The Impact of Tokenization Algorithms on Hungarian Language Model Performance

Mátyás Osváth, Máté Molnár, Roland Gunics, Noémi Ligeti-Nagy
ELTE Research Centre for Linguistics
1068 Budapest, Benczúr u. 33.
{surname.firstname}@nytud.elte.hu

Abstract

Tokenization is a crucial text processing step for preparing input for language models and can contribute to model performance, especially in morphologically rich languages. Currently, Byte Pair Encoding (BPE), WordPiece, and Unigram LM algorithms are predominantly used in language models, but their effects can vary in agglutinative languages. This work compares these tokenization algorithms across varying vocabulary sizes, as well as a modified Unigram LM variant with morphologically informed initialization, on the Hungarian subset of the OSCAR dataset. The evaluation is based on several metrics describing the inferred quality of the tokenizers and on the downstream performance of multiple BERT models on the HuLU benchmark. Results show that BPE produces the most compact and morphologically aligned subword representations, while the modified Unigram LM achieved the best overall downstream performance across tasks. However, differences between methods and vocabulary sizes were generally small and not statistically significant, with the exception of HuCoPA (a task within the HuLU benchmark), which showed sensitivity to both factors. These findings underscore that tokenizer choice and vocabulary design are critical determinants of language model efficiency and performance in morphologically rich languages.

Keywords: subword tokenization, BERT pretraining, Hungarian NLP

1. Introduction

Transformer models (Vaswani et al., 2017) have revolutionized natural language processing (NLP), achieving state-of-the-art performance in a variety of tasks such as machine translation, text generation and summarization. These models crucially depend on the appropriate preprocessing of the input text, with tokenization constituting a fundamental step (Chelombitko et al., 2024). Earlier word embedding approaches, like word2vec (Mikolov et al., 2013), often relied on surface forms of words; in contrast, current tokenization methods resolve the open-vocabulary (OOV) problem by decomposing text into smaller units, called tokens, which may correspond to characters, words, or subwords. In this way, tokenization provides a systematic mechanism for handling rare words by segmenting them into subwords, or into individual characters.

Currently, several tokenization algorithms exist, each employing distinct approaches for segmentation and for constructing the vocabulary. Among the most commonly used are Byte Pair Encoding (BPE) (Sennrich et al., 2016), WordPiece (Wu et al., 2016) and Unigram Language Model (Unigram LM) (Kudo, 2018), while various optimizations have been proposed to enhance their performance (Radford et al., 2019; Hofmann et al., 2022). For instance, BERT model (Devlin et al., 2019) and its derivative, DistilBERT (Sanh et al., 2019) employ WordPiece, GPT-2 (Radford et al., 2019) and subsequent GPT models as well as RoBERTa (Liu et al., 2019) rely on BPE (or its variant), while T5 (Raffel et al., 2019) and ALBERT (Lan et al., 2019) make use of the Unigram LM. The aforementioned studies do not examine in detail how the choice of tokenization algorithm impacts model performance, nor why a particular tokenizer was selected in each case.

In the pretraining of language models, the prevailing tendency is toward an English-centric focus (Jiang et al., 2023; Team et al., 2024; Grattafiori et al., 2024), although in recent years natively multilingual models - pretrained simultaneously on multiple target languages - have also appeared (Workshop et al., 2023), but their performance typically lags behind that of English-only models. Models are often adapted to other languages in one of two ways. In the first approach, the vocabulary (and tokenization) is left unchanged (Zijian et al., 2024b,a). In such cases, the tokenizer and vocabulary may be suboptimal for the target language, particularly if it differs substantially from the source language in morphology or lexicon. In the second approach, several studies have experimented with extending and/or modifying the vocabulary (together with the embedding layer) (Csaki et al., 2024; Liu et al., 2024; Moroni et al., 2025). However, relatively few works have examined the impact of different tokenization algorithms on under-represented and morphologically rich languages, which frequently exhibit complex word forms and larger lexicons.

Although efforts have been made to pre-train and fine-tune Hungarian language models (Zijian et al., 2024b,a), the choice of tokenizer and its parameterization remains a challenge, and the impact of different tokenization methods has not yet been

Proceedings of the Fifteenth Language Resources and Evaluation Conference (LREC 2026), pages 2545–2556
11–16 May 2026.

investigated for Hungarian.

Therefore, the aim of the present study is to compare various tokenization methods for Hungarian language. Three tokenization algorithms - BPE, WordPiece and Unigram LM - are evaluated across multiple vocabulary sizes, and against a morphology-based tokenizer. For evaluation, we employ a range of intrinsic and extrinsic metrics, with several BERT models trained in order to compare performance on the Hungarian subset of the OSCAR corpus (Ortiz Suárez et al., 2019).

2. Related work

Research has shown that the choice of tokenization method can significantly impact the performance of mono- and multilingual language models (Chelombitko et al., 2024; Rust et al., 2021; Wang et al., 2025). Although tokenization-free approaches have been proposed, like ByT5 (Xue et al., 2022), bGPT (Wu et al., 2024), CharFormer (Tay et al., 2022) and Canine (Clark et al., 2022), the tokenization-based methods remain dominant in linguistic applications. The tokenizers' regular expression, vocabulary size and training data can affect the model's ability to capture linguistic nuances and generalize across different languages and domains (Dagan et al., 2024; Wegmann et al., 2025).

The Unigram LM - proposed by Kudo (Kudo, 2018) - achieved better results than BPE in both Japanese and English, while also making more efficient use of vocabulary size by creating semantically meaningful tokens. However, several other studies have explored the impact of different tokenization methods, some of them on specific (e.g. morphologically rich, low-resource) languages and subdomains. Labrak et al. found that integration of morphemes in the tokenization process improved the performance of a model in certain downstream tasks in biomedical domain (Labrak et al., 2024). Toraman et al. (Toraman et al., 2023) compared the above-mentioned algorithms in Turkish - along with a morphological tokenizer - and found that BPE and WordPiece provided similar performance in favor of Unigram LM, but increasing vocabulary size improved performance across all methods. Beso (Mikaberidze et al., 2024) also found that increasing vocabulary size improved performance in Georgian, a morphologically rich language, but the differences between tokenization methods were less pronounced. The same conclusion was drawn by Tamang and Bura's research on multilingual and Indian languages (Tamang and Bora, 2024). In contrast, for Arabic language, UnigramLM outperformed BPE and WordPiece tokenizer (Qarah and Alsanoosy, 2024). Conflicting findings can be observed in the literature, also highlighted by Vemula (Vemula et al., 2025), whose results suggest that the algorithm choice has more impact on downstream performance than morphological alignment. Arnett found that morphological alignment does not explain very much difference in model performance. (Arnett et al., 2025).

