HuSpaCy: an industrial-strength Hungarian natural language processing toolkit

György Orosz, Zsolt Szántó,
Péter Berkecz, Gergő Szabó, Richárd Farkas
gyorgy@orosz.link
{szantozs,rfarkas}@inf.u-szeged.hu

Institute of Informatics, University of Szeged
2. Árpád tér, Szeged, Hungary

Abstract. Although there are a couple of open-source language processing pipelines available for Hungarian, none of them satisfies the requirements of today's NLP applications. A language processing pipeline should consist of close to state-of-the-art lemmatization, morphosyntactic analysis, entity recognition and word embeddings. Industrial text processing applications have to satisfy non-functional software quality requirements, what is more, frameworks supporting multiple languages are more and more favored. This paper introduces HuSpaCy, an industry-ready Hungarian language processing toolkit. The presented tool provides components for the most important basic linguistic analysis tasks. It is open-source and is available under a permissive license. Our system is built upon spaCy's NLP components resulting in an easily usable, fast yet accurate application. Experiments confirm that HuSpaCy has high accuracy while maintaining resource-efficient prediction capabilities.

1   Introduction

Basic natural language processing tasks such as tokenization, sentence splitting, part-of-speech tagging, lemmatization, dependency parsing and named entity recognition are amongst the most widely studied problems in natural language processing. Several text analysis applications have been developed during the last decades for both English and other less-resourced languages such as Hungarian. However, a large majority of them solely focus on achieving high scores on artificial benchmarks and ignore the importance of practical usability.

In this paper we introduce HuSpaCy, an industry-strength Hungarian text processing pipeline capable of parsing and tagging texts with high accuracy on limited computational resources. Our system is built upon spaCy's[1] NLP components, which means that it is fast, has a rich ecosystem of NLP applications and extensions, comes with extensive documentation and a well-known API.

---

[1] https://spacy.io/

First, we give an overview of the underlying models, then rigorous evaluation is presented using various datasets. Finally, experiments are presented confirming that HuSpaCy has high accuracy in many subtasks while maintaining resource-efficient prediction capabilities.

2 Background

2.1 Demands for a language processing pipeline in the 2020s

Starting from the release of the Penn Treebank (Marcus et al., 1993) in 1992, the research community developed language processing tools for particular tasks, like tokenization, part-of-speech tagging etc. These tools are usually run in a sequence and form a pipeline. In the 2000s, many language-specific corpora and treebanks were developed along with such pipelines. Hungarian was among the best supported languages (Simon et al., 2012) ten years ago.

In the early 2010s, Universal PoS (Petrov et al., 2012) and Universal Dependency (Nivre et al., 2016) labeling schemata were developed with the goals of "cross-linguistically consistent treebank annotation for many languages" and "facilitating multilingual parser development, cross-lingual learning, and parsing research from a language typology perspective." Many language-specific pipelines changed their representations to these universal annotation schema, but most of them stayed in their own software architecture. Industrial NLP applications are frequently multi-lingual, i.e. the same NLP task has to be solved in several languages. The demand for standardization over languages is high in commercial partners. Beyond universal PoS and dependency annotations, companies who are not NLP experts but want to apply language processing tools prefer multilingual software frameworks and make business decisions to support Hungarian, based on the availability in multilingual frameworks.

The last five years of NLP are dominated by neural language models (NLM) and the applications based on them (Young et al., 2018). Academic research has introduced various deep learning methods outperforming previous state of the art in many areas. Such systems usually employ a single neural network providing end-to-end NLP solutions without the need for specific pipeline steps. Also, several pre-trained multilingual NLMs are becoming available which provide standardized solutions for many languages. Regardless of the multilinguality and the high accuracy of deep learning solutions, a lot of critiques have been raised by real-world industrial NLP projects recently.

These are as follows: deep learning solutions often require far more computational resources compared to classic solutions. They heavily rely on GPU acceleration along with significant memory consumption. What is more, their running costs are usually 10 or even 100 times higher than that of alternative solutions. These drawbacks are questioning the commercial return of the accuracy gain. We also note that modern NLP pipelines consist of static word embedding representations and use deep learning for individual pipeline steps as well, hence, the advantage of large neural end-to-end systems might be very small.

Another industrial demand about language processing systems is to provide human-readable output. Most of the industrial applications are fully or partially rule-based solutions, as (enough) training data for a pure machine learning solution is not available. And there is no free lunch! Each and every real-world application has its own requirements. Rule-based components of these real-world applications require language-specific representations which can be used for defining rules. Such human-readable representations consist of tokens, lemmata, part-of-speech tags, morphological features, dependency parse trees and named entities. Static word embeddings are often integral parts of industrial applications, as many practical algorithms (e.g. semantic textual similarity methods) heavily rely on them.

