# Introducing huBERT

Nemeskey Dávid Márk[1]

[1]Számítástechnikai és Automatizálási Kutatóintézet
nemeskey.david@gmail.com

**Abstract.** This paper introduces the huBERT family of models. The flagship is the eponymous BERT Base model trained on the new Hungarian Webcorpus 2.0, a 9-billion-token corpus of Web text collected from the Common Crawl. This model outperforms the multilingual BERT in masked language modeling by a huge margin, and achieves state-of-the-art performance in named entity recognition and NP chunking. The models are freely downloadable.
**Keywords:** huBERT, BERT, evaluation, NER, chunking, masked language modeling

## 1 Introduction

Contextualized embeddings, since their introduction in McCann et al. (2017) have altered the natural language processing (NLP) landscape completely. Systems based on ELMo (Peters et al., 2018), and especially BERT (Devlin et al., 2019) have improved the state of the art for a wide range of benchmark tasks. The improvement is especially notable for high-level natural language understanding (NLU) tasks, such as the ones that make up the GLUE (Wang et al., 2018) and SQuAD (Rajpurkar et al., 2016, 2018) datasets. In the long run, BERT proved more successful than ELMo, not least because once it has been *pretrained* on large amount of texts, it can be *finetuned* on any downstream task, while ELMo cannot stand on its own and must be integrated into traditional NLP systems.

The triumph of BERT also marks the move away from LSTMs (Hochreiter and Schmidhuber, 1997) toward the attention-based Transformer (Vaswani et al., 2017) architecture as the backbone of language representation models. BERT was soon followed by an abundance of similar models, such as RoBERTa (Liu et al., 2019), XLNet (Yang et al., 2019) or BART (Lewis et al., 2019). These models tweak different aspects of BERT, including the amount of training data, the tasks used to pretrain it, or the architecture itself. Each paper reports improvements over the last.

As always in NLP[1], all the pioneering research above was centered on English. Support for other languages came in two forms: native contextual embeddings, such as CamemBERT (Martin et al., 2019) for French, or multilingual variations of the models above. Examples for the latter are multi-BERT and XLM-RoBERTa (Conneau et al., 2019), both of which were trained on corpora with around 100 languages (Wikipedia for the former, the Common Crawl[2] for latter).

[1] With the possible exception of morphology.
[2] https://commoncrawl.org/

Both alternatives have their own advantages and disadvantages: native models are as expensive to train as the original English ones, costing up to hundreds of thousands of euros; which seems excessive, especially since good quality multilingual models have already been published. On the other hand, the capacity of multilingual models is shared among the many languages they support, which hurts single-language performance. Medium-size languages, of which Hungarian is one, are further disadvantaged by the size of the available textual data. In the training corpora of multilingual models, larger languages are represented by a proportionally higher amount of text, which introduces serious bias into the final models. Taking this all into consideration, we came to the conclusion that Hungarian is probably better served by native models.

In this paper, we introduce the `huBERT` family of models. As of now, the family consists of two preliminary BERT Base models trained on Wikipedia and the eponymous `huBERT` model, trained on a new nine-billion-token corpus; it is also the first publicly available Hungarian BERT model. We evaluate `huBERT` against multi-BERT on the two tasks they were pretrained, as well as on two downstream tasks: named entity recognition (NER) and NP chunking. We find that `huBERT` outperforms multi-BERT on the training tasks by a huge margin, and achieves a new state of the art in both NER and NP chunking, thereby strengthening our concluding sentence in the last paragraph.

The rest of the paper is organized as follows. In Section 2, we describe the training corpora and the pretraining process behind the models. Section 3 details our experimental setup and presents our results. Section 4 highlights a few shortcomings of relying solely on the new contextualized embedding machinery. Finally, we conclude our work in Section 5.

## 2 Pretraining

In this section we describe the pretraining procedure in detail in the hope that it helps others embarking on a similar venture avoiding potential pitfalls along the way.