As there is no widely accepted and applied standard methodology to evaluate tokenizers, comparisons are often done intrinsically and extrinsically. Intrinsic metrics assess the quality of the tokenizer independently of the language model, correlating with language model performance. These measures include fertility, normalized sequence length, compression ratio, Rényi-entropy and corpus token count, among others. Extrinsic metrics evaluate the performance of the model (perplexity, cross-entropy loss) and its effectiveness in downstream tasks, such as text classification, sentiment analysis, named entity recognition (NER) or natural language inference (NLI), to infer the impact of tokenization (Wegmann et al., 2025).

2.1. Algorithms

Tokenization can be described by three stages: pre-tokenization, vocabulary construction and segmentation, with the algorithm involved in the latter two. Pre-tokenization is an optional step that splits the text into smaller units (e.g. splitting a corpus on whitespace, using certain regular expressions) to permit or constrain the generation of specific tokens. Vocabulary construction is the process of creating a vocabulary of tokens $t_i \in V \ \ (i \in \{1, \ldots, k\})$ of size k from a given D corpus, where $|V| = k$. Subsequently, in segmentation, for a vocabulary V and a new text d, a sequence of tokens $t_1, \ldots, t_k \in V$ is created according to this vocabulary, where the decoded tokens' sequence equals to d. In practice, these two steps are interrelated (Bostrom and Durrett, 2020; Schmidt et al., 2024).

Byte Pair Encoding (BPE) is a widely used tokenization (and compression) algorithm (Sennrich et al., 2016). It is an iterative procedure which begins with a specified set of characters forming an initial vocabulary. At each step, the most frequent pair of adjacent units (bigrams) in the corpus is merged into a new token, which is then added to the vocabulary. This process continues until the vocabulary size k is reached. It can represent previously unseen words by decomposing them into multiple tokens, thereby reducing the number of unknown (UNK) tokens and allowing for smaller vocabularies. A variant, byte-level BPE, operates not on a Unicode character set but directly on raw bytes (Radford et al., 2019). Since any text in any encoding can be reduced to bytes, this method guarantees complete coverage without unknown tokens or encoding errors. For a vocabulary V consisting of ordered merges, the tokenization applies

the merges for a new input d in the same order as they were created, until no further merges are possible.

The WordPiece algorithm (Wu et al., 2016) is a tokenization method closely related to BPE. It begins with an initial vocabulary, but instead of merging the most frequent bigrams, it assigns a score to each possible merge, based on the likelihood of an n-gram language model fitted on the version of the corpus produced by that merge. The algorithm prioritizes merges involving less frequent elements in the corpus.[1] Since computing scores for all possible merges is computationally expensive for large-scale language models, the implementations employ heuristics to reduce the number of considered merges (Bostrom and Durrett, 2020). As no official implementation of WordPiece has been released, in this study we rely on the Hugging Face reconstruction of the WordPiece algorithm[2].

In contrast to the previously described algorithms, Unigram LM method adopts a top-down approach to vocabulary construction. It begins with a superset of the final vocabulary and iteratively removing elements that causes the minimum likelihood decrease for the corpus until the desired vocabulary size is reached ($|V| = k$). During segmentation, given a fixed vocabulary V and language model parameters θ, the algorithm determines the optimal segmentation of the input using Viterbi algorithm, selecting the segmentation that maximizes the likelihood under θ (Kudo, 2018; Bostrom and Durrett, 2020).

The standard Unigram LM initializes the seed vocabulary using suffix arrays derived from the input text. In this study, we additionally modify this initialization phase. Instead of relying solely on surface-level frequency statistics, the initial vocabulary is derived from the morphologically analyzed dataset. Consequently, each candidate token in the seed vocabulary corresponds to a linguistically valid morpheme, providing a morphologically well-founded starting point for the subsequent vocabulary pruning process.[3]

3. Methods

The dataset for training the tokenizers and language models was taken from the Hungarian subset of the OSCAR corpus (Ortiz Suárez et al., 2019). OSCAR is a multilingual collection of texts extracted from Common Crawl corpus. The Hungarian dataset contains approximately 18,013,388 documents and 5,163,936,345 words. The preprocessing of the corpus is described in Appendix Appendix A, after which 6,296,186 documents and 1,804,576,307 words remained. The initial vocabulary for the morphology-seeded variant of the Unigram LM tokenizer was constructed from a linguistically correct set of morphemes. The seed vocabulary is derived using *emMorph* (Novák et al., 2016), a rule-based morphological analyzer for Hungarian. We analyze the corpus with *emMorph* and extract the set of unique morphemes, ensuring that every candidate token at initialization corresponds to a linguistically well-formed morphological unit. The tokenizers were trained using the Hugging Face tokenizers library[4]. The pretokenization consisted of splitting text on whitespace, NFC Unicode normalization was used and all parameters were kept constant across the variants of tokenizers with different vocabulary sizes (16K, 32K, 64K and 128K) in order to separate the tokenizer impact on language modeling.

The tokenizers were evaluated using several intrinsic metrics. First, fertility, which measures the average number of tokens produced per word in the input text. The second, corpus token count, which counts the total number of tokens generated for the corpus. Third, Normalized Sequence Length (NSL) (Dagan et al., 2024) that compares the compression of a given tokenizer with respect to a baseline tokenizer. As the baseline, we used the tokenizer of our most recent model, which is based on the tokenizer of the Llama 3.1 8B Instruct model (Yang et al., 2025a). Formally, for a corpus D and tokenizers T_λ, T_β, the NSL is defined as:

$$c_{\frac{\lambda}{\beta}} = \frac{\sum_{i=1}^{N} \text{length}\left(T_\lambda\left(D_i\right)\right)}{\sum_{i=1}^{N} \text{length}\left(T_\beta\left(D_i\right)\right)}$$

Several other metrics were also computed, including the average number of Bytes per Token, calculated by dividing the number of UTF-8 bytes by the number of tokens produced by the tokenizer for a given text. Rényi-efficiency provides a theoretical measure of how effectively a tokenizer balances information compression and distribution across tokens. The metric penalizes both highly frequent and extremely rare subword units, with higher scores generally associated with more effective tokenization and improved downstream performance (Zouhar et al., 2023). However, due to the absence of a computationally efficient implementation and recent critiques demonstrating counterexamples to its validity (Cognetta et al., 2024), we exclude Rényi-efficiency from our present analysis. Furthermore, a quantitative analysis was conducted to compare

[1] See https://huggingface.co/learn/llm-course/en/chapter6/6?fw=pt

[2] The publicly available SentencePiece library does not implement WordPiece, though Hugging Face has attempted to reconstruct it in their tokenizers library, based on publications.

