F1 of 94.37).

When comparing the performance of individual systems on the smaller PDB-UD dataset, Trankit turns out to be the undisputed winner in all tasks except lemmatisation. However, considering the average performance of all tasks, COMBO and Stanza perform the best.

In alignment with contemporary developments on zero-shot learning, we test the predictive capabilities of GPT-3.5 acquired via the prompting technique (Brown et al., 2020). Despite comprehensive instructions along with the UD tree examples in the prompt, the results are highly unsatisfactory. An error analysis has revealed that 1) the GPT model modifies the input texts (e.g. adds elided words, alters the word's declension and conjugation, leading also to non-existent words); 2) while parsing questions, it answers them or returns information that they cannot be answered; 3) it replaces Polish words with their foreign equivalents; 4) it outputs graphs with cycles, thus not adhering to UD trees. Even for GPTs, achieving UD-compliant morphosyntactic analysis is challenging when they lack access to training examples. GPT-3.5's results are not included in the leaderboard.

Impact of split heuristics As outlined in Section 3.1, NKJP1M has no official split into train, dev, and test subsets. Since intuitively, the type of document can affect text processing, we propose two alternative splits, i.e. *byName* and *byType*. We compare the F1 scores for these two splits to verify this hypothesis. For the *byName* split, the average F1 for tasks and systems is 90.69, and for the *byType* split, it is 90.56. The difference is negligible, indicating that the document type, and hence the text domain, does not affect the quality of the NLPre tasks. Based on this outcome, we arbitrarily choose the more balanced *byType* split as binding in the final NLPre-PL benchmarking system. The detailed results of all experiments are in Appendix 6.2.

Inference time In the context of benchmarking, quality is a fundamental factor. In our case, the best average F1 scores are achieved by COMBO and Stanza, far ahead of spaCy and Concraft. The second crucial issue is the processing time of the evaluated NLPre systems, especially their inference time.[20] We calculate the times in which the systems tokenise, tag and lemmatise the input text.[21] The exception is COMBO with the mandatory parsing module that cannot be disabled. Therefore, its calculations include the parsing time as well. The inference time, corresponding to the number of tokens processed per second, is provided in the last two columns of Tables 2 and 3. On CPU, the fastest systems are spaCy and UDPipe, and the slowest is Concraft. Other systems process one order of magnitude fewer tokens per second than the top ones. On GPU, spaCy is the undisputed winner, followed by Stanza, UDPipe, COMBO and Trankit.

Correlation analysis We conduct a statistical analysis to capture meaningful relations between the performance and the model types, the used embeddings, or the datasets. To check whether the performance of a given model on a given tagset allows us to expect similar relationships between the scores on another tagset, we calculate a correlation matrix of vectors composed of the F1 scores for various tasks, i.e. $\vec{v} = [$*Tokens*, *Sentences*, *Words*, *UPOS*, *XPOS*, *Lemmas*$]$, averaged over embeddings and datasets (see Figure 2). The vectors are calculated for a pair ($tagset_i$, $model_j$). To maintain comparability, we exclude PDB-UD from the study as it does not appear in the Morfeusz tagset.

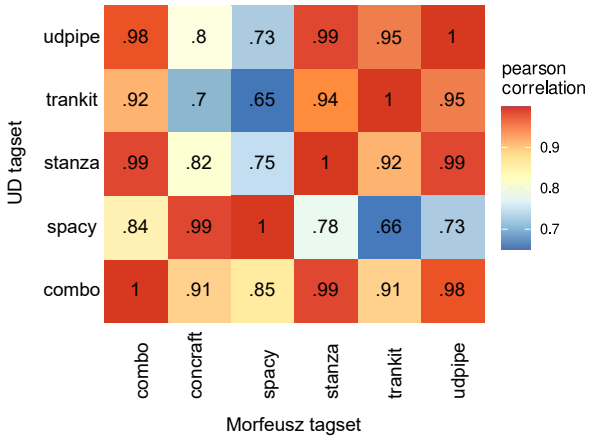


Figure 2: Pearson correlation coefficients between vectors of F1 scores on *Tokens*, *Sentences*, *Words*, *UPOS*, *XPOS*, *Lemmas* tasks averaged over datasets (excluding PDB-UD) and embeddings.

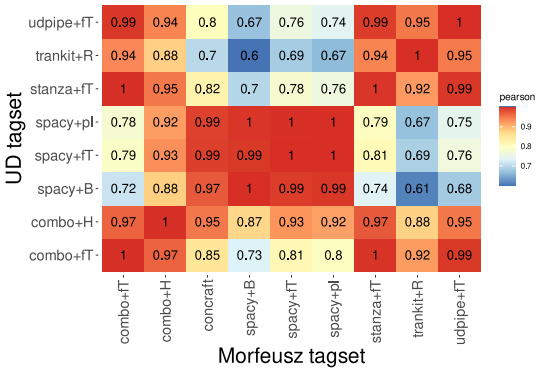


Figure 3: Pearson correlation coefficients between vectors of F1 scores on *Tokens*, *Sentences*, *Words*, *UPOS*, *XPOS*, *Lemmas* tasks averaged over datasets (excluding PDB-UD).