2.2 Requirements for industrial-strength language processing pipelines

Considering decades of experience of practical NLP applications, developing an "industrial-strength" text processing system is a challenging task. First of all, such a tool should tackle the most important text preprocessing tasks including tokenization, sentence splitting, PoS tagging, lemmatization, dependency parsing, named entity recognition and word embedding representation.

Next, the application has to be accurate enough for real world scenarios while it should be resource conscious at the same time. Furthermore, an industry focused system should be developer friendly, customizable and easy-to-integrate, as NLP modules are integral parts of a larger system in practical applications. These requirements imply that solid documentation should be available as well. Moreover, it is often desired that the underlying machine learning model(s) should be reproducible and controllable.

Last but not least, modern NLP applications are usually multilingual, thus compatibility with international annotation standards (Petrov et al., 2012; Nivre et al., 2016; de Marneffe et al., 2021; McCarthy et al., 2020) is necessary. Moreover, it is also preferred to be easily usable through a well-known multilingual toolkit.

2.3 The landscape of Hungarian language processing pipelines

Up until recently, only a few text processing applications were focused on meeting these criteria even for English. When spaCy was released in 2015 (Honnibal, 2015), it was one one of the first tools targeting industrial applications mainly. Its authors created an unprecedented tool which offers near state-of-the-art accuracy while being an order of magnitude faster than other tools available. SpaCy also comes with an intuitive API, has detailed documentation, and also fits well into the Python ecosystem of machine learning tools. What is more, it is easily deployable and offers built-in syntax and entity visualization tools as well.

The landscape of the Hungarian text processing systems is similar to that of English before the "industrial NLP revolution". There are a number of standalone

text analysis tools[2] (Simon et al., 2012) capable of performing individual text processing tasks, but they often do not play well with each other.

In contrast, there are only a few attempts at providing industrial Hungarian pipelines. One of them is `magyarlanc` (Zsibrita et al., 2013) which is a Java based system consisting of state-of-the-art pipeline steps, which were adapted and extended from various libraries. It was designed to serve industrial applications, what is more, a lot of effort has been made on software quality, speed, memory efficiency and customizability. It performs tokenization, sentence boundary detection (SBD), PoS tagging, lemmatization and dependency parsing, but lacks entity recognition and word embeddings. It uses the version 1 of the Universal Dependency (UD) annotations. Although the tool is still used in real world commercial applications, it is not maintained for years.

There is only one other attempt to provide a unified framework for Hungarian text processing tools: `emtsv` (Simon et al., 2020; Indig et al., 2019a,b) (and its predecessor e-magyar (Váradi et al., 2018; Váradi et al., 2017)) is a result of a multi-institute collaboration project aiming to integrate existing NLP toolkits into a single application. Unfortunately, neither computational efficiency nor developer ergonomics were amongst the main goals of the project. Although `emtsv` can yield Universal morphosyntactic annotations through conversion, it is rather inaccurate. What is more, it is not designed to efficiently deal with word vectors, therefore no such facility is available in the system.

Talking of Hungarian-specific pipelines, we must mention the contenders of the recent multilingual CoNLL text parsing competitions (Zeman et al., 2017, 2018). There were numerous submissions, but Stanza (Qi et al., 2020) and UDPipe (Straka, 2018) are by far the most popular freely available off-the-shelf applications. These tools provide morphological and syntactical analysis of raw texts for many languages, but lack entity annotations. Accuracy scores vary across tools, but all of them are limited by the small size of the publicly available UD annotated gold standard corpora.

	NER	Word embeddings	High throughput	Part of a multilingual pipeline	Free for commercial usage
`magyarlanc`	-	-	✓	-	✓
`emtsv`	✓	-	-	-	-
UDPipe	-	-	✓	✓	-
Stanza	✓	✓	-	✓	✓

Table 1. Hungarian language processing pipelines evaluated with regards of requirements of industrial applicability.

---

[2] cf. https://github.com/oroszgy/awesome-hungarian-nlp

Table 1 summarizes the landscape of the most important Hungarian language processing pipelines and show how they meet the requirements of today's NLP applications. We must note that `emtsv` does not have any restriction for using it in commercial applications, although some of its most important components have very restrictive licenses (e.g. `emMorph`). All in all, we can say that none of them is easily applicable in industrial settings.