### 2.1 Background

Pretraining modern contextualized representations is a costly business. The models themselves are huge (BERT-Base, which has become a standard, has 110 million parameters), and the associated training corpora also start at several billion words. The quadratic resource requirements of the attention mechanism can only be accommodated by high-end hardware. These factors all add up, and as a result, training a modern Transformer model takes days or weeks on hundreds of GPUs or TPUs.

The financial costs incurred by such a training regimen are prohibitive for smaller laboratories, unless they receive support from the industry. However, most of the time the support is limited and it does not allow experimentation with model architectures, let alone hyperparameter tuning. This means that pretraining for smaller groups is a leap of faith, which either succeeds or not; and this inequality of the playing field raises various ethical issues (Parra Escartín et al., 2017).

Our situation is not different. We were kindly given the use of 5 v3-8 TPUs by Google in the Tensorflow Research Cloud (TFRC)[3] program, as well as two weeks on a v3-256 TPU Pod. Our main goal was to train a BERT-Base model on Webcorpus 2.0: a new, 9-billion-token corpus compiled from the Hungarian subset of the Common Crawl (Nemeskey, 2020b). Based on the numbers in the original BERT paper, we calculated that two weeks should be enough to train the model to convergence. However, an earlier failed attempt at pretraining an ALBERT (Lan et al., 2019) model that never converged convinced us to start with a smaller corpus to ensure that the training process works.

### 2.2 huBERT Wiki

At about 170 million words in 400 thousand documents[4], the Hungarian Wikipedia is but a fraction of the English one. After filtering it according to the BERT guidelines, its size further decreases to about 110 million words in 260 thousand documents. This is considerably smaller than Webcorpus 2.0, but it contains good quality, edited text, which makes it a valuable training resource. Its small size also allowed us to pretrain a BERT-Base model on it in 2.5 days on a single v3-8 TPU.

BERT models usually come in two flavors: *cased* and *uncased*. The former operates on unprocessed text; in the latter, tokens are lower cased and diacritical marks are removed. In keeping with this practice, we also trained two variants. However, as diacritics are *distinctive* in Hungarian, we could not afford to lose them, and replaced the uncased model with a *lower cased* one.

BERT models are pretrained with two tasks: *masked language modeling* (*MLM*) and *next sentence prediction* (*NSP*). The language understanding capabilities of the model reportedly derive from the former (Lan et al., 2019; Liu et al., 2019), as NSP is very easy to learn. Since we used the original BERT training code, we kept both tasks.

As is the case with the English BERT, our models are all pretrained on sequences of up to 512 wordpieces. As the training configurations published in the literature are for much larger corpora, they are not directly adaptable to our case. Hence, we experimented with different training regimens for both the cased and lower cased variants:

1. Three models were trained with full-length sequences for 50,000, 100,000 and 200,000 steps. These roughly correspond to 90, 180 and 360 epochs, respectively;
2. Following the recommendation in the BERT GitHub repository, one model was trained with a sequence length of 128 for 500,000 steps (600 epochs) and with a sequence length of 512 for an additional 100,000 steps (or 180 epochs).

All models were trained with a maximum learning rate of $10^{-4}$ and the maximum possible batch size: 1024 for the model with 128-long sequences and 384 for the rest. The training data for the masked language modeling task was duplicated 40 times with different mask positions. The official training code uses a learning rate decay with a warmup period, which we set to 10% of the total number of training steps. The code unfortunately does not support early stopping; it does not even accept a validation set.

[3] `https://www.tensorflow.org/tfrc`

[4] 2018 snapshot

However, as we shall see in Section 3.1, performance on the test set showed no sign of overfitting.