Pearson's correlation r suggests that the results are linearly proportional for the same mod-

[20] We share a conviction favoured in the NLP community that the training time is slightly less requisite than the inference time since models are trained only once but then constantly reused for predictions. We thus provide inference times.

[21] We run tests uniformly on CPU – Intel Xeon Platinum 8268 processor (1 node with 12 cores), and GPU – 2x Tesla V100-SXM2. The machines used to train the models are listed in Appendix 6.1.

els and different tagsets, which we conclude from the values close to 1 at the intersection of (*model*$_i$, *tagset*$_{\text{UD}}$) and (*model*$_i$, *tagset*$_{\text{NKJP}}$). Even though correlation coefficients are generally high (i.e. $r \in [0.90, 0.99]$) for most pairs (*model*$_i$, *tagset*$_{\text{UD}}$) and (*model*$_j$, *tagset*$_{\text{NKJP}}$), there are noticeable lower values for spaCy, i.e. $r \in [0.66, 0.78]$. We hypothesise that this is due to the non-linear rate of changes between the scores, as all Spearman correlation coefficients exceed 0.89 (i.e. $\rho > 0.89$).

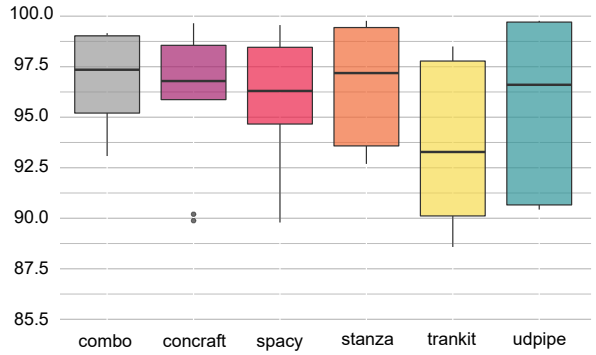


Figure 4: Dispersion of model performance measured by F1 on the Morfeusz tagset and *Sentences*, *Words*, *UPOS*, *XPOS*, and *Lemmas* tasks.

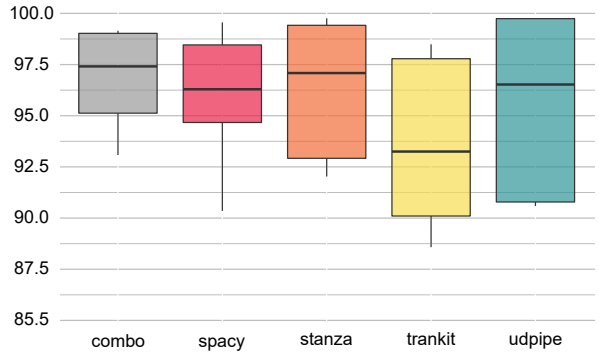


Figure 5: Dispersion of model performance measured by F1 on the UD tagset and *Sentences*, *Words*, *UPOS*, *XPOS*, and *Lemmas* tasks.

The results of a more granular analysis of Pearson's r between vectors of F1 scores for triples (tagset$_i$, model$_j$, embeddings$_k$), averaged over datasets, show a strong correlation for the same models, regardless of the tagset and the embedding (see Figure 3). Hence, if a change in the tagset or embedding causes an increase in one task, a proportional increase in remaining tasks is expected.

Boxplot charts (see Figures 4 and 5) determine the stability of the model results for a given tagset regardless of dataset and embedding. One box shows the scattering of F1 scores for *Tokens*, *Sentences*, *Words*, *UPOS*, *XPOS*, and *Lemmas* tasks. The shortest COMBO's box indicates a relatively similar performance of the model across tasks for each triplet (COMBO, *embedding*$_j$, *dataset*$_k$).

5. Conclusions

In this work, we propose a revised approach to NLPre evaluation via benchmarking. This is motivated by the widespread use of the benchmarking technique in other NLP fields on par with the shortcomings of existing NLPre evaluation solutions.

We implement said NLPre benchmarking approach as the online system that evaluates the submitted outcome of an NLPre system and updates the associated leaderboard with the results after the submitter's approval. The benchmarking system is designed to rank NLPre tools available for a given language in a trustworthy environment.

The endeavour of defining and enhancing the system's capabilities is conducted concurrently with the effort to create the NLPre benchmark for Polish that encompasses numerous factors, such as tasks not required in English or diverse tagsets. The NLPre-PL benchmark consists of the predefined NLPre tasks, coupled with two reformulated datasets. The NLPre-PL benchmark, therefore, sets the standard for evaluating the performance of the NLPre tools for Polish, which represents a derivative yet important outcome of our research.