[3] The code for the modified tokenizer can be found at https://github.com/matee8/tokenizers.

[4] https://github.com/huggingface/tokenizers

how different tokenizers align with Hungarian morpheme boundaries, using the Hungarian Universal Dependencies (UD) Treebank (Vincze et al., 2017) and the MorphScore framework (Arnett et al., 2025) to evaluate how well the tokenizers capture the morphological structure of the language.

For the language model training, we chose encoder-only transformer model with BERT architecture across our experiments while varying the tokenizer. All model parameters were initialized from a normal distribution with mean 0 and standard deviation equal to the configured initializer range, following the original BERT initialization scheme. Overall, 16 models (with 99M, 111M, 136M, and 187M parameters for each vocabulary size) were pretrained on the full dataset using the masked language modeling (MLM) objective with an MLM probability of 0.15. All models were pretrained and fine-tuned on 8 NVIDIA A100 GPUs with 80GB VRAM. Each model was trained for 2 epochs with 16-bit precision. Hyperparameters were kept constant across all experiments, and are shown in Appendix Appendix B.

We used the Hungarian Language Understanding (HuLU) benchmark kit to evaluate the models. It encompasses a variety of tasks - each representing different linguistic phenomena and task complexity - to evaluate the performance of neural language models on Hungarian (Ligeti-Nagy et al., 2024). The description related to each task and the hyperparameters used while finetuning for them can be found in Appendix Appendix C and Appendix Appendix D. For each task, we conducted three runs independently and report the mean and standard deviation of the results to ensure reliability. Furthermore, as some of the datasets were inbalanced, the performance of the models were measured using balanced accuracy instead of accuracy.

4. Results

4.1. Segmentation Comparison

We qualitatively analyzed the differences in the tokenization output between BPE, WordPiece, Unigram LM, and the modified Unigram LM. Our observations indicate that the standard Unigram LM does not yield substantially more morphologically aligned segmentations than BPE or WordPiece at any evaluated vocabulary size. In contrast, the modified Unigram LM variant demonstrates a slightly improved alignment with linguistically plausible subword boundaries across the vocabulary sizes. The differences between the tokenization methods (for 32K vocabulary size) is demonstrated in Table 1. In our experiments, the larger the vocabulary size, the more similar the tokenizations become across the different algorithms.

The most frequent tokens in tokenizers were also extracted and can be seen in Table 2. As BPE and WordPiece creates an initial vocabulary from the corpus, we show the first tokens after the initial vocabulary. The Unigram LM assigns a score to each token in the vocabulary, so we can display the top tokens with the highest scores. We observe the recognizable Hungarian affixes in Unigram LM method, however as we progress in the vocabulary, the affixes will appear in the other two methods as well.

Furthermore, we compared the tokenizers with different vocabulary sizes in terms of their alignment with Hungarian morpheme boundaries. Contrary to our expectations, our result suggests that for Hungarian language, the segmentations produced by BPE are more aligned with morphological boundaries compared to WordPiece and both variants of Unigram LM, as the vocabulary size increases, which is supported by higher precision and F1-scores achieved by BPE across almost all vocabulary sizes. The results are summarized in Table 3. This is in contrast with other studies, suggesting that Unigram LM method aligns more closely with morphological references (Bostrom and Durrett, 2020; Creutz et al., 2005). However, they experimented on English and Japanese with only 20K vocab size, while our experiments were conducted on Hungarian with larger vocabulary sizes, which may explain the differing results. It is possible that the greedy nature of BPE allows it to capture more frequent morphemes in Hungarian, while the probabilistic approach of Unigram LM may lead to less consistent segmentation. The modified Unigram LM achieved high recall but comparatively low precision, indicating a tendency toward oversegmentation: while it successfully identified the majority of true morpheme boundaries, it also introduced a substantial number of spurious boundary predictions. Although the model was initialized with a linguistically validated morpheme inventory, the probabilistic objective of the Unigram LM does not explicitly enforce morphologically minimal segmentations. During likelihood maximization, the model may therefore prefer segmentations containing multiple shorter, valid morphemes over larger units, resulting in additional boundary insertions. Consequently, the morphology-informed initialization improves boundary coverage (recall), but does not prevent the introduction of extra, contextually unnecessary morpheme boundaries, leading to reduced precision. Further analysis is needed to understand the underlying reasons for these differences.

4.2. Intrinsic Metrics Evaluation

The intrinsic metrics reveal clear and systematic trends across tokenization strategies and vocabu-

Original	formációkból	prímfaktorizáció	nanotechnológia
BPE	form ációk ból	pr ím faktor izáció	nan ote chn ológia
WordPiece	form ációk ból	pr ím fa k tori zá ció	na no techn ológia
Unigram LM	formáció k ból	prím faktor izáció	n a n o technológia
Modified Unigram LM	formáció k ból	prím faktor izá c ió	nanotechnológia

Table 1: Subword segmentations of three Hungarian terms under the four tokenizers.

More frequent in		
BPE	Unigram LM	WordPiece
sz et en gy	a , . i t	á r s k ó
er at al és tt	z és is A hogy	f t a l d

Table 2: Tokens more frequent in the three tokenizers (## omitted for space) for 32K vocab size.

Tokenizer	Vocab Size	Recall	Precision	F1-score
BPE	16K	74.9%	53.4%	62.2%
	32K	78.9%	63.9%	**70.6%**
	64K	83.3%	73.2%	**77.9%**
	128K	87.5%	81.2%	**84.2%**
Unigram LM	16K	83.4%	52.6%	**64.5%**
	32K	81.2%	57.7%	67.4%
	64K	80.0%	61.0%	69.2%
	128K	79.1%	62.2%	69.6%
WordPiece	16K	69.1%	49.2%	57.4%
	32K	76.8%	61.8%	68.4%
	64K	82.6%	72.0%	76.9%
	128K	87.4%	80.4%	83.7%
Modified Unigram LM	16K	96.2%	44.1%	60.46%
	32K	96.4%	45.1%	61.45%
	64K	96.5%	45.6%	61.93%
	128K	96.3%	54.7%	61.98%

Table 3: Comparison of subword boundary alignment between tokenization methods and reference morphological segmentations.

lary sizes. As expected, increasing the vocabulary size consistently reduces fertility — the average number of tokens produced per word — across all methods, indicating more compact segmentations.