All models use a wordpiece vocabulary of around 30,000 tokens to match the English BERT-Base models. Increasing it 5,000 tokens did not yield any improvements, so we opted for the smaller vocabulary in order to keep the model smaller.

| Model | Seq. length | Steps | Hours | Masked LM | Next sentence |
|---|---|---|---|---|---|
| Cased | 512 | 50,000 | 13 | 0.5544961 | 0.97125 |
| | 128 | 500,000 | 59 | 0.6669028 | 0.995 |
| | 512 | +100,000 | 25 | 0.6657926 | 0.99 |
| Lower | 512 | 50,000 | 13 | 0.5538445 | 0.9825 |
| | 512 | 100,000 | 25 | 0.6100383 | 0.9975 |
| | 512 | 200,000 | 50 | 0.6273391 | 0.9975 |
| | 128 | 500,000 | 59 | 0.6425686 | 0.99125 |
| | 512 | +100,000 | 25 | 0.665879 | 0.9975 |

**Table 1.** Training times and accuracies of the different BERT models on the two training tasks

Table 1 compares all configurations. In the cased case, the TPU went down for maintenance during training, so the 100,000 and 200,000-step models are missing from the results. Even without them, several observations can be made. First, the 50,000-step models clearly underfit the data, even though they were trained for twice as many epochs as the English BERT. On the other hand, the difference between the 100,000 and 200,000-step models is much smaller than between the 50,000 and 100,000-step models, suggesting a performance peak around 300,000–400,000 steps.

Second, in line with the findings of Lan et al. (2019); Liu et al. (2019), the next sentence prediction task seems very easy, as all but the first models attain over 99% accuracy. In contrast, the masked LM task proved much harder, and its accuracy seems rather low. Unfortunately, the evaluation results for the English BERT are not published anywhere, which makes it difficult to put the numbers in context. Based on the diminishing returns, the longest-trained models are likely to be close to the maximum achievable on Wikipedia alone.

Finally, our experiences confirmed that the two-stage training regimen recommended in the BERT repository indeed leads to better results. The rationale behind this method is that the first phase trains most of the model weights and the second phase is "*mostly needed to learn positional embeddings, which can be learned fairly quickly*"[5]. While this seems to be the case for the cased model, the masked LM accuracy of the lower cased model improved by more than 2% in the second phase, indicating either that sub-

[5] `https://github.com/google-research/BERT/#pre-training-tips-and-caveats`

stantial learning still happens at this stage or that some of the dependencies in the data can be better exploited by a 512-token window.

### 2.3 huBERT

Having confirmed that the BERT training code works and produces functional models on Wikipedia, we proceeded to train the main huBERT model on the much larger Webcorpus 2.0. We used the same configuration as for the preliminary models, with two notable exceptions.

First, we only had time to pretrain one model. We chose to focus on the cased model, as that is more universally usable. Second, as opposed to single TPUs, TPU Pods are always preemptible, and our earlier experience with ALBERT taught us that the training might be interrupted several times a day. Unfortunately, the original BERT training script is not prepared for this eventuality and once interrupted, it can never resume training. To mitigate this issue, we wrote a wrapper script around the BERT training code that monitors the log file and restarts training whenever the TPU Pod goes down. We also decreased the number of steps between checkpoints to 1,000 (from the default 5,000) to minimize the work lost.

In the end, our training quota expired after 189,000 steps, cutting the pretraining slightly short. To validate the model, we ran the same evaluations as we did for the preliminary models, this time on a held-out portion of Webcorpus 2.0. The results (MLM accuracy of 0.63 with a sequence length of 128 and 0.66 with 512) closely follows those reported in Table 1, which indicates that the model might similarly be close to convergence and better results could only be excepted of larger (e.g. BERT-Large) models.

### 2.4 Availability

All huBERT models can be downloaded freely from the huBERT homepage[6]. The main huBERT model is also available from the Hugging Face model repository[7] under the moniker SZTAKI-HLT/hubert-base-cc.

The emBERT NER and NP taggers, described in Section 3.2, replace the original models based on multi-BERT and can be downloaded from inside emtsv or from the GitHub repository[8].