In addition to integration into the benchmarking system, NLPre-PL is used to conduct empirical experiments. We perform a robust and extensive comparison of different NLPre methods, including the classical non-neural tools and the modern neural network-based techniques. The results of these experiments on datasets in two tagsets are discussed in detail. The experiments confirm our assumptions that modern architectures obtain better results. Because NLP is a discipline undergoing rapid progress, new NLPre solutions, e.g. multilingual or zero-shot, can be expected in the coming years. These new solutions can be easily tested and compared with the tools evaluated so far in our benchmarking system.

Finally, we release the open-source code of the benchmarking system in hopes that this endeavour could be replicated for other languages. To expedite this process, we ensure that the system is fully configurable and language- and tagset-agnostic. The NLPre system, configured for a specified language, can be self-hosted on a chosen server, and the results from the leaderboard are conveniently accessible via an API. We see a potential future application of our system to the UD repository, where for 141 languages, there are currently 245 treebanks with supposedly discrepant versions of the UD tagset.

6. Appendices

6.1. Infrastructure used

We train the models using several types of computational nodes at our disposal, including NVIDIA V100 32GB, NVIDIA GeForce RTX 2080 8GB, NVIDIA GeForce 3070 8GB and Intel Xeon E5-2697 processor. Since we do not perform hyperparameter tuning, this should not impact our results.

6.2. Further results of experiments

Herein, we present a comprehensive depiction of our experimental findings as they are displayed on the NLPre-PL leaderboard.

In Table 5, we present the full results of the evaluation of the selected models on the Morfeusz-based datasets *byName* and *byType*. These results are provided for all available tasks that can be performed on the above-mentioned datasets. As NKJP1M datasets contain no syntantic trees, it is thus impossible to test the dependency parsing task that rely on these trees and measure *UAS*, *LAS*, *CLAS*, *MLAS* and *BLEX*.

In Table 6, we present the results of the evaluation of the selected models on the UD-based datasets *byName*, *byType*, and *PDB*. This table contains the results of segmentation, tagging, and lemmatization tasks. Table 7 is a continuation of Table 6 and it contains the results for the same tagset and dataset on the dependency parsing task.

Model / Task	Dataset	Scores	Average	Tokens	Sentences	Words	UPOS	XPOS	UFeats	AllTags	Lemmas
combo	bN	AA	97.31	-	-	-	98.74	96.63	96.70	96.15	98.36
+ *H*	bN	F1	96.68	99.07	93.57	99.01	97.76	95.67	95.74	95.20	97.39
	bT	AA	97.28	-	-	-	98.76	96.54	96.65	96.08	98.35
	bT	F1	96.65	99.16	93.08	99.07	97.84	95.65	95.76	95.19	97.44
stanza	bN	AA	95.58	-	-	-	97.97	94.09	94.44	93.89	97.51
+ *fT*	bN	F1	95.88	99.75	92.69	99.43	97.41	93.55	93.91	93.36	96.96
	bT	AA	95.55	-	-	-	97.97	94.10	94.38	93.85	97.43
	bT	F1	95.89	99.77	92.70	99.46	97.45	93.59	93.88	93.35	96.9
combo	bN	AA	95.87	-	-	-	98.15	94.81	94.60	93.97	97.80
+ *fT*	bN	F1	95.78	99.07	93.57	99.01	97.18	93.87	93.67	93.04	96.84
	bT	AA	95.78	-	-	-	98.22	94.63	94.41	93.77	97.85
	bT	F1	95.72	99.16	93.08	99.07	97.31	93.76	93.54	92.91	96.95
udpipe	bN	AA	93.34	-	-	-	97.57	90.90	91.22	90.90	96.12
+ *fT*	bN	F1	94.44	99.77	90.43	99.75	97.33	90.68	90.99	90.68	95.88
	bT	AA	93.35	-	-	-	97.67	90.87	91.21	90.87	96.13
	bT	F1	94.42	99.73	90.58	99.70	97.38	90.60	90.94	90.60	95.84
trankit	bN	AA	93.06	-	-	-	97.49	91.77	92.05	90.73	93.25
+ *R*	bN	F1	92.81	98.50	90.19	97.96	95.50	89.89	90.17	88.88	91.35
	bT	AA	92.99	-	-	-	97.43	91.69	92.03	90.65	93.15
	bT	F1	92.36	98.24	88.58	97.72	95.21	89.59	89.93	88.58	91.02
concraft	bN	AA	93.09	-	-	-	96.24	90.51	91.05	90.51	97.13
	bN	F1	91.70	98.56	71.55	99.65	95.90	90.20	90.73	90.20	96.79
	bT	AA	92.92	-	-	-	96.22	90.22	90.79	90.22	97.15
	bT	F1	91.52	98.55	71.10	99.62	95.86	89.88	90.45	89.88	96.79
spacy	bN	AA	70.94	-	-	-	98.54	96.12	31.54	30.96	97.52
+ *P*	bN	F1	76.23	99.56	62.64	98.45	97.01	94.64	31.05	30.48	96.01
	bT	AA	70.88	-	-	-	98.55	96.03	31.49	30.91	97.44
	bT	F1	76.00	99.56	61.06	98.46	97.03	94.55	31.00	30.43	95.94
spacy	bN	AA	69.77	-	-	-	97.86	92.47	31.54	30.68	96.30
+ *pl*	bN	F1	75.51	99.56	62.64	98.45	96.34	91.04	31.05	30.21	94.81
	bT	AA	69.66	-	-	-	97.77	92.31	31.49	30.54	96.21
	bT	F1	75.25	99.56	61.06	98.46	96.26	90.89	31.00	30.07	94.73
spacy	bN	AA	69.39	-	-	-	97.42	91.48	31.54	30.61	95.89
+ *fT*	bN	F1	75.28	99.56	62.64	98.45	95.92	90.06	31.05	30.13	94.40
	bT	AA	69.29	-	-	-	97.35	91.20	31.49	30.49	95.94
	bT	F1	75.02	99.56	61.06	98.46	95.85	89.79	31.00	30.02	94.46