Among the three tokenizers, BPE exhibits the lowest fertility and corpus token count (CTC) at every vocabulary size, implying that its greedy merge mechanism yields the most efficient segmentation for Hungarian. This reduction in CTC comes with a corresponding increase in the average bytes per token (BPT), reflecting a shift toward larger, more information-dense subword units. In contrast, the Unigram LM tokenizer produces higher fertility and larger CTC values, which would suggest a tendency toward finer-grained segmentation - as reported by previous studies, though contradicted by the morphological alignment results presented earlier. This behaviour is further reflected in its slightly lower compression rate (i.e., lower Bytes per Token, BPT), indicating that more tokens are required to represent the same amount of text. While such finer segmentation may be advantageous for morphologically complex languages, it comes at the cost of longer input sequences and increased computational overhead. WordPiece sits between these two extremes, displaying stable behaviour but slightly less compression efficiency than BPE as the vocabulary grows.

In addition to these three tokenizers, we also evaluated the morphology-seeded Unigram LM, which exhibits a distinct segmentation pattern. Because its initial vocabulary is constructed from morphemes extracted using a morphological analyzer, the model produces substantially finer-grained segmentations than the other methods. This tendency is reflected in both fertility and total corpus token count, which remain consistently high across all vocabulary sizes. In contrast to the standard Unigram LM, increasing the vocabulary size has only a limited impact on segmentation granularity, sug-

gesting that the morphology-informed initialization constrains the tokenizer toward persistently fine-grained analyses.

Across all tokenizers and vocabulary sizes, NSL values remained below one, indicating that each method achieves a measurable degree of compression relative to the baseline tokenizer used in Llama 3.1 8B Instruct, the backbone of our latest chat model (Yang et al., 2025b). BPE consistently attains the lowest NSL, underscoring its higher compression efficiency. Unigram LM shows the highest NSL values, suggesting that its segmentations are less compact relative to the baseline, while WordPiece again falls between Unigram LM and BPE.

Overall, results suggest that BPE provides the most compact and efficient segmentation for Hungarian, balancing granularity and compression. WordPiece offers a middle ground but does not outperform BPE in any metric. Unigram LM produces more fine-grained segmentations with higher fertility and CTC. The modified Unigram LM shows the same tendency in a more pronounced form, since it produces longer sequences and only limited changes as the vocabulary grows. At the same time, increasing the vocabulary size steadily improves BPE's alignment with morphological boundaries. These trends are reported in Table 4 and align with the empirical morphology alignment results reported in Table 3.

4.3. Impact of Tokenization on Pretraining

During pretraining, clear differences emerged in convergence behaviour across tokenization algorithms and vocabulary sizes (see Figure 1). All models exhibited rapid loss reduction within the first few thousand steps, followed by gradual stabilization after approximately one epoch. Among the compared approaches, the modified Unigram LM tokenizer consistently achieved the lowest final loss across all vocabulary sizes, with the best result at 16K (a loss value of 1.39 after two epochs). The standard algorithms yielded notably higher loss values, ranging from 2.14 to 2.85, with WordPiece performing the best among them at a 16K vocabulary size (2.14). The Unigram LM tokenizer achieved a slightly higher, but comparable loss value (2.24) with the 16K vocabulary. For BPE and WordPiece tokenizers, larger vocabularies (32K, 64K, and 128K) converged more slowly and plateaued at higher losses ranging from 2.4 to 2.9, suggesting that excessive vocabulary granularity increases data sparsity and hinders optimization. Specifically, BPE models reached final losses of 2.43, 2.65, 2.65, and 2.85 for the 16K, 32K, 64K, and 128K vocabularies, while WordPiece models followed a similar pattern with 2.14, 2.43, 2.49, and 2.72. The Unigram LM tokenizers exhibited the most stable performance across vocabulary sizes. The standard Unigram LM algorithm achieved losses between 2.24 and 2.40, while the modified Unigram LM tokenizers converged at a consistently lower loss, reaching 1.49, 1.44, and 1.45 for the 32K, 64K, and 128K vocabularies.

Overall, smaller vocabularies consistently outperformed larger ones across all algorithms, with the modified Unigram LM achieving the lowest absolute loss. These results indicate that increasing vocabulary size does not inherently improve training efficiency or model quality in morphologically rich languages.

4.4. Downstream Task Performance

All models were fine-tuned on the HuLU benchmark tasks as described in the Methods section, and the results are summarized in Table 5.

Despite its considerably higher fertility and corpus token counts, the modified Unigram LM tokenizer achieved the highest task-specific scores on average in every task, obtaining top results on *HuCOLA* (72.77), *HuCoPA* (57.49), *HuRTE* (56.22), *HuSST* (66.82), and *HuCB* (49.60). These results indicate that it performs particularly well on tasks involving semantic inference and sentence-level reasoning, where capturing frequent word-level co-occurrence patterns is advantageous. For instance, since *HuCOLA* rely on recognizing grammatical or referential structure, the seeded vocabulary's emphasis on morphemes may have provided a slight advantage in these cases.

Generally, all tokenizers performed competitively and improved with increasing vocabulary size, but in some cases performance slightly varied with vocabulary size. For example, the modified Unigram LM model with a 32K vocabulary achieved 56.22 on *HuRTE*, while its performance dropped to 52.89 at 128K. This sensitivity likely arises from Unigram LM's probabilistic-based segmentation, which could exhibit variability as the vocabulary grows and rare subwords proliferate. Given the moderate size of both the pretraining corpus and the fine-tuning datasets, the vocabulary construction process may have overfit to corpus-specific token distributions. As a result, many infrequent subwords were poorly represented during training, leading to less consistent downstream performance. This behaviour can be observed across other tokenization methods as well, where the relationship between vocabulary size and performance is not monotonic: for instance, WordPiece peaks at 64K on *HuCOLA*, *HuCoPA*, and *HuRTE*, before declining slightly at 128K, while Unigram LM similarly achieves its best on *HuCOLA*, *HuRTE*, *HuSST* and *HuCB* at 64K, compared to 128K. This pattern suggests that beyond a certain vocabulary threshold, data sparsity