## 3 Evaluation

BERT models are usually evaluated on high-level natural language understanding tasks, such as question answering or textual entailment. Unfortunately, no Hungarian benchmark datasets exist for these tasks. Because of this, we evaluate our models by contrasting their performance to the multi-language version of BERT in two ways:

[6] https://hlt.bme.hu/en/resources/hubert
[7] https://huggingface.co/SZTAKI-HLT/hubert-base-cc
[8] https://github.com/dlt-rilmta/emBERT-models

1. We compare their accuracy on the two training tasks on a held-out portion of Wikipedia and Webcorpus 2.0.
2. We include our models in the `emBERT` module (Nemeskey, 2020a) and measure their performance on named entity recognition and NP chunking.

### 3.1 Training tasks

Table 2 presents the results of the first experiment. Both our cased and lower cased models achieve similar accuracies on the held-out set as on the training data, allaying any suspicion of overfitting. The `huBERT` Wiki models perform slightly better on Wikipedia than `huBERT`, but attain significantly lower accuracy on Webcorpus 2.0. Compared to this, `huBERT` is fairly robust across both corpora, no doubt benefiting from its much larger and more varied training corpus. All cased models clearly outperform multi-BERT on both tasks (multi-BERT is only available in the cased configuration).

| Case | Model | Wikipedia | | Webcorpus 2.0 | |
|---|---|---|---|---|---|
| | | MLM | NSP | MLM | NSP |
| Cased | multi-BERT | *0.00001* | *0.560* | *0.000004* | *0.455* |
| | `huBERT` Wiki | **0.65** | **0.988** | 0.46 | 0.786 |
| | `huBERT` | 0.64 | 0.985 | **0.61** | **0.959** |
| Lower | `huBERT` Wiki | 0.641 | 0.99 | | |

**Table 2.** Accuracy of multi-language BERT and members of the `huBERT` family on the two training tasks on the held-out set of the two training corpora.

In fact, the performance of multi-BERT leaves a lot to be desired. Its accuracy on the next sentence prediction task is, at 50%, effectively random. The masked LM loss is equivalent to a perplexity of about 130,000, which, given its vocabulary of 120,000 wordpieces, is even worse than that.

On the one hand, this abysmal performance comes as a surprise, for two reasons: first, it was also trained on Wikipedia; and second, multi-BERT fares much better on downstream tasks (see Section 3.2, below). On the other, it goes to show that multi-language models sacrifice too much of single-language performance to be of actual use for the tasks they were trained on. This underlines the importance of native Hungarian contextual embeddings.

### 3.2 NLP tasks

Tables 3 and 4 show the performance of `huBERT`-based models against leading Hungarian systems on NP chunking and NER, respectively[9]. The tables are extended versions

[9] For training details and a more thorough description of the tasks and the corresponding data, the reader is referred to Nemeskey (2020a)

of those found in Nemeskey (2020a). One difference to note is that, for the sake of a fair comparison, we only included systems in Table 4 that were trained and tested on the standard split of the Szeged NER corpus.

Table 3 demonstrates that BERT-based models in general perform favorably compared to traditional statistical models, represented here by members of the `hunchunk` family. multi-BERT already outperforms HunTag3 in maximal NP-chunking by 1.5% F1 score on the test set, but it could only match `hunchunk`'s results on minimal NPs. `huBERT Wiki`, on the other hand, improves both scores by 1–1.5%. `huBERT` tops the list with another 0.5% increase on both tasks, achieving a new state of the art on both.

| System | Minimal | Maximal |
|---|---|---|
| `hunchunk`/HunTag (Recski, 2010) | 95.48% | 89.11% |
| HunTag3 (Endrédy and Indig, 2015) | – | 93.59% |
| `emBERT` w/ multi-BERT | 95.58% | 95.05% |
| `emBERT` w/ `huBERT Wiki` | 96.64% | 96,41% |
| `emBERT` w/ `huBERT` | **97.14%** | **96,97%** |

