
Comparative Intrinsic Evaluation of LLM Tokenizers on Serbian Text

Mihailo Škorić¹[\[0000-0003-4811-8692\]](#), Nikola Janković²[\[0000-0003-3484-4220\]](#),
Tanja Samardžić³[\[0000-0001-6451-3946\]](#)

¹ University of Belgrade, Faculty of Mining and Geology, Serbia,
mihailo.skoric@rgf.bg.ac.rs

² University of Belgrade, Faculty of Philology, Serbia,
nikolajankovickv@gmail.com

³ IDSIA USI-SUPSI, Lugano, Switzerland, tanja.samardzic@supsi.ch

Scientific paper

DOI: 10.18485/judig.2026.2.ch9

Abstract

This study presents a comparative intrinsic evaluation of large language model tokenizers trained specifically for Serbian, a language characterized by rich morphology and dual script usage. We construct several tokenizers using diverse strategies, including GPT2-style and trailing space BPE, as well as some experimental approaches: minimum redundancy exploration, casing and script compression, and morphology-aware tokenization (based on a Serbian morphological dictionary). In addition, we benchmark tokenizers from contemporary models such as Qwen3.5 and Mistral3.5. Training is conducted on a Serbian corpus of approximately one billion words, with evaluation performed on a held out set one-third of that size. Intrinsic properties are analysed using Tokenizer Analysis Suite, calculating metrics such as fertility, compression ratio, Rényi Entropy and MorphScore. Our results highlight systematic differences different tokenizer building strategies: morphology driven approaches achieve higher MorphScore, while script compression strategies result in lower fertility and improved compression, especially while handling mixed Cyrillic–Latin corpora. In comparison, existing general purpose tokenizers reveal that they are consistently inferior to language specific tokenizers across most intrinsic metrics, underscoring the importance of tailored tokenizer design.

This work provides the initial comparative study of Serbian tokenizers, offering insights into how design choices affect the intrinsic properties. By systematically evaluating both newly trained and existing models, we establish a foundation for future research on tokenizer development for morphologically rich languages, and contribute benchmarks that can help future designing of tailored, specific purpose tokenizers.

Keywords: Tokenization, Serbian, Intrinsic evaluation, LLMs

1. Introduction

Subword tokenization, originally introduced in machine translation (Sennrich et al., 2016) and then in other Natural Language Processing (NLP) tasks, opened the question of how the subword vocabulary should be designed. An early proposal was to set the vocabulary size to $0.4 \times$ the initial (pre-tokenization) vocabulary (Mielke et al., 2019). Other proposals were to optimize the vocabulary for a particular task (Kudo, 2018) or to take into account subword morphology (Park et al., 2021), which has been considered an appropriate approach to languages with richer subword structures. What can be observed as a global trend in practical solutions is an increase in the vocabulary size from around 8,000 units that were considered optimal for processing Wikipedia data to around 128,000 that are used in present-day multilingual models. More recently, the optimal size of the vocabulary has been related to the model size: for training larger models, one should aim at a larger vocabulary (Tao et al., 2024). On the other hand, smaller vocabularies might be better adapted in the settings that include low-resource languages (NLLB-Team, 2022).

The question of the vocabulary design is especially challenging in the context of the increasingly multilingual large language models (LLMs), which are trained on multilingual data mixtures and intended to be portable to diverse languages (Conneau et al., 2020; NLLB-Team, 2022; Apertus, 2025). Multilingual models are often taken as the starting point for developing language-specific models, but their pre-trained vocabularies currently represent an important obstacle to model adaptation to a new language. Subword vocabularies that are shared between many languages tend to favour the languages that are better represented in the training mixtures. When applied to less resourced languages, they tend to result in over-segmentation of input data (Petrov et al., 2023). This over-segmentation is shown to impact considerably downstream performance, while increasing the costs of processing (Ahia et al., 2023).

Language-specific vocabularies have been shown to give better results than shared multilingual vocabularies in terms of both downstream performance and data-efficiency (Rust et al., 2021; Liang et al., 2023), but mapping a new vocabulary to a pre-trained model remains an open research problem (Minixhofer et al., 2025).

Due to these concerns the tokenization quality has started to be evaluated *intrinsically*, that is independently of downstream model performance, as a means of enabling informed tokenization choices, which also impact downstream performance (Ali et al., 2024; Kanjirangat et al., 2025).