Tokenizer	Vocab	Fertility	CTC	Avg. BPT	NSL
BPE	16K	**1.759**	**3 175 144 336**	**4.622**	**0.576**
	32K	**1.596**	**2 880 891 301**	**5.095**	**0.532**
	64K	**1.479**	**2 669 717 316**	**5.498**	**0.485**
	128K	**1.396**	**2 519 525 649**	**5.825**	**0.458**
Unigram LM	16K	1.911	3 447 805 725	4.257	0.625
	32K	1.751	3 160 304 792	4.644	0.573
	64K	1.656	2 988 280 812	4.911	0.542
	128K	1.612	2 908 523 690	5.046	0.528
WordPiece	16K	1.847	3 333 802 410	4.402	0.605
	32K	1.646	2 970 159 465	4.941	0.539
	64K	1.509	2 723 565 175	5.389	0.495
	128K	1.414	2 551 888 975	5.751	0.464
Modified Unigram LM	16K	2.0684	10 078 253 490	3.9321	0.67
	32K	2.0058	9 773 358 537	4.0548	0.66
	64K	1.9821	9 657 837 432	4.1033	0.65
	128K	1.9766	9 630 744 879	4.1148	0.65

Table 4: Intrinsic evaluation metrics. CTC denotes Corpus Token Count, BPT denotes Bytes per Token.

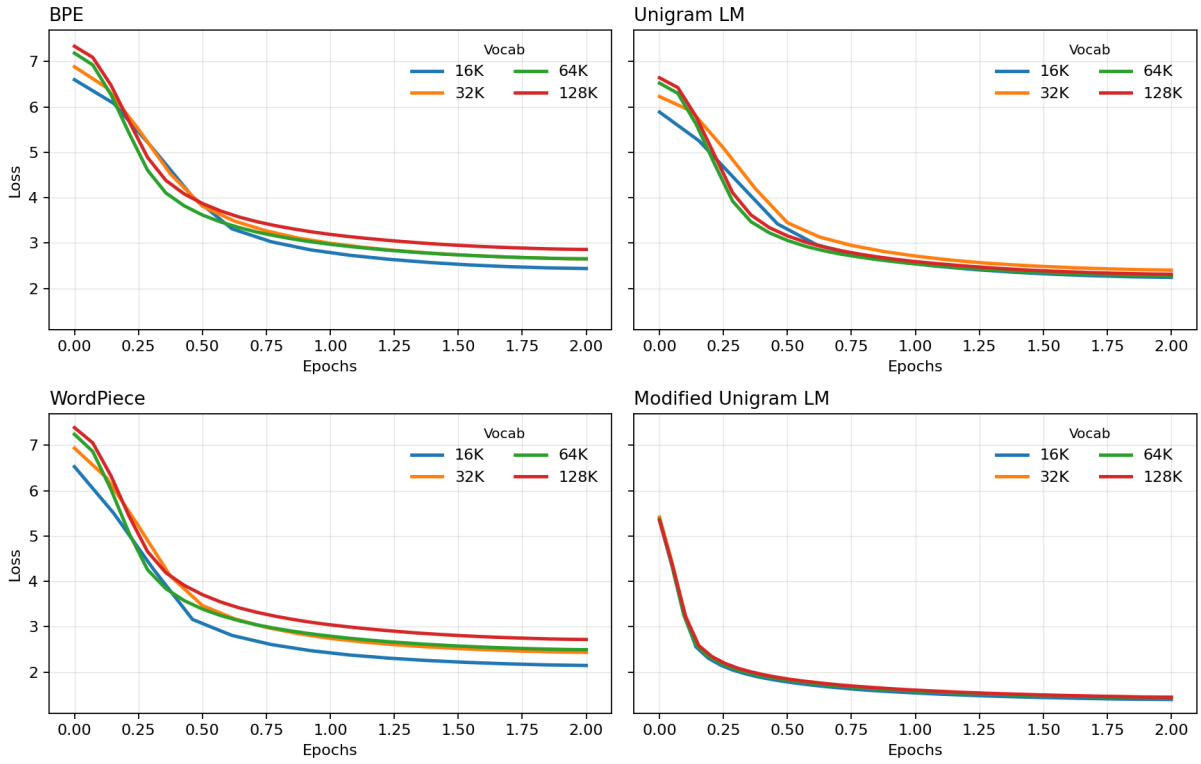

Figure 1: Validation loss across tokenization methods and vocabulary sizes during BERT pretraining.

may become a limiting factor, where many tokens appear infrequently in the training data, resulting in less effective learning and generalization.

Furthermore, high variability can be observed for the *HuCB* task, which can be interpreted as a consequence of fine-tuning on a small dataset, where the limited number of training examples makes it difficult for the model to generalize consistently, leading to unstable performance across runs. This instability suggests that results on *HuCB* should be interpreted with caution, as the reported means may not reliably reflect the true capacity of each tokenizer configuration.

Given the overlapping standard deviations observed across configurations, we conducted three more runs on HuLU benchmark and performed Kruskal-Wallis H-tests - a non-parametric statistical test for comparing multiple groups - to assess whether the differences in performance between

Tokenizer	Vocab	HuCOLA	HuCoPA	HuRTE	HuSST	HuCB
BPE	16K	68.78 ± 0.43	53.22 ± 3.02	50.00 ± 0.29	60.04 ± 3.19	32.80 ± 0.54
	32K	70.22 ± 1.08	**54.51 ± 3.91**	52.43 ± 2.59	61.82 ± 0.84	33.97 ± 0.63
	64K	**71.62 ± 1.20**	52.57 ± 1.04	52.34 ± 2.55	64.04 ± 1.80	43.75 ± 10.41
	128K	70.93 ± 0.32	53.11 ± 1.10	**54.24 ± 2.05**	**65.13 ± 0.17**	**47.73 ± 15.17**
WordPiece	16K	68.36 ± 0.56	51.90 ± 1.31	53.46 ± 3.61	60.95 ± 3.27	33.67 ± 0.33
	32K	68.15 ± 1.81	54.31 ± 0.49	53.00 ± 0.21	62.04 ± 3.06	38.50 ± 3.92
	64K	**72.24 ± 0.81**	**56.12 ± 1.10**	**54.44 ± 4.45**	63.82 ± 1.84	43.31 ± 5.58
	128K	71.97 ± 0.72	54.51 ± 1.11	52.87 ± 0.46	**65.72 ± 1.78**	**48.46 ± 14.45**
Unigram LM	16K	66.63 ± 1.19	52.93 ± 1.29	52.47 ± 2.56	61.01 ± 2.04	34.98 ± 1.64
	32K	67.23 ± 0.28	55.32 ± 0.90	53.72 ± 3.74	63.37 ± 1.63	37.30 ± 3.96
	64K	**71.34 ± 0.43**	53.09 ± 0.30	**55.72 ± 1.24**	**65.38 ± 2.30**	**40.57 ± 7.25**
	128K	71.26 ± 1.54	**56.72 ± 2.10**	53.45 ± 3.46	64.69 ± 1.65	35.05 ± 1.72
Modified Unigram LM	16K	70.20 ± 1.55	54.71 ± 0.90	55.06 ± 5.06	65.48 ± 2.05	37.61 ± 4.27
	32K	70.62 ± 0.81	56.91 ± 0.68	**56.22 ± 6.23***	63.97 ± 0.99	43.65 ± 10.31
	64K	**72.77 ± 1.11***	57.31 ± 0.29	55.77 ± 5.83	65.85 ± 1.30	43.81 ± 10.48
	128K	71.51 ± 1.19	**57.49 ± 0.49***	52.89 ± 0.56	**66.82 ± 2.32***	**49.60 ± 16.27***