**Table 3.** Comparison of Hungarian NP chunkers

The results for named entity recognition (see Table 4) are less straightforward. `emBERT` with multi-BERT achieves 1% higher F1 score than the previous best (Simon, 2013). As opposed to the NP chunking tasks, `huBERT Wiki` could not improve on the multilingual model – in fact, it reaches a slightly lower F1 score, even though the difference is not significant. `huBERT`, however, again manages to squeeze another 0.5% out of the data, setting a new record on the Szeged NER corpus.

| System | F1 |
|---|---|
| (Szarvas et al., 2006) | 94,77% |
| `hunner` (Varga and Simon, 2007) | 95.06% |
| `hunner` (Simon, 2013) | 96.10% |
| `emBERT` w/ multi-BERT | 97,08% |
| `emBERT` w/ `huBERT Wiki` | 97,03% |
| `emBERT` w/ `huBERT` | **97,62%** |

**Table 4.** Comparison of Hungarian NER taggers

# 4 All that glitters is not gold

In this section, we dive briefly behind the numbers and show that even though our BERT models established new state of the art on two downstream benchmarks, their actual behavior on real-world data might lack in some areas. It must be pointed out that the two issues described below occur only to the named entity tagger, which implies a problem with insufficient training data (see Nemeskey (2020a)) rather than with the capabilities of the model architecture itself.

## 4.1 Invalid tag sequences

The numbers for both NP chunking and NER paint a similar picture: all BERT-based taggers outperform traditional machine learning systems on both tasks, with `huBERT` beating multi-BERT by a few percent. In case of NER, the gap is as small as 0.5%, which hardly justifies spending the resources needed to train a native Hungarian BERT model. However, when the taggers are applied to data outside the Szeged NER corpus, a different picture emerges.

In the original `emBERT` system, the labels emitted by the taggers were output as-is. This runs the risk of producing invalid tag sequences, of which an example is shown in Table 5. Here, multi-BERT generates invalid sequences such as `B-ORG B-ORG`, `E-MISC E-MISC` and even `B-MISC I-PER`. The tag sequence emitted by `huBERT` Wiki also contains an error, and its classification is not better than multi-BERT's, either. `huBERT`'s output, on the other hand, is perfectly valid and the tagging is much more accurate as well.

It is worth mentioning that invalid tag sequences are rare, as the attention mechanism BERT is based on is able to use information from all tokens in the sequence, and hence the model finds the boundaries of named entities most of the time. It is only when the input sentence has an odd structure that we encountered invalid tag sequences. Indeed, the sentence in Table 5 is not a sentence in the grammatical sense; instead, it is the list of characters in a play, mistakenly grouped together by the sentence splitter. Still, tokenization errors and fragmented data crop up in all corpora, and our systems have to be robust enough to handle them. The `huBERT`-based tagger can be more robust to unfamiliar input than the other two because it was trained solely on (large and often fragmented) Hungarian data.

Nevertheless, we cannot be sure that `huBERT` taggers always generate valid output and hence we implemented a Viterbi-like algorithm on top of the tagger that prevents invalid tag transitions. The transition probabilities are uniform for each valid transition between tags (i.e. `B-PER` $\rightarrow$ `I-PER`) and 0 otherwise. We decided against learning the probabilities from the training corpus, as it would downweight rarely seen but otherwise valid transitions. This would effectively prevent us from correctly tagging `1-`$\star$ entities, as the `O` $\rightarrow$ `O` transition is much more probable than `O` $\rightarrow$ `1-MISC`, etc.