We present HuSpaCy, a new industry-ready Hungarian natural language processing toolkit. It provides all the aforementioned basic text processing modules with high accuracy. The underlying models are optimized to be light on memory consumption and CPU usage. The presented tool is open source[3] and is available under the permissive CC-BY-SA-4.0 license. Our system is built on top of spaCy's infrastructure, thus extensive documentation, debugging tools, an ergonomic API and a flourishing ecosystem are already provided.

3   HuSpaCy internals

This section introduces the NLP algorithms behind the presented tool. As our system is built on spaCy's architecture, we mainly relied on its symbolic and ML-based text processing infrastructure. The following paragraphs give a high-level overview of the framework utilized and also describes the contributions of this work.

3.1   Tokenization

HuSpaCy builds on spaCy's (Honnibal, 2021) tokenization infrastructure which works as follows: first the input text is split on whitespaces, then token boundaries are identified by splitting prefixing or suffixing character sequences. To make this algorithm viable for Hungarian, we extended it with language specific prefix and suffix splitting rules. Furthermore, we had to deal with the ambiguity of tokens around full stops, thus an extensive abbreviation list has been incorporated to increase the module's accuracy. During this process we mostly relied on the test cases of HunToken (Németh and Zséder, 2013) to fine-tune the algorithm.

3.2   Morphosyntactic tagging, sentence splitting and parsing

Sentence boundaries, dependency parse trees, PoS tags, and the corresponding morphosyntactic features are predicted by a multitask deep learning model of the underlying NLP framework. SpaCy's machine learning approach can be summarized as "embed, encode, attend, predict" (Honnibal, 2016) which our system adapts for its tagging and parsing components.

---

[3] https://github.com/huspacy/huspacy

Tokens are embedded using the concatenation of static (pretrained) word vectors and ones learned during the task-specific training process. We use a publicly[4] available 300d word embedding which has been trained on the Hungarian Webcorpus (Halácsy et al., 2004) and a snapshot of the Hungarian Wikipedia with CBOW methodology (Mikolov et al., 2013). Task specific word vectors are 256 wide consisting 64 dimensional embeddings of the tokens' prefixes, suffixes, shapes and the lowercase forms. To make such computation efficient, feature hashing is extensively applied to all kinds of input strings.

During the encoding part, vectors are passed through a four deep stacked CNN encoder (Lecun et al., 1998) which uses residual connections and is accompanied with maxout pooling[5] (Honnibal, 2017). Efficient prediction is guaranteed by the underlying greedy tagger consisting only of a linear and a softmax layer. As for the dependency parsing, an arc-eager transition system (Honnibal et al., 2013) is utilized, which shares weights with the tagger model through multitask learning.

Finally, sentence boundary recognition is formalized as a sequence tagging problem where tokens are tagged with a binary label indicating the first token of a sentence. This component is an integral part of the multitask architecture, thus it also shares its neural model with the parser and the morphosyntactic tagger.

3.3   Lemmatization

SpaCy's default lemmatization model is mainly designed for English. It is not suitable for morphologically complex languages such as Hungarian as it only uses lookup tables. Hence, we decided to look for a more sophisticated solution and adapted the Lemmy toolkit (Kristiansen, 2019) which is an open-source Python implementation of the CST rule-learning engine (Jongejan and Dalianis, 2009). To improve its accuracy we incorporated three minor modifications. First, prefixing numbers of numeric tokens are masked to help the engine in case of inflected numbers. (For example the masked token of '2021-ben' becomes '0000-ben'.) Second, we enforce lowercasing of sentence starting tokens if they are not proper nouns. Finally, if there are multiple lemma candidates available for a given (word, tag) pair, we pick the one with the highest frequency on the training dataset.

3.4   Named entity recognition

SpaCy's entity recognizer is built on the transition-based parser architecture described in Section 3.2 (similarly to Lample et al. (2016)). However, there are two key differences compared to the system of Lample et al. (2016). The first is that the set of possible transition actions reflects the BILOU tagging scheme.

---

[4] https://github.com/oroszgy/hunlp-resources/releases/tag/webcorpuswiki_word2vec_v0.1

[5] The pooling step is considered to be the "attention" mechanism.

This trick allows the model to have better discrimination ability between different entity classes, furthermore it makes the learning problem easier. Second, the state vector computation includes clues not just from the surrounding words but the tokens of previous entities as well. The sequence tagger model uses BILOU tags for encoding entity boundaries and the decoder is built on a greedy softmax layer similar to that of the morphosyntactic tagger.