In this paper, we present an initial comparison between several multilingual and language-specific subword vocabularies using a range of intrinsic evaluation metrics and focusing on Serbian as the target language. Our findings show a clear advantage of language-specific vocabularies in terms of intrinsic measures establishing an evaluation framework for future steps towards developing an optimal tokenizer for Serbian.

The remainder of this paper is organized as follows: the next section (section 2.) describes the training and testing datasets as well as the morphological resources employed; Section 3. introduces different tokenizers and their building strategies; Section 4. presents the evaluation framework and intrinsic metric results based on an pre-existing tokenizer evaluation suite; and the final section (section 5.) discusses the implications of our findings and outlines directions for future research on Serbian tokenizer development.

2. Data

For this study, we rely on a large, carefully prepared corpus of Serbian text designed to capture the language’s full morphological richness and dual script usage (Cyrillic and Latin). The training dataset comprises approximately one billion words drawn from diverse sources, ensuring balanced representation of orthographic variation. To enable robust evaluation, a held-out test set of roughly one third of the corpus was constructed under the same conditions.

In addition to this we also employ several morphological resources: Serbian morphological dictionary and three curated affix/root lists for building the morphologically-aware tokenizer *Morfo-Tok* and Universal Dependencies Serbian-SET treebank which is used as a held-out set for the calculation of MorphScore metric (Arnett and Bergen, 2025).

2.1. Training and testing dataset

For training and evaluation we employ *SrTokenizerTest* corpus, large-scale Serbian text resource specifically curated for tok-

enizer development. The dataset unifies multiple sources in both Cyrillic and Latin scripts into a consistent JSONL format with id and text fields, stripped of metadata for simplicity. It totals approximately 11.3 GB of raw text or around 1.3 billion words, combining three major components: the Serbian subset of *Znanje* corpus, a highly curated collection of South Slavic scientific and professional publications (700 million words) by Škorić and Janković (2024); the *Wiki.korpus*, resource built on a fresh May 2026 snapshot of Serbian Wikipedia, Wikisource, Wikibooks, Wikinews, and Wikiquotes (135 million words) by Škorić and Palma (2026); and the *PDRS 1.0* web corpus, which resulted from a crawl of the .rs domain in 2022 (500 million words) by Wasserscheidt (2023).

By combining curated scientific texts (written in both Cyrillic and Latin), encyclopedic resources (written predominantly in Cyrillic), and diverse web content (all Latin), this corpus provides both scale and linguistic diversity, making it well-suited for assessing intrinsic tokenizer properties in the context of Serbian’s dual script and morphological complexity.

Each source is processed and split into train (75%) and test (25%) set, ensuring robust evaluation on held-out material. The combined training dataset is set to hold around a billion words, while the held out test set contains a third of that number, but they are spread across the same range of domains (science, encyclopedia and web). The whole dataset is published on HuggingFace¹. The composition of the corpus is briefly presented in Table 1.

Table 1: Composition of the *SrTokenizerTest* Dataset

Corpus	Script	Size (GB)	Word Count (Millions)
Znanje (sr)	Mixed	6.6	700
Wiki.korpus (sr)	Cyrillic	1.5	135
PDRS 1.0	Latin	3.2	500
Total	Mixed	11.3	1,335

2.2. Morphological data

For building the morphologically-aware tokenizer *MorfoTok* we employ the following morphological resources:

¹ <https://huggingface.co/datasets/procesaur/sr-tokenizer-test>

- A Cyrillic lemma–form dictionary mapping around 280 thousand lemmas in Serbian to millions of inflected forms, which was assembled by combining permissively-licensed and open Serbian morphological dictionaries produced by [Smiljanić \(2019\)](#) and [Krstev et al. \(2020\)](#).
- Three affix/root lists curated specially for this experiment: 210 prefixes, 194 derivational suffixes, and 32 compounding heads (productive bound roots that head compounds, e.g. **-pad** in **vodopad**; kept separate from suffixes because they are roots, not derivational morphemes).

For the calculation of *MorphScore* metric we use a held-out set provided by the authors and based on the Universal Dependencies *Serbian-SET* treebank. This data set was manually annotated as part of the Serbian linguistic training corpus *SETimes.SR* [Batanović et al. \(2023\)](#). Among other annotation layers, this corpus includes lemmatization, which is used to extract automatically morphological boundaries for the *MorphScore*.