Table 5: Benchmark results for the Morfeusz tagset performed on two datasets: NKJP-*byType* (bT) and NKJP-*byName* (bN); AA – Aligned Accuracy; F1 – F1 score. Embeddings used in the models are: *R* – xlm-RoBERTa-base, *fT* – fastText, *P* – Polbert-base, *pl* – pl-core-news-lg, *H* – HerBERT.

Model / Task	Dataset	Scores	Average	Tokens	Sentences	Words	UPOS	XPOS	UFeats	AllTags	Lemmas
combo	bN	AA	97.18	-	-	-	98.63	96.45	96.77	95.60	98.42
+ *H*	bN	F1	96.59	99.07	93.57	99.01	97.65	95.50	95.81	94.66	97.45
	bT	AA	97.17	-	-	-	98.66	96.46	96.70	95.57	98.48
	bT	F1	96.58	99.15	93.08	99.07	97.75	95.56	95.80	94.68	97.57
	PDB	AA	93.62	-	-	-	99.07	96.65	96.96	96.00	98.13
	PDB	F1	93.37	99.40	96.22	98.22	97.31	94.92	95.23	94.29	96.38
stanza	bN	AA	94.66	-	-	-	97.67	93.11	93.13	91.73	97.68
+ *fT*	bN	F1	95.20	99.70	92.03	99.40	97.08	92.55	92.56	91.17	97.09
	bT	AA	94.82	-	-	-	97.78	93.42	93.41	92.05	97.45
	bT	F1	95.46	99.76	92.89	99.47	97.26	92.93	92.91	91.56	96.93
	PDB	AA	90.60	-	-	-	98.21	93.71	93.76	92.69	96.32
	PDB	F1	92.10	99.86	96.83	99.42	97.64	93.17	93.22	92.15	95.77
combo	bN	AA	95.93	-	-	-	98.20	94.79	95.01	93.59	98.04
+ *fT*	bN	F1	95.82	99.07	93.57	99.01	97.22	93.86	94.07	92.67	97.07
	bT	AA	96.00	-	-	-	98.28	94.88	95.05	93.69	98.07
	bT	F1	95.85	99.13	93.08	99.07	97.37	94.00	94.17	92.83	97.16
	PDB	AA	89.77	-	-	-	97.07	93.70	93.88	92.05	97.11
	PDB	F1	90.46	99.35	96.22	98.22	95.34	92.03	92.21	90.41	95.37
trankit	bN	AA	92.68	-	-	-	97.31	91.56	91.72	89.51	93.30
+ *R*	bN	F1	92.57	98.49	90.24	97.95	95.32	89.68	89.84	87.68	91.39
	bT	AA	92.58	-	-	-	97.32	91.43	91.62	89.40	93.15
	bT	F1	92.12	98.24	88.58	97.73	95.11	89.36	89.55	87.37	91.04
	PDB	AA	92.51	-	-	-	99.18	96.28	96.44	95.68	89.08
	PDB	F1	94.03	99.90	98.51	99.89	99.07	96.18	96.34	95.57	88.98
udpipe	bN	AA	93.20	-	-	-	97.61	90.91	91.27	90.29	95.94
+ *fT*	bN	F1	94.39	99.75	90.82	99.74	97.36	90.68	91.03	90.06	95.70
	bT	AA	93.17	-	-	-	97.59	90.88	91.24	90.20	95.94
	bT	F1	94.35	99.77	90.59	99.76	97.35	90.65	91.02	89.98	95.70
	PDB	AA	85.14	-	-	-	97.43	88.71	89.21	88.16	94.44
	PDB	F1	88.16	99.86	95.90	99.84	97.28	88.57	89.07	88.02	94.29
spacy	bN	AA	96.82	-	-	-	98.63	96.37	96.50	95.72	96.87
+ *P*	bN	F1	92.15	99.56	62.64	98.45	97.10	94.88	95.00	94.23	95.37
	bT	AA	96.83	-	-	-	98.67	96.30	96.48	95.68	97.02
	bT	F1	91.97	99.54	61.06	98.46	97.15	94.82	94.99	94.20	95.52
	PDB	AA	87.45	-	-	-	99.02	95.77	95.95	95.27	95.19
	PDB	F1	87.98	99.65	71.46	98.51	97.54	94.35	94.52	93.85	93.77
spacy	bN	AA	94.27	-	-	-	97.77	92.82	93.12	91.58	96.05
+ *pl*	bN	F1	90.59	99.56	62.64	98.45	96.25	91.38	91.68	90.16	94.56
	bT	AA	94.24	-	-	-	97.84	92.75	93.01	91.50	96.10
	bT	F1	90.38	99.54	61.06	98.46	96.34	91.32	91.57	90.09	94.62
	PDB	AA	82.58	-	-	-	97.95	91.50	91.79	90.05	93.07
	PDB	F1	84.21	99.65	71.46	98.51	96.49	90.14	90.42	88.71	91.69
spacy	bN	AA	93.47	-	-	-	97.34	91.83	92.17	90.48	95.56
+ *fT*	bN	F1	90.10	99.56	62.64	98.45	95.83	90.40	90.74	89.07	94.07
	bT	AA	93.49	-	-	-	97.44	91.77	92.03	90.42	95.79
	bT	F1	89.91	99.54	61.06	98.46	95.93	90.35	90.62	89.03	94.32
	PDB	AA	81.07	-	-	-	97.06	89.91	90.13	88.31	93.10
	PDB	F1	83.03	99.65	71.46	98.51	95.62	88.57	88.79	87.00	91.72