4   Experiments and results

4.1   Text parsing

In order to benchmark HuSpaCy, we performed a series of experiments comparing its performance with the most popular off-the-shelf pipelines available. Evaluation is carried out on the test set of the Hungarian Universal Dependencies Corpus (de Marneffe et al., 2021) by using the evaluation script of the CoNLL 2018 Shared Task[6].

Three popular text processing tools have been selected for comparison. emtsv is a Hungarian specific pipeline integrating state-of-the-art NLP components, UDPipe is used as a baseline system in CoNLL competitions, while Stanza has high scores on parsing UD corpora. All systems are used as black boxes meaning they have not been retrained or fine-tuned.

Up until now, there has only been a single Hungarian corpus (Csendes et al., 2004) having both morphosyntactic and dependency parse annotations. What is more, UD annotations are available only in a rather small subcorpus of it (de Marneffe et al., 2021). As PoS and morphosyntactic labels can be transcribed automatically from the Hungarian-specific formalism to UD with high accuracy, additional silver standard data can be utilized to train taggers.

In case of HuSpaCy, we applied a two-step learning strategy [7] to best utilize all available training data. In the first step, the tagger and the SBD components are pre-trained on the whole transcribed SZC[8]. This is followed by a fine-tuning step on the gold standard UD dataset where dependency annotations are also learned. To allow fair comparison with Stanza and UDPipe, a single step model relying solely on the UD data is also involved in the evaluation.

For similar reasons, the lemmatizer has been trained with two configurations. First we used only the training set of the Hungarian UD corpus, then we allowed the tool to learn from the whole transcribed Szeged Corpus (except the sentences overlapping with either our test or development sets).

---

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

[7] Hyperparameters of the models are available in the tool's repository (tag v0.4.2) as configuration files.

[8] When we refer to the Szeged Corpus as a training set, we mean all the sentences that are not part of the development or the test set of the Universal Dependencies corpus.

The authors of Stanza[9] and UDPipe[10] have already published their tools' accuracy on the Hungarian UD corpus, however the same is not true for `emtsv`. To evaluate the latter toolkit we used the following (default) configuration to produce an UD-compatible output: `emToken, emMorph, emLem, emTag, emmorph2ud, emDep, emConll`. While `emtsv` can provide parse trees, their annotation schema is not compatible with that of the Universal Dependencies, hence, its output is not evaluable. We must also note that comparison with the `emtsv`'s tagger and lemmatizer might not be fair, as this tool was trained on a different train-test split which might conflict with ours. (There is a high chance that its training data overlaps with the sentences of our test set.)

	Tokenization	Sentence splitting
Stanza	99.87%	97.00%
UDPipe	99.80%	95.90%
`emtsv`	99.77%	98.67%
HuSpaCy (UD)	99.89%	97.66%
HuSpaCy (SZC)		97.54%

Table 2. Tokenization and sentence boundary detection F1 scores on the test of the Hungarian UD Corpus.

F1 scores in Table 2 suggest that tokenization is easily handled by all of the systems, although HuSpaCy is marginally better compared to the rest of the tools. Sentence boundary detection is a more complex task, where language specific knowledge is necessary. This can be either built into the system (as it is the case with `emtsv`) or learned by a ML model. Numbers show that both approaches can yield satisfactory SBD components, although the rule-based solution of `emtsv` stands out followed by the tagging approach of our pipeline.

	PoS acc.	Morph. acc.	UAS	LAS
Stanza	96.03%	93.76%	83.62%	78.86%
UDPipe v1	90.60%	88.50%	72.80%	67.20%
`emtsv`	89.19%	89.12%	–	
HuSpaCy (UD)	94.70%	89.03%	79.03%	73.17%
HuSpaCy (SZC)	96.58%	93.23%	79.39%	74.22%

Table 3. Comparison of tagging accuracy and attachment scores of the benchmarked pipelines on the test set of the Hungarian UD Corpus.

---

[9] https://stanfordnlp.github.io/stanza/performance.html
[10] https://ufal.mff.cuni.cz/udpipe/1/models

Tagging accuracy and attachment scores are presented in Table 3. Results show that Stanza is a clear winner in dependency parsing while the PoS tagging score of HuSpaCy (the one using additional training data) is the highest one. It can be seen that the usage of the extra silver standard data yields better performance for our models both during tagging and dependency parsing. UDPipe and `emtsv` have relatively low scores: the results of UDPipe are not surprising (cf. Zeman et al. (2018)), but `emtsv`'s scores are unexpected given that it is built upon state-of-the-art morphosyntactic tagging facilities (Orosz and Novák, 2013).