3. Tokenizers

To explore the impact of different design choices on the subword tokenization of Serbian text, we construct a diverse set of models ranging from established approaches to novel experimental strategies. Standard GPT-2 style byte pair encoding or BPE ([Radford et al., 2019](#)) and trailing space BPE serve as baselines, while more exploratory methods include BPE-based minimum redundancy exploration, casing and script compression, and morphology-aware tokenization built from a Serbian morphological dictionary. We further extend this line with a hybrid setup of combining case and script compression with different approaches. Alongside the newly trained tokenizers, we benchmark contemporary general-purpose models such as *Qwen3.5* and *Mistral3.5*, as well as the newest Serbian-specific tokenizer published recently for an encoder-only Serbian model.

All new tokenizers share a consistent setup to ensure comparability. We fixed the vocabulary size at 30,000 subword units, around the same as for the foundational BERT model ([Devlin et al., 2018](#)). The initial alphabet was derived from the ByteLevel pre-tokenizer, guaranteeing coverage of all characters encountered in raw text. Each tokenizer adds the same set of special tokens — **[PAD]**, **[UNK]**, **[CLS]**, **[SEP]**, and **[MASK]** — to support downstream language-modelling

tasks, and uses a `ByteLevel` pre-tokenizer that adds prefix spaces, for consistency. Decoding was handled by a `ByteLevel` decoder to ensure robust reconstruction of text. The trained tokenizers were learned on the same corpus of approximately one billion words using the HuggingFace tokenizer library; training data were streamed from disk with a batch iterator yielding chunks of 10,000 examples.

For most of the tokenizers, training was conducted using the *BpeTrainer*, parameterised with the fixed vocabulary size, special tokens, and initial alphabet. The morphology-aware tokenizer (*MorphoTok*) departs from this training procedure. Rather than learning merges from corpus statistics, its subword inventory is selected from a Serbian morphological knowledge base together with corpus frequencies, and then injected into a complementary BPE as entries with synthesized merges.

Each tokenizer was built twice to account for the orthographic duality of Serbian. In the first configuration, the entire dataset was transliterated into Latin script, following a common practice in Serbian NLP that simplifies preprocessing and aligns with many existing multilingual models. In the second configuration, training was performed on the unaltered dataset containing both Cyrillic and Latin scripts, thereby preserving the natural distribution of orthography in real-world usage. This dual training strategy allows us to systematically assess how script normalization versus mixed-script representation influences the calculated metrics. In each scenario, the testing is performed on the same kind of dataset: if the tokenizer was trained with Latin text only, the testing dataset is also transliterated before evaluation. This setup ensures that our evaluation framework not only benchmarks tokenizer architectures but also tests their robustness to script variation, a critical factor for languages with multiple writing systems.

3.1. GPT2-style BPE (Baseline)

The standard GPT-2 style BPE tokenizer serves as our baseline, reflecting the widely adopted approach introduced in (Radford et al., 2019). This method applies byte-pair encoding over a byte-level representation of text, ensuring that all characters from both scripts are covered. By operating at the byte level, the tokenizer avoids issues with out-of-vocabulary characters and maintains robustness across diverse inputs. The vocabulary size is fixed at 30,000 units,

and the merges are learned directly from the training corpus. As the current state-of-the-art, this design provides a strong reference point for evaluating more experimental tokenizers, as it balances efficiency and generality while lacking explicit morphological or script-aware adaptations. We train one model on the Latin-only dataset (*bpe*) and one on the mixed script dataset (*bpe_mix*).

3.2. Trailing Space BPE (Secondary Baseline)

The Trailing Space BPE tokenizer extends the standard GPT-2 style approach by introducing two key modifications tailored to Serbian morphology. First, it employs a *Whitespace* pre-tokenizer in combination with the ByteLevel pre-tokenizer, ensuring that word boundaries are explicitly recognized before subword segmentation. Second, it incorporates an end-of-word suffix marker, set to $\mathring{\text{G}}$, which encodes trailing spaces as part of the learned vocabulary. This results in a BPE tokenizer, where whole words and suffixes end with $\mathring{\text{G}}$ indicating end of the word, which is contrary to GPT-2 BPE, where it denotes the beginning of a new word. This design choice allows the tokenizer to more faithfully capture word boundaries and suffixes, which are particularly important in Serbian given its rich inflectional morphology and high frequency of suffixal variation. Trailing space BPE is expected to achieve better alignment with morphological structure, potentially reducing fertility and improving compression relative to standard BPE approach. Like for the previous, we train one model on the Latin-only dataset (*ts_bpe*) and one on the mixed script dataset (*ts_bpe_mix*).