Table 6: Benchmark results for the UD tagset performed on three datasets: NKJP-*byType* (bT), NKJP-*byName* (bN), and PDB-UD (PDB) for segmentation, tagging and lemmatization tasks; AA – Aligned Accuracy; F1 – F1 score. Embeddings used in the models are: *R* – xlm-RoBERTa-base, *fT* – fastText, *P* – Polbert-base, *pl* – pl-core-news-lg, *H* – HerBERT-base.

Model / Task	Dataset	Scores	Average	UAS	LAS	CLAS	MLAS	BLEX
combo	bN	AA	-	-	-	-	-	
+ H	bN	F1	-	-	-	-	-	
	bT	AA	-	-	-	-	-	
	bT	F1	-	-	-	-	-	
	PDB	AA	93.62	92.97	91.61	90.47	86.15	88.18
	PDB	F1	93.37	91.31	89.98	89.03	84.77	86.77
stanza	bN	AA	-	-	-	-	-	
+ fT	bN	F1	-	-	-	-	-	
	bT	AA	-	-	-	-	-	
	bT	F1	-	-	-	-	-	
	PDB	AA	90.60	91.62	89.34	87.25	80.22	82.87
	PDB	F1	92.10	91.09	88.83	86.90	79.90	82.53
combo	bN	AA	-	-	-	-	-	
+ fT	bN	F1	-	-	-	-	-	
	bT	AA	-	-	-	-	-	
	bT	F1	-	-	-	-	-	
	PDB	AA	89.77	90.10	87.76	85.49	77.88	82.65
	PDB	F1	90.46	88.49	86.19	84.14	76.64	81.34
trankit	bN	AA	-	-	-	-	-	
+ R	bN	F1	-	-	-	-	-	
	bT	AA	-	-	-	-	-	
	bT	F1	-	-	-	-	-	
	PDB	AA	92.51	95.89	94.34	93.10	87.88	77.26
	PDB	F1	94.03	95.79	94.24	93.00	87.79	77.18
udpipe	bN	AA	-	-	-	-	-	
+ fT	bN	F1	-	-	-	-	-	
	bT	AA	-	-	-	-	-	
	bT	F1	-	-	-	-	-	
	PDB	AA	85.14	86.82	83.14	79.52	69.52	74.48
	PDB	F1	88.16	86.68	83.01	79.53	69.53	74.49
spacy	bN	AA	-	-	-	-	-	
+ P	bN	F1	-	-	-	-	-	
	bT	AA	-	-	-	-	-	
	bT	F1	-	-	-	-	-	
	PDB	AA	87.45	89.41	81.54	77.23	72.67	72.41
	PDB	F1	87.98	88.08	80.33	80.50	75.75	75.48
spacy	bN	AA	-	-	-	-	-	
+ pl	bN	F1	-	-	-	-	-	
	bT	AA	-	-	-	-	-	
	bT	F1	-	-	-	-	-	
	PDB	AA	82.58	83.39	74.71	72.66	64.21	66.50
	PDB	F1	84.21	82.15	73.60	75.71	66.90	69.29
spacy	bN	AA	-	-	-	-	-	
+ fT	bN	F1	-	-	-	-	-	
	bT	AA	-	-	-	-	-	
	bT	F1	-	-	-	-	-	
	PDB	AA	81.07	82.13	73.33	70.76	61.02	64.93
	PDB	F1	83.03	80.91	72.24	73.72	63.57	67.64

Table 7: Benchmark results for the UD tagset performed on three datasets: NKJP-*byType* (bT), NKJP-*byName* (bN), and PDB-UD (PDB) for the dependency parsing task; AA – Aligned Accuracy; F1 – F1 score. Embeddings used in the models are: *R* – xlm-RoBERTa-base, *fT* – fastText, *P* – Polbert-base, *pl* – pl-core-news-lg, *H* – HerBERT.