Table 5: Models performance on HuLU benchmark. All scores are reported as balanced accuracy instead of accuracy. Best scores are highlighted in bold for each vocabulary size and with asterisk for each task.

vocabulary sizes and tokenizer methods are statistically significant for each task. Two research questions were formulated: (Q1) does vocabulary size significantly affect downstream task performance within each tokenizer method, and (Q2) does the choice of tokenizer method significantly affect downstream task performance within each vocabulary size. For each question, five tests were conducted - one per task - resulting in a total of 40 tests across both questions. To control for the family-wise error rate introduced by multiple comparisons, we applied Bonferroni correction, adjusting the usual significance threshold to $\alpha = 0.05/40 = 0.00125$.

The majority of comparisons yielded no statistically significant differences, as shown in Appendix Appendix E, suggesting that neither vocabulary size nor tokenizer choice consistently leads to significant performance differences across tasks. The only statistically significant effect was observed on *HuCoPA*, where vocabulary size significantly affected performance for both WordPiece ($H = 16.976, p = 0.0007$) and Unigram LM ($H = 18.071, p = 0.0004$), and tokenizer choice significantly affected performance at 64K vocabulary size ($H = 18.070, p = 0.0004$). This implies that *HuCoPA* is the most sensitive task to tokenization decisions, while the remaining tasks appear largely robust to both vocabulary size and tokenizer method under the current experimental conditions. We note that due to the relatively small size of the fine-tuning datasets, several runs produced identical evaluation scores, resulting in a high number of tied ranks. Therefore, Kruskal-Wallis results should be interpreted with caution, and the absence of significance should not be taken as evidence of equivalence between configurations.

Overall, the results indicate that the different tokenization methods performed similarly across all Hungarian downstream tasks, with only a small, but not significant improvement when using the modified Unigram LM tokenizer. All methods exhibit some variability in performance depending on vocabulary size and task type, with no consistent improvement observed as the vocabulary size increases. The modified Unigram LM achieves the highest task-specific peaks and outperforms its non-seeded counterpart. BPE offers a balanced trade-off between efficiency and interpretability. Unigram LM and WordPiece algorithms also demonstrate consistent generalization across both syntactic and semantic benchmarks. These findings highlight the importance of aligning tokenization strategy and vocabulary design with the linguistic characteristics of the target language and the specific objectives of the model.

5. Conclusion

We compared BPE, WordPiece, Unigram LM, and a morphology-seeded Unigram LM across multiple vocabulary sizes for Hungarian language modeling. BPE yielded the most compact segmentations and the strongest alignment with Hungarian morpheme boundaries, especially at larger vocabulary sizes. Smaller vocabularies consistently improved pretraining efficiency, while the modified Unigram LM achieved the lowest pretraining loss and the strongest average downstream results. However, differences between tokenization methods and vocabulary sizes were generally small and mostly not statistically significant, except for HuCoPA. Overall, our findings suggest that tokenizer choice matters for efficiency and linguistic coverage, but under the present setup its impact on downstream performance is limited.

6. Bibliographical References

Catherine Arnett, Marisa Hudspeth, and Brendan O'Connor. 2025. Evaluating Morphological Alignment of Tokenizers in 70 Languages.

Kaj Bostrom and Greg Durrett. 2020. Byte Pair Encoding is Suboptimal for Language Model Pretraining. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4617–4624, Online. Association for Computational Linguistics.

Iaroslav Chelombitko, Egor Safronov, and Aleksey Komissarov. 2024. Qtok: A Comprehensive Framework for Evaluating Multilingual Tokenizer Quality in Large Language Models.

Jonathan H. Clark, Dan Garrette, Iulia Turc, and John Wieting. 2022. CANINE: Pre-training an Efficient Tokenization-Free Encoder for Language Representation. *Transactions of the Association for Computational Linguistics*, 10:73–91.

Marco Cognetta, Vilém Zouhar, Sangwhan Moon, and Naoaki Okazaki. 2024. Two Counterexamples to Tokenization and the Noiseless Channel.

Mathias Creutz, Krista Lagus, Krista Lagus@hut, and Fi. 2005. Unsupervised morpheme segmentation and morphology induction from text corpora using Morfessor 1.0.

Zoltan Csaki, Bo Li, Jonathan Li, Qiantong Xu, Pian Pawakapan, Leon Zhang, Yun Du, Hengyu Zhao, Changran Hu, and Urmish Thakker. 2024. SambaLingo: Teaching Large Language Models New Languages.

Gautier Dagan, Gabriel Synnaeve, and Baptiste Rozière. 2024. Getting the most out of your tokenizer for pre-training and domain adaptation.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, and Al-Dahle. 2024. The Llama 3 Herd of Models.

Valentin Hofmann, Hinrich Schuetze, and Janet Pierrehumbert. 2022. An Embarrassingly Simple Method to Mitigate Undesirable Properties of Pretrained Language Model Tokenizers. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 385–393, Dublin, Ireland. Association for Computational Linguistics.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023. Mistral 7B.

Taku Kudo. 2018. Subword Regularization: Improving Neural Network Translation Models with Multiple Subword Candidates.

Yanis Labrak, Adrien Bazoge, Beatrice Daille, Mickael Rouvier, and Richard Dufour. 2024. How Important Is Tokenization in French Medical Masked Language Models?