3.3. Minimum Redundancy BPE (mireBPE)

These tokenizers are intended to exploit the fact that BPE is a general data compression algorithm that reduces the redundancy of text data by discovering the most frequent subword patterns merging iteratively the most frequent pairs of symbols. It has been shown, however, that this behaviour applies only in the first BPE steps, approximately 300. At this point, text redundancy minimum is reached and further BPE steps do no result in further compression (Gutierrez-Vasques et al., 2021). This fact opens a possibility for more efficient creation of subword vocabularies.

Instead of iterating over text data many times to find only one symbol per iteration, we propose to create the subword vocabulary in two phases illustrated in Table 2 with a constructed English example.

Table 2: Merging vs. concatenation in mireBPE tokenizers. Subword segments are represented with "-", pre-tokens (words) with white spaces.

BPE step 0	g-o-d c-r-e-a-t-e-d t-h-e h-e-a-v-e-n
Merging phase	god c-r-e-a-t- ed the h-e-a-v- en
Concatenation phase	god creat- ed the heav- en

Merging phase We train BPE in the standard way up to the point of minimum redundancy.

Concatenation phase We populate the rest of the vocabulary with the most frequent subword tokens created by concatenating all the remaining single adjacent characters.

Following this general idea, we try out three transition points, same for both Latin and mixed script tokenizer. In addition to the actual minimum redundancy point detected at step 291 (base alphabet + 30 merges), we also picked manually two later transition points, one at symbol 768 (507 merges), and one at symbol 1275 (double the merge amount from the first control point). We refer to these later transition points as **ctrl-1** and **ctrl-2** respectively. The purpose of trying later transition points after the redundancy minimum was to verify whether intuitive interpretations of the initial vocabularies would result in better overall tokenization.

Figure 1 shows the values of text (Shannon) entropy and redundancy in batches of 10 BPE steps from step 261 (end of the initial vocabulary) to step 741 (**ctrl-1**) of the merging phase. Two graphs represent two versions of the training data, only Latin and a mix of Latin and Cyrillic scripts. We can see that the point of minimum redundancy is the same in both versions of data coinciding with a rather sharp flexion in entropy growth. These values confirm that BPE does not find subword regularities that would compress text data beyond step 291. The monotone growth in both redundancy and entropy is caused by new symbols added to the vocabulary (one at each step). We build a total of six mireBPE tokenizers, two for each control point: *mireBPE* and *mireBPE_mix*, *mireBPE_ctrl-1* and *mireBPE_ctrl-1_mix*, *mireBPE_ctrl-2* and *mireBPE_ctrl-2_mix*.

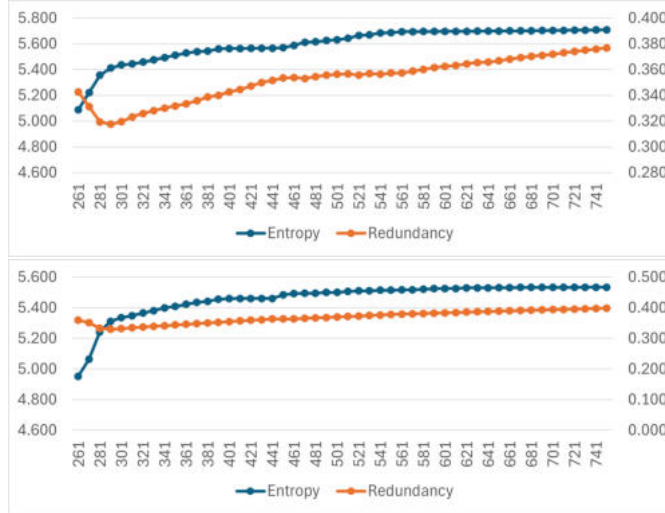


Figure 1: Shannon entropy (y-axis, left) and redundancy (y-axis, right) measured at increasing BPE steps (x-axis) of the merging phase during the training of trailing space BPE for Latin only (top) and mixed (bottom).