7. Acknowledgements

This work was supported by the European Regional Development Fund as a part of 2014–2020 Smart Growth Operational Programme, CLARIN — Common Language Resources and Technology Infrastructure (project no. POIR.04.02.00-00C002/19) and DARIAH-PL — Digital Research Infrastructure for the Arts and Humanities (project no. POIR.04.02.00-00-D006/20-0), and as part of the investment CLARIN ERIC: Common Language Resources and Technology Infrastructure (period: 2024-2026) funded by the Polish Ministry of Science and Higher Education (agreement no. 2024/WK/01). We gratefully acknowledge Poland's high-performance computing infrastructure PLGrid (HPC Centers: ACK Cyfronet AGH) for providing computer facilities and support within computational grant no. PLG/2022/015872.

8. Bibliographical References

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language Models are Few-Shot Learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Sabine Buchholz and Erwin Marsi. 2006. CoNLL-X shared task on multilingual dependency parsing. In *Proceedings of the Tenth Conference on Computational Natural Language Learning (CoNLL-X)*, pages 149–164, New York City. Association for Computational Linguistics.

Kehai Chen, Tiejun Zhao, Muyun Yang, and Lemao Liu. 2017. Translation prediction with source dependency-based context representation. *Proceedings of the AAAI Conference on Artificial Intelligence*, 31(1).

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online. Association for Computational Linguistics.

Dick Crouch, Mary Dalrymple, Ronald M. Kaplan, Tracy Holloway King, John Maxwell, and Paula Newman. 2011. XLE Documentation. Palo Alto Research Center.

Marie-Catherine de Marneffe, Christopher D. Manning, Joakim Nivre, and Daniel Zeman. 2021. Universal Dependencies. *Computational Linguistics*, 47(2):255–308.

Sebastian Gehrmann, Tosin Adewumi, Karmanya Aggarwal, Pawan Sasanka Ammanamanchi, Anuoluwapo Aremu, Antoine Bosselut, Khyathi Raghavi Chandu, Miruna-Adriana Clinciu, Dipanjan Das, Kaustubh Dhole, Wanyu Du, Esin Durmus, Ondřej Dušek, Chris Chinenye Emezue, Varun Gangal, Cristina Garbacea, Tatsunori Hashimoto, Yufang Hou, Yacine Jernite, Harsh Jhamtani, Yangfeng Ji, Shailza Jolly, Mihir Kale, Dhruv Kumar, Faisal Ladhak, Aman Madaan, Mounica Maddela, Khyati Mahajan, Saad Mahamood, Bodhisattwa Prasad Majumder, Pedro Henrique Martins, Angelina McMillan-Major, Simon Mille, Emiel van Miltenburg, Moin Nadeem, Shashi Narayan, Vitaly Nikolaev, Andre Niyongabo Rubungo, Salomey Osei, Ankur Parikh, Laura Perez-Beltrachini, Niranjan Ramesh Rao, Vikas Raunak, Juan Diego Rodriguez, Sashank Santhanam, João Sedoc, Thibault Sellam, Samira Shaikh, Anastasia Shimorina, Marco Antonio Sobrevilla Cabezudo, Hendrik Strobelt, Nishant Subramani, Wei Xu, Diyi Yang, Akhila Yerukola, and Jiawei Zhou. 2021. The GEM benchmark: Natural language generation, its evaluation and metrics. In *Proceedings of the 1st Workshop on Natural Language Generation, Evaluation, and Metrics (GEM 2021)*, pages 96–120, Online. Association for Computational Linguistics.

Alex Graves and Jürgen Schmidhuber. 2005. Framewise phoneme classification with bidirectional lstm and other neural network architectures. *Neural Networks*, 18(5):602–610. IJCNN 2005.

Zhijiang Guo, Yan Zhang, and Wei Lu. 2019. Attention guided graph convolutional networks for relation extraction. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 241–251, Florence, Italy. Association for Computational Linguistics.

Junjie Hu, Sebastian Ruder, Aditya Siddhant, Graham Neubig, Orhan Firat, and Melvin Johnson. 2020. XTREME: A massively multilingual multi-task benchmark for evaluating cross-lingual generalisation. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 4411–4421. PMLR.