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2019. ALBERT: A lite BERT for self-supervised learning of language representations. *CoRR*, abs/1909.11942.

Noémi Ligeti-Nagy, Gergő Ferenczi, Enikő Héja, László János Laki, Noémi Vadász, Zijian Győző Yang, and Tamás Váradi. 2024. HuLU: Hungarian Language Understanding Benchmark Kit. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 8360–8371, Torino, Italia. ELRA and ICCL.

Yihong Liu, Peiqin Lin, Mingyang Wang, and Hinrich Schuetze. 2024. OFA: A framework of initializing unseen subword embeddings for efficient large-scale multilingual continued pretraining. In *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 1067–1097, Mexico City, Mexico. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach.

Beso Mikaberidze, Temo Saghinadze, Guram Mikaberidze, Raphael Kalandadze, Konstantine Pkhakadze, Josef van Genabith, Simon Ostermann, Lonneke van der Plas, and Philipp Müller. 2024. A Comparison of Different Tokenization Methods for the Georgian Language. In *Proceedings of the 7th International Conference on Natural Language and Speech Processing (ICNLSP 2024)*, pages 199–208, Trento. Association for Computational Linguistics.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient Estimation of Word Representations in Vector Space.

Luca Moroni, Giovanni Puccetti, Pere-Lluis Huguet Cabot, Andrei Stefan Bejgu, Edoardo Barba, Alessio Miaschi, Felice Dell'Orletta, Andrea Esuli, and Roberto Navigli. 2025. Optimizing LLMs for Italian: Reducing Token Fertility and Enhancing Efficiency Through Vocabulary Adaptation.

Attila Novák, Borbála Siklósi, and Csaba Oravecz. 2016. A new integrated open-source morphological analyzer for Hungarian. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016)*, Paris, France. European Language Resources Association (ELRA).

Pedro Javier Ortiz Suárez, Benoît Sagot, and Laurent Romary. 2019. Asynchronous pipelines for processing huge corpora on medium to low resource infrastructures. Proceedings of the Workshop on Challenges in the Management of Large Corpora (CMLC-7) 2019. Cardiff, 22nd July 2019, pages 9 – 16, Mannheim. Leibniz-Institut für Deutsche Sprache.

Faisal Qarah and Tawfeeq Alsanoosy. 2024. A Comprehensive Analysis of Various Tokenizers for Arabic Large Language Models. *Applied Sciences*, 14(13):5696.

Alec Radford, Jeff Wu, R. Child, D. Luan, Dario Amodei, and I. Sutskever. 2019. Language Models are Unsupervised Multitask Learners.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *CoRR*, abs/1910.10683.

Phillip Rust, Jonas Pfeiffer, Ivan Vulić, Sebastian Ruder, and Iryna Gurevych. 2021. How Good is Your Tokenizer? On the Monolingual Performance of Multilingual Language Models.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of BERT: smaller, faster, cheaper and lighter. *CoRR*, abs/1910.01108.

Craig W. Schmidt, Varshini Reddy, Haoran Zhang, Alec Alameddine, Omri Uzan, Yuval Pinter, and Chris Tanner. 2024. Tokenization Is More Than Compression.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural Machine Translation of Rare Words with Subword Units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

S. Tamang and D. J. Bora. 2024. Evaluating Tokenizer Performance of Large Language Models Across Official Indian Languages.

Yi Tay, Vinh Q. Tran, Sebastian Ruder, Jai Gupta, Hyung Won Chung, Dara Bahri, Zhen Qin, Simon Baumgartner, Cong Yu, and Donald Metzler. 2022. Charformer: Fast Character Transformers via Gradient-based Subword Tokenization.

Gemma Team, Thomas Mesnard, Cassidy Hardin, Robert Dadashi, Surya Bhupatiraju, and Shreya Pathak. 2024. Gemma: Open Models Based on Gemini Research and Technology.

Cagri Toraman, Eyup Halit Yilmaz, Furkan Şahinuç, and Oguzhan Ozcelik. 2023. Impact of Tokenization on Language Models: An Analysis for Turkish. *ACM Transactions on Asian and Low-Resource Language Information Processing*, 22(4):1–21.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention Is All You Need.

Saketh Reddy Vemula, Dipti Misra Sharma, and Parameswari Krishnamurthy. 2025. Rethinking Tokenization for Rich Morphology: The Dominance of Unigram over BPE and Morphological Alignment.

Veronika Vincze, Katalin Simkó, Zsolt Szántó, and Richárd Farkas. 2017. Universal Dependencies and Morphology for Hungarian - and on the Price of Universality. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 356–365, Valencia, Spain. Association for Computational Linguistics.

Dixuan Wang, Yanda Li, Junyuan Jiang, Zepeng Ding, Ziqin Luo, Guochao Jiang, Jiaqing Liang, and Deqing Yang. 2025. Tokenization Matters! Degrading Large Language Models through Challenging Their Tokenization.

Anna Wegmann, Dong Nguyen, and David Jurgens. 2025. Tokenization is Sensitive to Language Variation.

BigScience Workshop, Teven Le Scao, Angela Fan, Christopher Akiki, Ellie Pavlick, and Suzana Ilić. 2023. BLOOM: A 176B-Parameter Open-Access Multilingual Language Model.

Shangda Wu, Xu Tan, Zili Wang, Rui Wang, Xiaobing Li, and Maosong Sun. 2024. Beyond Language Models: Byte Models are Digital World Simulators.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Łukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2016. Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation.

Linting Xue, Aditya Barua, Noah Constant, Rami Al-Rfou, Sharan Narang, Mihir Kale, Adam Roberts, and Colin Raffel. 2022. ByT5: Towards a Token-Free Future with Pre-trained Byte-to-Byte Models. *Transactions of the Association for Computational Linguistics*, 10:291–306.

Zijian Győző Yang, Ágnes Bánfi, Réka Dodé, Gergő Ferenczi, Péter Hatvani, Enikő Héja, Mariann Lengyel, Gábor Madarasz, Mátyás Osváth, Bence Sárossy, Kristóf Varga, Tamás Váradi, Gábor Prószéky, and Noémi Ligeti-Nagy. 2025a. ChatPULI: Enhancement to the first Hungarian conversational model. *Annales Mathematicae et Informaticae*, 61:261–274.