3.4. Casing and Script Compression (Srna)

The casing and script compression models introduce a lightweight normalization layer prior to BPE training, with the goal of reducing the number of redundant tokens by collapsing orthographic variants into a shared representation, while marking distinctions with special tokens. Three different cases are covered:

Capitalization handling A special regular expression² identifies capitalised words. Detected word is then lowercased, and a special token `<cap>` is prepend. For example, `Text` becomes `<cap> text`. This ensures that `text` and `Text` are represented with the same base token, with capitalization encoded separately. The space behind the word is preserved as well, to ensure proper segmentation.

Uppercase handling Same as for the previous case a regular expression³ matches fully uppercase words. The `mark_up` function prepends a `<up>` token and lowercases the text, so `TEXT` becomes `<up> text`. This reduces duplication between lowercase and uppercase variants of the same text/tokens.

² $\backslash s? \backslash b \backslash p \{Lu\} \backslash p \{Ll\} + \backslash b$

³ $\backslash s? \backslash b \backslash p \{Lu\} \{2,\} \backslash b$

Script compression Like before, a regular expression⁴ detects Cyrillic text. The whole block of text is transliterated into Latin using character mapping, and is then wrapped with `<boc>` (begin-of-Cyrillic) and `<eoc>` (end-of-Cyrillic) markers. For instance, `текст` becomes `<boc> tekst<eoc>`. This allows Cyrillic and Latin forms to share the same token, while still preserving script information.

After normalization, the text is passed to a standard BPE trainer with the same vocabulary size and configuration as other models. Hypothetically, this approach preserves the strengths of BPE while reducing redundancy, thereby freeing vocabulary space for more meaningful subword units (Pawar et al., 2025). Using this approach, we expect improvements in compression ratio and reductions in fertility, especially in Serbian where capitalization and script variation otherwise inflate token counts.

The main trade-off of the approach is decoding complexity: while capitalization restoration is not ambiguous, transliterating Latin back into Cyrillic when markers are present, must be handled carefully, as this task is not yet solved for the Serbian language. We train two version of this tokenizer, one with (*SrnaCF*) and one without case handling (*Srna*), in order to properly assess pros and cons of both compressions.

3.5. Morphology-based (MorfoTok)

In this approach, we construct a subword tokenizer for Serbian that is morphology-aware and bilingual across the language’s two scripts (Magner, 2001). The design follows a line of linguistically informed tokenization for morphologically rich languages (Ataman et al., 2017; Huck et al., 2017; Držík and Forgac, 2024) and is built in three phases:

Building morpheme inventory Serbian is productive and phonologically alternating, so one morpheme surfaces with different spellings (e.g. `junak` → `junaci`). Aided by curated prefix, suffix, and compounding-root lists each form from the dictionary is split into a prefix, root, derivational suffix, and inflectional ending, with the suffix exposed at several granularities (for

⁴ `\s?\p{Cyrillic}[\p{Latin}]*`

radošću: ending **u**, derivational **ost**, and the mutated surface joint **ošću**) so the optimizer can choose whichever tokenizes most economically. Morpheme frequencies are then counted over a large corpus to create candidate list.

Vocabulary optimization From these candidates (word-initial forms space-prefixed, suffixes bare), we select the inventory that minimizes total token count. A Viterbi segmentation (Kudo, 2018) computes, per word, the cheapest partition into vocabulary tokens and byte-pair-encoding fallback spans (Sennrich et al., 2016); the fallback is a 768-token byte-level BPE trained only on the spans the morphemes leave uncovered, making the two complementary. A round-based refinement minimizes total tokens rather than coverage, favouring whole frequent inflected forms — a deliberate trade of morpheme-boundary recall for lower fertility.

Final tokenizer compilation Rather than registering the morphemes as add-on tokens we inject them into the BPE model itself: each morpheme is a space-prefixed (word-initial) or bare (mid-word) vocabulary entry, made reachable by synthesizing the merge chain that builds it, processed shortest-first against the current merge set so it reproduces at run time. Because these merges add intermediate tokens that also consume the budget, we keep the highest-priority morphemes whose full merge-dependency closure fits and top up from a reserve to exactly 30,000 tokens. The result is an ordinary `PreTrainedTokenizerFast` with a `ByteLevel` pre-tokenizer (`add_prefix_space=True`) and decoder; its special tokens are `[PAD]`, `[UNK]`, `[CLS]`, `[SEP]`, `[MASK]` plus the per-configuration script/casing control tags below.