	Accuracy
Stanza	94.25%
UDPipe v1	88.50%
`emtsv`	94.94%
HuSpaCy (UD)	94.82%
HuSpaCy (SZC)	95.53%

Table 4. Lemmatization accuracy of NLP pipelines measured on the test set of the Hungarian UD Corpus. HuSpaCy (UD) uses the same setting as its contenders, while HuSpaCy (SZC) builds on the whole Szeged Corpus for training.

Lemmatization results in Table 4 show that all the systems except UDPipe are accurate enough. HuSpaCy trained on the full Szeged Corpus stands out, its score is more than 0.5% higher than the second best system (`emtsv`). The best configuration of HuSpaCy scores more than 0.5% higher than the one trained solely on the UD dataset.

4.2   Named entity recognition

Comparing NER components is not as straightforward as it is for the parsing subtasks. There are multiple evaluation datasets, but there is no consensus between researchers on their usage. NYTK-NerKor (NerKor) (Simon and Vadász, 2021) is a relatively new corpus consisting of 1 million tokens, while SzegedNER (Szarvas et al., 2006a) is a 200,000 token subset of the Szeged Corpus. Simon et al. (2022) uses the former dataset to benchmark some of the most popular tools, while previous work mainly rely on the latter one. UDPipe does not have a NER component, thus we cannot include it in this investigation. As for `emtsv`, its entity recognizer was trained using the whole SzegedNER corpus, its comparison against other tools would not be fair.

HuSpaCy's entity recognition capabilities are benchmarked in this work on both corpora using the same train-test splits as Szarvas et al. (2006b) and Simon et al. (2022) suggest. As Hungarian entity recognition datasets share the same

tagset and rely on similar annotation guides, it is possible to train models using both corpora. In this regards, we follow the work of Simon et al. (2022) and evaluate HuSpaCy on the combined corpus as well. We also include results of previous entity recognition attempts so as to put our results in context. One of the first systems was developed by Szarvas et al. (2006b), which utilizes decision trees for tackling the problem. Next, there is HunTag (Recski and Varga, 2009; Simon, 2013), which is a statistical tagger utilizing a linear model combined with Hidden Markov models. Simon (2013) also showed that it is possible to improve on the F1 score of the base system by incorporating silver standard data. Most recently, Nemeskey (2020a) developed an entity recognizer on top of Hungarian BERT models (Nemeskey, 2020b) achieving state-of-the-art results.

	SzegedNer	NerKor	Combined
Simon (2013)	95.06%	–	–
Szarvas et al. (2006b)	94.77%	–	–
emBERT	97.40%	92.09%	92.99%
Stanza	91.78%	80.53%	83.75%
HuSpaCy	95.31%	80.75%	83.46%

Table 5. Comparison of entity recognition F1 scores on the SzegedNER test set (Szarvas et al., 2006b), on the NerKor test set and on the combined test.

Table 5 contains F1 scores of all the entity recognizers mentioned above. It can be seen that the BERT-based model achieves the highest scores on all of the datasets with a large margin. However, these models are also well-known for their enormous computational costs. HuSpaCy is the second best contender on SzegedNER, although its performance is on pair with Stanza on other datasets. emBERT's results are outstanding when NerKor is involved in the comparison. As Simon et al. concludes these measurements are in accordance with similar English NER benchmarks (cf. Qi et al. (2020)). Pretrained transformer-based models often yield significantly higher performance scores compared to other sequence tagging approaches due to the underlying attention mechanism and the their model's increased capacity. But there is no free lunch, higher accuracy comes with significantly increased prediction costs.

The final model of HuSpaCy builds on the weights of a pretrained neural tagger (using the strategy described in Section 4.1) yielding 84.56% F1 on the combined dataset. This result is a significant improvement compared to Stanza's score and also confirms the usefulness of additional silver standard training data usage for spaCy's multitask neural model.

4.3   Resource usage

Resource usage such as memory consumption and processing speed is an important aspect of practical text processing systems, thus we benchmarked[11] text parsing pipelines (cf. Section 4.1) in this respect. In order to have a fair comparison, we configured all systems to perform only tokenization, sentence splitting, PoS tagging, lemmatization and dependency parsing. As timing measurements should ignore model loading times, Stanza and UDPipe were used by their Python interfaces, while `emtsv` was utilized through its REST API. We used the UD test set to measure throughput and peak memory consumption.