| Sentence | multi-BERT | huBERT Wiki | huBERT | m-B Viterbi |
|---|---|---|---|---|
| BARABÁS | B-PER | B-ORG | B-PER | B-PER |
| ÁDÁMNÉ | E-PER | E-ORG | E-PER | E-PER |
| az | O | O | O | O |
| édesanyja | O | O | O | O |
| A | B-ORG | O | O | O |
| MESTER | B-ORG | B-ORG | B-ORG | B-ORG |
| SZTELLA | E-ORG | E-ORG | E-ORG | E-ORG |
| a | O | O | O | O |
| partnernője | O | O | O | O |
| MISI | B-MISC | 1-MISC | 1-PER | B-MISC |
| bohóc | E-MISC | O | O | E-MISC |
| NOVOTNI | B-MISC | B-PER | B-PER | B-MISC |
| NÁNÁSI | I-MISC | I-MISC | I-PER | I-MISC |
| PIRI | E-MISC | E-PER | E-PER | E-MISC |
| lektor | E-MISC | O | O | O |
| MAROSI | 1-MISC | 1-MISC | 1-MISC | 1-MISC |
| újságíró | O | O | O | O |
| LITTKÉNÉ | B-MISC | B-MISC | 1-PER | B-PER |
| NÉGY | I-PER | I-MISC | O | I-PER |
| KATONA | I-PER | I-MISC | O | I-PER |
| PERECESLÁNY | E-PER | E-MISC | O | E-PER |

**Table 5.** Invalid tag sequences on a text fragment from the screenplay of Tragédia (1979) by István Örkény

As seen in the last column of Table 5, applying the Viterbi algorithm to the class transitions prevents the emission of invalid tag sequences, and occasionally improves the results as well.

### 4.2 Overenthusiasm

Applying the NER taggers to single words demonstrates another peculiarity of our BERT-based taggers: they seem overly enthusiastic to give a non-O label to almost any single word, including "*a*" (the) , "*macska*" (cat) or "*fut*" (run). This does not happen when the words are in a sentential context, e.g. "*a macska fut*" (the cat is running) is correctly tagged as O O O. The cause of this behavior is not yet clear, as the training corpus contains no one-word "sentences", and thus requires further research. As mentioned above, the chunker models are unaffected by this issue, which makes the NER training corpus the primary suspect.

## 5 Conclusion and future work

In this paper, we have introduced the huBERT family of models. The first three members of the family are two preliminary BERT-Base models pretrained on Wikipedia and

the eponymous huBERT model pretrained in Webcorpus 2.0. According to our tests, all models, but especially the latter, outperform the multilingual BERT model both in the tasks used to pretrain them and in token classification tasks, such as NP chunking and NER. huBERT achieves a new state of the art in both NLP tasks. Additionally, the models trained on solely Hungarian corpora seemed more stable when applied to unfamiliar text. huBERT is available on the Hugging Face Model Hub in both Pytorch and TensorFlow flavors.

In the future, we expect further, more recent models, such as Electra (Clark et al., 2020), to be added to the family.

## Acknowledgements

This work was partially supported by National Research, Development and Innovation Office (NKFIH) grants #115288: "*Algebra and algorithms*" and #120145: "*Deep Learning of Morphological Structure*", as well as by National Excellence Programme 2018-1.2.1-NKP-00008: "*Exploring the Mathematical Foundations of Artificial Intelligence*".

huBERT was trained on TPUs provided by the Tensorflow Research Cloud program. Their support is gratefully acknowledged.

The authors thank Eszter Simon for bringing the issue of the emBERT NER model's outputting invalid tag sequences to their attention and for the anonymous reviewers for their valuable insights.

## Bibliography

Clark, K., Luong, M.T., Le, Q.V., Manning, C.D.: Electra: Pre-training text encoders as discriminators rather than generators. In: 8th International Conference on Learning Representations, ICLR 2020 (2020), https://openreview.net/forum?id=r1xMH1BtvB

Conneau, A., Khandelwal, K., Goyal, N., Chaudhary, V., Wenzek, G., Guzmán, F., Grave, E., Ott, M., Zettlemoyer, L., Stoyanov, V.: Unsupervised cross-lingual representation learning at scale (2019)

Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: BERT: Pre-training of deep bidirectional transformers for language understanding. In: Proc. of NAACL (2019)

Endrédy, I., Indig, B.: HunTag3, a General-purpose, Modular Sequential Tagger – Chunking Phrases in English and Maximal NPs and NER for Hungarian, p. 213–218. Uniwersytet im. Adama Mickiewicza w Poznaniu, Poznan (2015)

Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Computation 9(8), 1735–1780 (11 1997)

Lan, Z., Chen, M., Goodman, S., Gimpel, K., Sharma, P., Soricut, R.: ALBERT: A lite BERT for self-supervised learning of language representations. CoRR abs/1909.11942 (2019), http://arxiv.org/abs/1909.11942

Lewis, M., Liu, Y., Goyal, N., Ghazvininejad, M., Mohamed, A., Levy, O., Stoyanov, V., Zettlemoyer, L.: BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension (2019)

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., Stoyanov, V.: RoBERTa: A robustly optimized bert pretraining approach (2019)

Martin, L., Muller, B., Suárez, P.J.O., Dupont, Y., Romary, L., de la Clergerie, É.V., Seddah, D., Sagot, B.: Camembert: a tasty french language model. arXiv preprint arXiv:1911.03894 (2019)

McCann, B., Bradbury, J., Xiong, C., Socher, R.: Learned in translation: Contextualized word vectors. In: Advances in Neural Information Processing Systems. pp. 6294–6305 (2017)

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

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

Parra Escartín, C., Reijers, W., Lynn, T., Moorkens, J., Way, A., Liu, C.H.: Ethical considerations in NLP shared tasks. In: Proceedings of the First ACL Workshop on Ethics in Natural Language Processing. pp. 66–73. Association for Computational Linguistics, Valencia, Spain (04 2017), `https://www.aclweb.org/anthology/W17-1608`

Peters, M.E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., Zettlemoyer, L.: Deep contextualized word representations. arXiv preprint arXiv:1802.05365 (2018)

Rajpurkar, P., Jia, R., Liang, P.: Know what you don't know: Unanswerable questions for SQuAD. In: Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers). pp. 784–789. Association for Computational Linguistics, Melbourne, Australia (07 2018), `https://www.aclweb.org/anthology/P18-2124`

Rajpurkar, P., Zhang, J., Lopyrev, K., Liang, P.: SQuAD: 100,000+ questions for machine comprehension of text. In: Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing. pp. 2383–2392. Association for Computational Linguistics, Austin, Texas (11 2016), `https://www.aclweb.org/anthology/D16-1264`

Recski, G.: Főnévi csoportok azonosítása szabályalapú és hibrid módszerekkel. In: Tanács, A., Vincze, V. (eds.) VII. Magyar Számítógépes Nyelvészeti Konferencia. pp. 333–341 (2010)

Simon, E.: Approaches to Hungarian Named Entity Recognition (2013), ph.D. Thesis, Budapest University of Technology and Economics

Szarvas, G., Farkas, R., Kocsor, A.: A multilingual named entity recognition system using boosting and C4.5 decision tree learning algorithms. In: Discovery Science, 9th International Conference, DS 2006, Barcelona, Spain, October 8-10, 2006, Proceedings. pp. 268–278 (2006)

Varga, D., Simon, E.: Hungarian named entity recognition with a maximum entropy approach. Acta Cybern. 18(2), 293–301 (Feb 2007)

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L.u., Polosukhin, I.: Attention is all you need. In: Guyon, I., Luxburg, U.V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., Garnett, R. (eds.) Advances in Neural Information Processing Systems 30, pp. 5998–6008. Curran As-

sociates, Inc. (2017), http://papers.nips.cc/paper/7181-attention-is-all-you-need.pdf

Wang, A., Singh, A., Michael, J., Hill, F., Levy, O., Bowman, S.R.: Glue: A multi-task benchmark and analysis platform for natural language understanding (2018)

Yang, Z., Dai, Z., Yang, Y., Carbonell, J., Salakhutdinov, R., Le, Q.V.: Xlnet: Generalized autoregressive pretraining for language understanding. In: Advances in neural information processing systems. pp. 5754–5764 (2019), https://papers.nips.cc/paper/8812-xlnet-generalized-autoregressive-pretraining-for-language-understanding.pdf