To address digraphia, an optional bidirectional preprocessor is also introduced to transliterate Cyrillic to Latin, optionally factoring casing into `<|CAP|>`, `<|UPPER|>` and `<|CASE|>` tags, and marking embedded foreign words with `<|LAT|>`; a unified wrapper inverts this on decode. By applying different levels of optional preprocessing, we build a total of four tokenizers: two without case and script compression (*MorfoTok_base* and *MorfoTok_base_mix*, one with script compression only (*MorfoTok* and one with both script and case compression (*MorfoTokCF*). The pairwise comparisons of these tokenizers

isolate the cost of a mixed-script vocabulary (*MorfoTok_base_mix* vs. *MorfoTok_base*), of the two script-control tags (*MorfoTok_base* vs. *MorfoTok*), and of surface versus factored casing (*MorfoTok* vs. *MorfoTokCF*).

3.6. Other included tokenizers

In addition to our newly constructed models, we benchmark two widely used contemporary tokenizers to provide external reference. The *Qwen3.5* tokenizer represents the latest generation of tokenization strategies from the Qwen family of large language models Team (2026), optimized for multilingual coverage and efficient handling of diverse scripts. It employs a variant of BPE with refinements in vocabulary construction and normalization, with roughly 152 thousand vocabulary entries and making it a strong reference point for evaluating Serbian tokenization against a state-of-the-art general-purpose system. The *Mistral3.5* tokenizer, taken from the newly released Mistral Medium model AI (2024), reflects another high-performance design that emphasizes compact vocabularies and efficient segmentation across European languages with somewhat smaller multilingual vocabulary with around 130 thousand entries. Including these tokenizers allows us to compare Serbian-specific approaches against cutting-edge general models, highlighting the benefits of morphology and script-aware designs while situating our results within the broader landscape of contemporary language modelling.

We also evaluate the newest Serbian-specific tokenizer (*ModernBertic*) introduced by Perovic (2026), which was trained from scratch using BPE trainer with a 50,304 entry vocabulary and on Latin script text only. The model authors report low fertility (1.59 tokens per word) and significant gains compared to multilingual baselines. While these external models have much higher vocabulary counts, our goal is to evaluate how our more compact tokenizers fare against them, highlighting the trade-offs between sheer vocabulary size and language-specific efficiency.

4. Evaluation & Results

For evaluation, we relied primarily on the *Tokenizer Analysis Suite* (Meister, 2025), a framework designed for systematic intrinsic benchmarking of tokenization strategies. However, given the scale of our Serbian corpus and the need for custom normalization, we had to

adapt the pipeline substantially. The original system processes data by tokenizing with all dictionaries simultaneously and storing results in a single dictionary object, which proved infeasible for a dataset of 300 million words. Instead, we serialized the pipeline: tokenizing with one tokenizer at a time, calculating metrics, and concatenating results downstream. In addition, we implemented custom pre-tokenization and decoding hooks by building new wrapper classes. These wrappers invoke different normalization functions depending on the tokenizer under evaluation, ensuring that byte encoding of added tokens, and casing and script compression were properly integrated into the analysis. Table 3 lists all the metrics used in the evaluation, which is structured around four categories.

Table 3: Tokenizer evaluation metrics by group.

Category	Metric	Definition / Formula
Basic Tokenization	Compression Rate	$CR = \frac{\text{Text size (bytes)}}{\text{Number of tokens}}$
	Fertility	$\text{Fertility} = \frac{\text{Tokens}}{\text{bytes}}$
	Token Length	Average token size in bytes
	Type-Token Ratio	$TTR = \frac{\text{Unique tokens}}{\text{Total tokens}}$
Information Theoretic	Vocab. Utilization	$VU = \frac{\text{Used vocab size}}{\text{Total vocab size}}$
	Rényi Entropy	$H_\alpha(X) = \frac{1}{1-\alpha} \log \sum_i p_i^\alpha$
Morphological	Boundary Precision/Recall	Alignment of tokens with morpheme boundaries
Fairness	Lorenz cost	$LC = \frac{\text{Sum of Frequencies up to Token } i}{\text{Total Sum of All Frequencies}}$

Basic tokenization metrics (compression rate, fertility, average token length, and type-token ratio) are interrelated quantifying, from different points of view, the level of text fragmentation. High compression rate means that token repetition is high. High token repetition is also expressed by low TTR. Higher fertility means that the evaluated tokenizer tends to fragment text more creating shorter tokens, whose length is directly assessed by the average token length. In principle, the goal of the tokenizer is to achieve a high rate of token repetition (high compression rate, low TTR) while producing relatively long tokens (low fertility, high token length). This allows an

LLM to fit longer sequences into a fixed context window, effectively reducing memory overhead and training/inference costs per sentence.