Jungo Kasai, Dan Friedman, Robert Frank, Dragomir Radev, and Owen Rambow. 2019. Syntax-aware neural semantic role labeling with supertags. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 701–709, Minneapolis, Minnesota. Association for Computational Linguistics.

Daniel Khashabi, Tushar Khot, Ashish Sabharwal, and Dan Roth. 2018. Question answering as global reasoning over semantic abstractions. *Proceedings of the AAAI Conference on Artificial Intelligence*, 32(1).

Douwe Kiela, Max Bartolo, Yixin Nie, Divyansh Kaushik, Atticus Geiger, Zhengxuan Wu, Bertie Vidgen, Grusha Prasad, Amanpreet Singh, Pratik Ringshia, Zhiyi Ma, Tristan Thrush, Sebastian Riedel, Zeerak Waseem, Pontus Stenetorp, Robin Jia, Mohit Bansal, Christopher Potts, and Adina Williams. 2021. Dynabench: Rethinking benchmarking in NLP. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4110–4124, Online. Association for Computational Linguistics.

Witold Kieraś and Marcin Woliński. 2017. Morfeusz 2 – analizator i generator fleksyjny dla języka polskiego. *Język Polski*, XCVII(1):75–83.

Mateusz Klimaszewski and Alina Wróblewska. 2021. COMBO: State-of-the-art morphosyntactic analysis. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 50–62, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

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

Ines Montani and Matthew Honnibal. 2022. spaCy: Industrial-Strength Natural Language Processing in Python. Version 3.4.1.

Robert Mroczkowski, Piotr Rybak, Alina Wróblewska, and Ireneusz Gawlik. 2021. HerBERT: Efficiently pretrained transformer-based language model for Polish. In *Proceedings of the 8th Workshop on Balto-Slavic Natural Language Processing*, pages 1–10, Kiyv, Ukraine. Association for Computational Linguistics.

Minh Van Nguyen, Viet Dac Lai, Amir Pouran Ben Veyseh, and Thien Huu Nguyen. 2021a. Trankit: A light-weight transformer-based toolkit for multilingual natural language processing. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations*, pages 80–90, Online. Association for Computational Linguistics.

Minh Van Nguyen, Viet Dac Lai, Amir Pouran Ben Veyseh, and Thien Huu Nguyen. 2021b. Trankit: A light-weight transformer-based toolkit for multilingual natural language processing. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations*.

Joakim Nivre. 2009. Non-projective dependency parsing in expected linear time. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 351–359, Suntec, Singapore. Association for Computational Linguistics.

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. 2022. Training language models to follow instructions with human feedback.

Adrien Pavao, Isabelle Guyon, Anne-Catherine Letournel, Xavier Baró, Hugo Escalante, Sergio Escalera, Tyler Thomas, and Zhen Xu. 2022. Codalab competitions: An open source platform to organize scientific challenges. Technical report, Université Paris-Saclay.

Jonas Pfeiffer, Andreas Rücklé, Clifton Poth, Aishwarya Kamath, Ivan Vulić, Sebastian Ruder, Kyunghyun Cho, and Iryna Gurevych. 2020a. AdapterHub: A framework for adapting transformers. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 46–54, Online. Association for Computational Linguistics.

Jonas Pfeiffer, Ivan Vulić, Iryna Gurevych, and Sebastian Ruder. 2020b. MAD-X: An Adapter-Based Framework for Multi-Task Cross-Lingual Transfer. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7654–7673, Online. Association for Computational Linguistics.

Adam Przepiórkowski, Mirosław Bańko, Rafał L. Górski, and Barbara Lewandowska-Tomaszczyk, editors. 2012. *Narodowy Korpus Języka Polskiego*. Wydawnictwo Naukowe PWN, Warsaw.

Piotr Przybyła. 2022. LAMBO: Layered Approach to Multi-level BOundary identification.

Peng Qi, Yuhao Zhang, Yuhui Zhang, Jason Bolton, and Christopher D. Manning. 2020. Stanza: A python natural language processing toolkit for many human languages. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 101–108, Online. 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*, pages 45–54, Brussels, Belgium. Association for Computational Linguistics.

Devendra Sachan, Yuhao Zhang, Peng Qi, and William L. Hamilton. 2021. Do syntax trees help pre-trained transformers extract information? In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 2647–2661, Online. Association for Computational Linguistics.

Djamé Seddah, Reut Tsarfaty, Sandra Kübler, Marie Candito, Jinho D. Choi, Richárd Farkas, Jennifer Foster, Iakes Goenaga, Koldo Gojenola Galletebeitia, Yoav Goldberg, Spence Green, Nizar Habash, Marco Kuhlmann, Wolfgang Maier, Joakim Nivre, Adam Przepiórkowski, Ryan Roth, Wolfgang Seeker, Yannick Versley, Veronika Vincze, Marcin Woliński, Alina Wróblewska, and Eric Villemonte de la Clergerie. 2013. Overview of the SPMRL 2013 shared task: A cross-framework evaluation of parsing morphologically rich languages. In *Proceedings of the Fourth Workshop on Statistical Parsing of Morphologically-Rich Languages*, pages 146–182, Seattle, Washington, USA. 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 Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 4290–4297, Portorož, Slovenia. European Language Resources Association (ELRA).