	Throughput (tokens/sec)	Memory usage (GB)
Stanza	222	0.9
UDPipe	1741	0.4
`emtsv`	122	3.9
HuSpaCy	2612	2.1

Table 6. Throughput (measured in tokens/second) and peak memory consumption of benchmarked NLP pipelines.

Table 6 presents computational efficiency measures suggesting that our system has the highest throughput amongst all the tools. HuSpaCy is almost 50% faster than UDPipe, while producing significantly better parses. As regards Stanza, there is a huge tradeoff on having the best dependency parser: it is almost 8 times slower than UDPipe and more than 10 times slower compared to our system.

Memory consumption of the pipelines are acceptable as all of them could easily fit in a modern computer's RAM. Our tool has the highest memory usage which is due to its 300-dimensional word vectors. In comparison, Stanza is the only other tool having word embeddings, but its vectors' sizes are limited to 100d.

5   Conclusions

We presented HuSpaCy, a new industry-ready Hungarian language processing pipeline that is open source and is freely available. While previous approaches have failed to provide a tool which can be easily used to solve practical text processing problems, our system builds on the solid foundations of an industrial NLP framework. We presented how our toolkit utilizes spaCy's underlying ML

---

[11] All experiments were performed on a computer having an Intel Core i7-8750H CPU and 16 GB RAM running Ubuntu Linux 20.04 LTS.

models to provide all the basic language analysis components. We performed various experiments proving that our system has high accuracy in many text processing tasks while using only moderate computation resources.

As results show, the accuracy of HuSpaCy's dependency parser needs further improvements. Further advancement opportunities lie in fine-tuning the NER model and in using a new neural lemmatizer.

In summary, this study described a new freely available tool which is suitable for real-world industrial applications.

Acknowledgements

Bibliography

Csendes, D., Csirik, J., Gyimóthy, T.: The Szeged Corpus: A POS tagged and syntactically annotated Hungarian natural language corpus. In: Proceedings of the 5th International Workshop on Linguistically Interpreted Corpora LINC 2004 at The 20th International Conference on Computational Linguistics COLING 2004. pp. 19–23 (2004)

Halácsy, P., Kornai, A., Németh, L., Rung, A., Szakadát, I., Trón, V.: Creating open language resources for Hungarian. In: Proceedings of the Fourth International Conference on Language Resources and Evaluation (LREC'04). European Language Resources Association (ELRA), Lisbon, Portugal (May 2004), http://www.lrec-conf.org/proceedings/lrec2004/pdf/525.pdf

Honnibal, M.: Introducing spaCy (Feb 2015), https://explosion.ai/blog/introducing-spacy

Honnibal, M.: Embed, encode, attend, predict: The new deep learning formula for state-of-the-art NLP models (Nov 2016), https://explosion.ai/blog/deep-learning-formula-nlp

Honnibal, M.: Multi-task cnn for parser, tagger and ner (issue #1057) (May 2017), https://github.com/explosion/spaCy/issues/1057

Honnibal, M.: Tokenization - spaCy Usage Documentation (Nov 2021), https://spacy.io/usage/linguistic-features

Honnibal, M., Goldberg, Y., Johnson, M.: A non-monotonic arc-eager transition system for dependency parsing. In: Proceedings of the Seventeenth Conference on Computational Natural Language Learning. pp. 163–172. Association for Computational Linguistics, Sofia, Bulgaria (Aug 2013), https://aclanthology.org/W13-3518

Indig, B., Sass, B., Simon, E., Mittelholcz, I., Vadász, N., Makrai, M.: One format to rule them all – the `emtsv` pipeline for Hungarian. In: Proceedings of the 13th Linguistic Annotation Workshop. pp. 155–165. Association for Computational Linguistics, Florence, Italy (aug 2019a), `https://www.aclweb.org/anthology/W19-4018`

Indig, B., Sass, B., Simon, E., Mittelholcz, I., Kundráth, P., Vadász, N.: `emtsv` — egy formátum mind felett. In: Berend, G., Gosztolya, G., Vincze, V. (eds.) XV. Magyar Számítógépes Nyelvészeti Konferencia (MSZNY 2019). pp. 235–247. Szegedi Tudományegyetem Informatikai Tanszékcsoport, Szeged (2019b)

Jongejan, B., Dalianis, H.: Automatic training of lemmatization rules that handle morphological changes in pre- , in- and suffixes alike. In: Joint conference of the 47th Annual Meeting of the Association for Computational Linguistics and the 4th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing. pp. 145–153 (2009)