Information-theoretic metrics (Rényi entropy and vocabulary utilization) extend this analysis by providing insight into how information is distributed across tokens. Rényi entropy is a generalization of Shannon entropy for different values of the parameter α . For $\alpha = 1$, Rényi entropy is equivalent to Shannon entropy; other values of α express different upper bounds on the entropy of a given set of symbols. Intuitively, this measure shows how predictable symbols (tokens) are on average: the higher the value the less predictable tokens. Since the goal of tokenization is to compress text data, higher Rényi entropy values are desirable. However, note that high values can be achieved via larger vocabularies, but this goes against vocabulary utilization. Large vocabularies will tend to contain rarer items, which might not be seen in the held-out test. In this sense parts of a larger vocabulary will not be used. In the case of these measures too, the goal of the tokenizer is to balance these two opposite objectives, that is achieve high entropy but also high vocabulary utilization.

Morphological metrics assess to what degree obtained tokenization agrees with manual linguistic analysis of subword morphology. BPE-based tokenizers are known to produce subword segments that do not map well to linguistic analyses. While linguistic fidelity currently does not seem to play a crucial role in downstream behaviour of large models (Saleva and Lignos, 2021), it might turn out beneficial to some NLP tasks. The *MorphScore* framework (Arnett and Bergen, 2025) includes the measures of morphological boundary precision/recall and morpheme preservation with respect to a given manual analysis. Subwords that preserve morphemes enable the model to generalize better to unseen inflected word forms and retain semantic consistency across complex grammatical variations.

Finally, multilingual fairness was quantified using Lorenz curves to measure representation inequality across scripts. In unbalanced tokenizers, non-Latin scripts (such as Cyrillic) are often over-fragmented compared to Latin scripts. This fragmentation introduces severe bias: it inflates inference costs, increases latency, and effectively shrinks the available context window for users operating in non-Latin scripts. Evaluating fairness ensures the tokenizer processes both Latin and Cyrillic text with equivalent computational efficiency.

4.1. Results

Results are reported in two separate settings: for the first test the whole evaluation dataset was transliterated to Latin, and Latin-specific tokenizers were used (along with three external and four that use script compression), while for the second test, the evaluation dataset was left as is (containing both Serbian scripts) and the tokenizer trained exclusively on Latin script were replaced with their mixed script variants. The results of the first test are presented in Table 4 and the results of the second test are presented in Table 5. Exceptional results, either among all or only among the experimental tokenizers are marked bold.

Due to the size of evaluation set, all of type-token ratio scores gravitated towards zero, so we omit them here. Similarly, vocabulary utilization gravitated towards 100% for most of the tokenizers, but we include this data because of existing exceptions. Finally, because the data used to evaluate MorphScore is in Latin, we leave this metric out from the second table and replace the results with Lorenz total cost indicating the tokenizer fairness in treating the two scripts.

5. Discussion

5.1. Baseline

Monolingual GPT2-style BPE models (*bpe*, *bpe_mix*) proved to be a remarkable baseline, outperforming the external tokenizers on all basic tokenizer metrics (CR, fertility, token length and VU) on both test sets. Especially notable is the gap over ModernBertic (also trained on Serbian text) on Latin despite the smaller vocabulary. BPE also achieves high Rényi entropy and MorphScore results, outperforming most of the other models on both tests.

Trailing space BPE models (*ts_bpe*, *ts_bpe_mix*) achieved the best overall basic tokenization scores on Latin test as well as the lowest fertility on mixed script text, confirming our hypothesis that the trailing space fits better the Serbian language, which is also confirmed via higher morphoscore, although all improvements over baseline BPE are slight. Additionally it achieved lower Lorenz cost indicating more even distribution of performance over the two scripts. The only downside overall is slightly lower vocabulary usage than GPT2-style BPE models coupled with slightly lower entropy scores.

Table 4: Compared results of evaluated tokenization metrics for thirteen tokenizers for Latin text only.