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*, pages 88–99, Vancouver, Canada. Association for Computational Linguistics.

Kai Sun, Richong Zhang, Samuel Mensah, Yongyi Mao, and Xudong Liu. 2019. Aspect-level sentiment analysis via convolution over dependency tree. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5679–5688, Hong Kong, China. Association for Computational Linguistics.

Łukasz Szałkiewicz and Adam Przepiórkowski. 2012. Anotacja morfoskładniowa. In Adam Przepiórkowski, Mirosław Bańko, Rafał L. Górski, and Barbara Lewandowska-Tomaszczyk, editors, *Narodowy Korpus Języka Polskiego*, pages 59–96. Wydawnictwo Naukowe PWN, Warsaw.

Shikhar Vashishth, Rishabh Joshi, Sai Suman Prayaga, Chiranjib Bhattacharyya, and Partha Talukdar. 2018. RESIDE: Improving distantly-supervised neural relation extraction using side information. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1257–1266, Brussels, Belgium. Association for Computational Linguistics.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355, Brussels, Belgium. Association for Computational Linguistics.

Yufei Wang, Mark Johnson, Stephen Wan, Yifang Sun, and Wei Wang. 2019. How to best use syntax in semantic role labelling. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5338–5343, Florence, Italy. Association for Computational Linguistics.

Jakub Waszczuk. 2012. Harnessing the crf complexity with domain-specific constraints. the case

of morphosyntactic tagging of a highly inflected language. In *Proceedings of COLING 2012*, pages 2789–2804.

Jakub Waszczuk, Witold Kieraś, and Marcin Woliński. 2018. Morphosyntactic disambiguation and segmentation for historical polish with graph-based conditional random fields. In *International Conference on Text, Speech, and Dialogue*, pages 188–196. Springer.

Marcin Woliński. 2014. Morfeusz reloaded. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation, LREC 2014*, pages 1106–1111. European Language Resources Association (ELRA).

Marcin Woliński. 2019. *Automatyczna analiza składnikowa języka polskiego*. Wydawnictwa Uniwersytetu Warszawskiego, Warsaw.

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*, pages 1–21, Brussels, Belgium. 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 Gokirmak, Anna Nedoluzhko, Silvie Cinková, Jan Hajič jr., Jaroslava Hlaváčová, Václava Kettnerová, Zdeňka Urešová, Jenna Kanerva, Stina Ojala, Anna Missilä, Christopher D. 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 Strnadová, 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*, pages 1–19, Vancouver, Canada. Association for Computational Linguistics.

Meishan Zhang, Zhenghua Li, Guohong Fu, and Min Zhang. 2019. Syntax-enhanced neural machine translation with syntax-aware word representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1151–1161, Minneapolis, Minnesota. Association for Computational Linguistics.

Yuhao Zhang, Peng Qi, and Christopher D. Manning. 2018. Graph convolution over pruned dependency trees improves relation extraction. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2205–2215, Brussels, Belgium. Association for Computational Linguistics.

9. Language Resource References

Alexis Conneau and Kartikay Khandelwal and Naman Goyal and Vishrav Chaudhary and Guillaume Wenzek and Francisco Guzmán and Edouard Grave and Myle Ott and Luke Zettlemoyer and Veselin Stoyanov. 2019. *XLM-RoBERTa*. Hugging Face.

Grave, Edouard and Bojanowski, Piotr and Gupta, Prakhar and Joulin, Armand and Mikolov, Tomas. 2018. *fastText*. Facebook.

Kłeczek, Dariusz. 2021. *Polbert*. Hugging Face.

Lynn, Teresa and Foster, Jennifer and McGuinness, Sarah and Walsh, Abigail and Phelan, Jason and Scannell, Kevin. 2015. *Irish Dependency Treebank (UD Irish-IDT)*. Universal Dependencies Consortium. PID http://hdl.handle.net/11234/1-4611.

Mroczkowski, Robert and Rybak, Piotr and Wróblewska, Alina and Gawlik, Ireneusz. 2021. *HerBERT*. Hugging Face.

Przepiórkowski, Adam and Bańko, Mirosław and Górski, Rafał L. and Lewandowska-Tomaszczyk, Barbara. 2018. *National Corpus of Polish*. Institute of Computer Science.

Shen, Mo and McDonald, Ryan and Zeman, Daniel and Qi, Peng. 2019. *Chinese Dependency Treebank (UD Chinese-GSD)*. Universal Dependencies Consortium. PID http://hdl.handle.net/11234/1-4611.

Wróblewska, Alina. 2018. *Polish Dependency Bank (UD Polish-PDB)*. Universal Dependencies Consortium. PID http://hdl.handle.net/11234/1-5150.