Kristiansen, S.L.: lemmy: Lemmy a lemmatizer for Danish and Swedish (Apr 2019), `https://github.com/sorenlind/lemmy`

Lample, G., Ballesteros, M., Subramanian, S., Kawakami, K., Dyer, C.: Neural Architectures for Named Entity Recognition (2016)

Lecun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. Proceedings of the IEEE 86(11), 2278–2324 (1998)

Marcus, M.P., Marcinkiewicz, M.A., Santorini, B.: Building a large annotated corpus of english: The penn treebank. Comput. Linguist. 19(2), 313—330 (jun 1993)

de Marneffe, M.C., Manning, C.D., Nivre, J., Zeman, D.: Universal Dependencies. Computational Linguistics 47(2), 255–308 (07 2021), `https://doi.org/10.1162/coli_a_00402`

McCarthy, A.D., Kirov, C., Grella, M., Nidhi, A., Xia, P., Gorman, K., Vylomova, E., Mielke, S.J., Nicolai, G., Silfverberg, M., Arkhangelskiy, T., Krizhanovsky, N., Krizhanovsky, A., Klyachko, E., Sorokin, A., Mansfield, J., Ernštreits, V., Pinter, Y., Jacobs, C.L., Cotterell, R., Hulden, M., Yarowsky, D.: UniMorph 3.0: Universal Morphology. In: Proceedings of the 12th Language Resources and Evaluation Conference. pp. 3922–3931. European Language Resources Association, Marseille, France (May 2020), `https://aclanthology.org/2020.lrec-1.483`

Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space (2013)

Nemeskey, D.M.: Egy `emBERT` próbáló feladat. In: XVI. Magyar Számítógépes Nyelvészeti Konferencia (MSZNY2020). pp. 409–418. Szeged (2020a)

Nemeskey, D.M.: Natural Language Processing Methods for Language Modeling. Ph.D. thesis, Eötvös Loránd University (2020b)

Nivre, J., de Marneffe, M.C., Ginter, F., Goldberg, Y., Hajič, J., Manning, C.D., McDonald, R., Petrov, S., Pyysalo, S., Silveira, N., Tsarfaty, R., Zeman, D.: Universal Dependencies v1: A multilingual treebank collection. In: Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16). pp. 1659–1666. European Language Resources Associa-

tion (ELRA), Portorož, Slovenia (May 2016), https://aclanthology.org/L16-1262

Németh, L., Zséder, A.: huntoken: word and sentence tokenizer (2013), https://github.com/zseder/huntoken

Orosz, G., Novák, A.: PurePos 2.0: a hybrid tool for morphological disambiguation. In: Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2013). p. 539–545. INCOMA Ltd. Shoumen, Hissar, Bulgaria (2013)

Petrov, S., Das, D., McDonald, R.: A universal part-of-speech tagset. In: Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12). pp. 2089–2096. European Language Resources Association (ELRA), Istanbul, Turkey (May 2012), http://www.lrec-conf.org/proceedings/lrec2012/pdf/274_Paper.pdf

Qi, P., Zhang, Y., Zhang, Y., Bolton, J., Manning, C.D.: 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 (2020)

Recski, G., Varga, D.: A Hungarian NP Chunker. The Odd Yearbook. ELTE SEAS Undergraduate Papers in Linguistics pp. 87–93 (2009)

Simon, E., Lendvai, P., Németh, G., Olaszy, G., Vicsi, K.: A magyar nyelv a digitális korban – The Hungarian Language in the Digital Age. Georg Rehm and Hans Uszkoreit (Series Editors): META-NET White Paper Series, Springer (2012)

Simon, E.: Approaches to Hungarian Named Entity Recognition. Ph.D. thesis, PhD School in Cognitive Sciences, Budapest University of Technology and Economics (2013)

Simon, E., Indig, B., Kalivoda, Á., Mittelholcz Iván, S.B., Vadász, N.: Újabb fejlemények az e-magyar háza táján. In: Berend, G., Gosztolya, G., Vincze, V. (eds.) XVI. Magyar Számítógépes Nyelvészeti Konferencia. pp. 29–42. Szegedi Tudományegyetem Informatikai Tanszékcsoport, Szeged (2020)

Simon, E., Vadász, N.: Introducing nytk-nerkor, A gold standard hungarian named entity annotated corpus. In: Ekstein, K., Pártl, F., Konopík, M. (eds.) Text, Speech, and Dialogue - 24th International Conference, TSD 2021, Olomouc, Czech Republic, September 6-9, 2021, Proceedings. Lecture Notes in Computer Science, vol. 12848, pp. 222–234. Springer (2021)