Zijian Győző Yang, Ágnes Bánfi, Réka Dodé, Gergő Ferenczi, Péter Hatvani, Enikő Héja, Mariann Lengyel, Gábor Madarasz, Mátyás Osváth, Bence Sárossy, Kristóf Varga, Tamás Váradi, Gábor Prószéky, and Noémi Ligeti-Nagy. 2025b. PULI Chat Our First Hungarian Conversational Model. Eger, Hungary. International Conference on Formal Methods and Foundations of Artificial Intelligence.

Győző Yang Zijian, Enikő Héja, and Tamás Váradi. 2024a. The First Instruct-Following Large Language Models for Hungarian.

Győző Yang Zijian, Enikő Héja, and Tamás Váradi. 2024b. ParancsPULI: Az utasításkövető PULI-modell.

Vilém Zouhar, Clara Meister, Juan Luis Gastaldi, Li Du, Mrinmaya Sachan, and Ryan Cotterell. 2023. Tokenization and the Noiseless Channel.

Appendix A. Text pre-processing

The pre-processing pipeline consisted of filtering texts based on the grammatical correctness (linguistic acceptability) of documents and handling duplicate sentences (based on exact-duplicate match). Furthermore, data cleaning were applied to the documents. Page numbers, excessible whitespace, image captions, email-addresses, non-Hungarian characters, very short lines (<3 lines) and hyphenation at line endings were removed.

Appendix B. BERT language model pretraining

Hyperparameter	Value
Batch size	96
Epoch	2
Adam ϵ	1×10^{-6}
Adam β_1	0.9
Adam β_2	0.999
Learning rate	5×10^{-5}
Learning rate schedule	Linear
Sequence length	1024
Attention dropout	0.1
Dropout	0.1
Hidden Size	768
Number of Attention Heads	12
Number of Hidden Layers	12

For the pretraining, the batch size was determined by the available GPU allocation, resulting in a smaller effective batch size than in some standard large-scale training pipelines.

Appendix C. HuLU benchmark tasks

The HuLU benchmark comprises a diverse suite of Hungarian natural language understanding tasks designed to assess various aspects of linguistic and semantic competence in transformer-based models:

- **HuCOLA (Hungarian Corpus of Linguistic Acceptability)** includes 9,076 Hungarian sentences annotated for grammatical acceptability (binary 0/1). The material was sourced by two linguists from three academic grammar texts, and each sentence was evaluated independently by four annotators. The majority label was taken as the gold standard. The corpus is split into training (7,276; 80%), validation (900; 10%), and test (900; 10%) partitions.
- **HuCoPA (Hungarian Choice of Plausible Alternatives)** consists of 1,000 examples, each comprising a premise and two possible alternatives. The model must select the alternative that stands in a causal relationship with the premise. The dataset is a high-quality Hungarian translation and re-annotation of the original

English CoPA corpus, divided into 400 training, 100 validation, and 500 test items.

- **HuRTE (Hungarian Recognizing Textual Entailment)** contains 4,504 instances, each composed of a (sometimes multi-sentence) premise and a one-sentence hypothesis. The model must decide whether the former entails the latter or not. The corpus was created by translating and re-annotating the instances of the RTE datasets that are part of the GLUE benchmark. The train, validation and test set contain 2,131; 242 and 2,131 instances, respectively.
- **HuSST (Hungarian Stanford Sentiment Treebank)** comprises 11,683 sentences annotated on a three-point sentiment scale. The dataset was produced through machine translation and human re-annotation of the complete sentences from the English Stanford Sentiment Treebank. The corpus is divided into 9,347 training, 1,168 validation, and 1,168 test sentences.
- **HuCB (Hungarian CommitmentBank)** features short text fragments containing sentences with embedded clauses governed by inference-canceling operators. Each instance pairs a premise (the full fragment) with a hypothesis (the embedded clause), and the model must infer the author's degree of commitment to the truth of the hypothesis. The dataset is split into 250 training, 103 validation, and 250 test examples.

Appendix D. Hyperparameters for BERT language model training

Hyperparameter	Value
Batch size	32
Adam ϵ	1×10^{-6}
Adam β_1	0.9
Adam β_2	0.999
Learning rate	5×10^{-5}
Learning rate schedule	Linear
Attention dropout	0.1
Dropout	0.1

Appendix E. Results of the Kruskal-Wallis Tests

Tokenizer	Task	H-stat	p-value
BPE	HuCOLA	2.544	0.4673
	HuCoPA	1.095	0.7782
	HuRTE	11.444	0.0096
	HuSST	10.405	0.0154
	HuCB	6.316	0.0972
WordPiece	HuCOLA	4.334	0.2276
	HuCoPA	**16.976**	**0.0007***
	HuRTE	0.569	0.9035
	HuSST	5.469	0.1405
	HuCB	13.690	0.0034
Unigram LM	HuCOLA	4.729	0.1928
	HuCoPA	**18.071**	**0.0004***
	HuRTE	2.849	0.4156
	HuSST	7.461	0.0586
	HuCB	1.552	0.6704
Modified Unigram LM	HuCOLA	3.070	0.3810
	HuCoPA	14.786	0.0020
	HuRTE	0.471	0.9252
	HuSST	4.972	0.1739
	HuCB	1.552	0.6704

Table 6: Kruskal-Wallis test results for Q1: effect of vocabulary size on downstream task performance within each tokenizer method. Significance assessed using Bonferroni-corrected threshold ($\alpha = 0.00125$). Bold entries denote $p < 0.00125$.

Vocab	Task	H-stat	p-value
16K	HuCOLA	3.221	0.3587
	HuCoPA	4.929	0.1771
	HuRTE	2.850	0.4153
	HuSST	9.883	0.0196
	HuCB	9.000	0.0293
32K	HuCOLA	0.809	0.8473
	HuCoPA	7.119	0.0682
	HuRTE	0.641	0.8869
	HuSST	3.823	0.2812
	HuCB	2.875	0.4113
64K	HuCOLA	0.372	0.9460
	HuCoPA	**18.070**	**0.0004***
	HuRTE	1.114	0.7737
	HuSST	4.929	0.1771
	HuCB	1.593	0.6610
128K	HuCOLA	2.046	0.5629
	HuCoPA	14.786	0.0020
	HuRTE	0.033	0.9984
	HuSST	2.187	0.5346
	HuCB	1.801	0.6147

Table 7: Kruskal-Wallis test results for Q2: effect of tokenizer method on downstream task performance within each vocabulary size. Significance assessed using Bonferroni-corrected threshold ($\alpha = 0.00125$). Bold entries denote $p < 0.00125$.