Simon, E., Vadász, N., Lévai, D., Nemeskey, D., Szántó, Z., Orosz, G.: Az NYTK-NerKor több szempontú kiértékelése. In: XVIII. Magyar Számítógépes Nyelvészeti Konferencia. pp. 375–388 (2022)

Straka, M.: 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. pp. 197–207. Association for Computational Linguistics, Brussels, Belgium (Oct 2018), https://www.aclweb.org/anthology/K18-2020

Szarvas, G., Farkas, R., Felföldi, L., Kocsor, A., Csirik, J.: A highly accurate named entity corpus for Hungarian. In: Proceedings of the Fifth International Conference on Language Resources and Evaluation (LREC'06). Euro-

pean Language Resources Association (ELRA), Genoa, Italy (May 2006a), http://www.lrec-conf.org/proceedings/lrec2006/pdf/365_pdf.pdf

Szarvas, G., Farkas, R., Kocsor, A.: A multilingual named entity recognition system using boosting and C4.5 decision tree learning algorithms. In: International Conference on Discovery Science. pp. 267–278. Springer (2006b)

Váradi, T., Simon, E., Sass, B., Gerőcs, M., Mittelholtz, I., Novák, A., Indig, B., Prószéky, G., Vincze, V.: Az e-magyar digitális nyelvfeldolgozó rendszer. In: Berend, G., Gosztolya, G., Vincze, V. (eds.) XIII. Magyar Számítógépes Nyelvészeti Konferencia (MSZNY 2017). pp. 49–60. Szegedi Tudományegyetem Informatikai Tanszékcsoport, Szeged (2017)

Váradi, T., Simon, E., Sass, B., Mittelholcz, I., Novák, A., Indig, B., Farkas, R., Vincze, V.: E-magyar – A Digital Language Processing System. In: chair), N.C.C., Choukri, K., Cieri, C., Declerck, T., Goggi, S., Hasida, K., Isahara, H., Maegaard, B., Mariani, J., Mazo, H., Moreno, A., Odijk, J., Piperidis, S., Tokunaga, T. (eds.) Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018). European Language Resources Association (ELRA), Miyazaki, Japan (May 7-12, 2018 2018)

Young, T., Hazarika, D., Poria, S., Cambria, E.: Recent Trends in Deep Learning Based Natural Language Processing. IEEE Computational Intelligence Magazine 13(3), 55–75 (2018)

Zeman, D., Hajič, J., Popel, M., Potthast, M., Straka, M., Ginter, F., Nivre, J., Petrov, S.: 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. pp. 1–21. Association for Computational Linguistics, Brussels, Belgium (Oct 2018), https://aclanthology.org/K18-2001

Zeman, D., Popel, M., Straka, M., Hajič, J., Nivre, J., Ginter, F., Luotolahti, J., Pyysalo, S., Petrov, S., Potthast, M., Tyers, F., Badmaeva, E., Gokirmak, M., Nedoluzhko, A., Cinková, S., Hajič jr., J., Hlaváčová, J., Kettnerová, V., Urešová, Z., Kanerva, J., Ojala, S., Missilä, A., Manning, C.D., Schuster, S., Reddy, S., Taji, D., Habash, N., Leung, H., de Marneffe, M.C., Sanguinetti, M., Simi, M., Kanayama, H., de Paiva, V., Droganova, K., Martínez Alonso, H., Çöltekin, Ç., Sulubacak, U., Uszkoreit, H., Macketanz, V., Burchardt, A., Harris, K., Marheinecke, K., Rehm, G., Kayadelen, T., Attia, M., Elkahky, A., Yu, Z., Pitler, E., Lertpradit, S., Mandl, M., Kirchner, J., Alcalde, H.F., Strnadová, J., Banerjee, E., Manurung, R., Stella, A., Shimada, A., Kwak, S., Mendonça, G., Lando, T., Nitisaroj, R., Li, J.: 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. pp. 1–19. Association for Computational Linguistics, Vancouver, Canada (Aug 2017), https://aclanthology.org/K17-3001

Zsibrita, J., Vincze, V., Farkas, R.: magyarlanc: A Toolkit for Morphological and Dependency Parsing of Hungarian. In: Proceedings of Recent Advances in Natural Language Provessing 2013. pp. 763–771. Association for Computational Linguistics, Hissar, Bulgaria (2013)