Tokenizer	CR	Fertility	Token Length	VU
Qwen3.5	2.785	0.383	2.71	48.3%
Mistral3.5	2.828	0.378	2.75	69.8%
ModernBertic	3.710	0.322	3.61	99.8%
bpe	3.870	0.289	3.77	99.9%
ts_bpe	3.941	0.284	3.84	99.8%
mireBPE	3.308	0.324	3.22	98.0%
mireBPE_ctrl-1	3.298	0.327	3.21	98.9%
mireBPE_ctrl-2	3.297	0.327	3.21	98.5%
Srna	3.866	0.289	3.76	99.8%
SrnaCF	3.591	0.319	3.49	99.4%
MorfoTok_base	3.217	0.368	3.13	99.3%
MorfoTok	3.217	0.368	3.13	99.3%
MorfoTokCF	2.823	0.423	2.75	99.5%
Tokenizer	Rényi ($\alpha=1$)	Rényi ($\alpha=2.5$)	Boundary Recall	Boundary Precision
Qwen3.5	10.75	7.77	0.312	0.126
Mistral3.5	10.74	7.75	0.292	0.111
ModernBertic	12.09	6.95	0.781	0.761
bpe	11.97	6.99	0.712	0.689
ts_bpe	11.88	6.84	0.716	0.689
mireBPE	10.18	5.77	0.754	0.397
mireBPE_ctrl-1	10.26	5.82	0.758	0.399
mireBPE_ctrl-2	10.26	5.81	0.757	0.399
Srna	11.96	7.00	0.712	0.689
SrnaCF	11.45	6.76	0.668	0.644
MorfoTok_base	11.00	7.08	0.747	0.680
MorfoTok	10.98	6.99	0.747	0.680
MorfoTokCF	10.11	5.54	0.773	0.714

5.2. mireBPE tokenizers

Through the mutual comparison of the mireBPE models it is confirmed that the minimum redundancy point cut-off can be applied to Serbian, as it outperformed the control points (*ctrl-1* and *ctrl-2*) on both tests regarding basic tokenization metrics (the differences on the mixed script test show a much higher impact). Information

Table 5: Compared results of evaluated tokenization metrics for thirteen tokenizers for mixed Cyrillic and Latin text.

Tokenizer	CR	Fertility	Token Length	VU
Qwen3.5	3.597	0.313	2.49	54.8%
Mistral3.5	3.789	0.303	2.60	75.3%
ModernBertic	2.075	0.695	1.76	99.7%
bpe_mix	5.003	0.241	3.40	99.9%
ts_bpe_mix	5.093	0.237	3.46	99.8%
mireBPE_mix	3.092	0.355	2.24	95.4%
mireBPE_ctrl-1_mix	2.721	0.404	2.03	99.0%
mireBPE_ctrl-2_mix	2.033	0.541	1.52	98.6%
Srna	5.136	0.242	3.50	99.9%
SrnaCF	4.816	0.266	3.28	99.2%
MorfoTok_base_mix	1.810	0.751	1.52	99.2%
MorfoTok	4.436	0.303	2.99	99.3%
MorfoTokCF	3.935	0.344	2.65	99.5%
Tokenizer	Rényi ($\alpha=1$)	Rényi ($\alpha=2.5$)	Lorenz Cost	
Qwen3.5	11.23	8.07	0.583	
Mistral3.5	11.28	8.02	0.564	
ModernBertic	6.57	3.37	0.938	
bpe_mix	12.34	7.38	0.431	
ts_bpe_mix	12.26	7.20	0.425	
mireBPE_mix	9.13	4.35	0.611	
mireBPE_ctrl-1_mix	9.13	4.76	0.676	
mireBPE_ctrl-2_mix	7.66	3.24	0.899	
Srna	11.89	7.02	0.403	
SrnaCF	11.39	6.77	0.407	
MorfoTok_base_mix	6.37	3.20	1.037	
MorfoTok	10.93	6.96	0.476	
MorfoTokCF	10.09	5.53	0.542	

theoretic and MorphScore results are, however, not in line, showing best performance (higher Rényi entropy, vocabulary usage and MorphScore) on the first control point which was manually picked. By the second control point these scores tend to fall again reaching either somewhat better scores (*mireBPE_ctrl-2*) or somewhat lower scores (*mireBPE_ctrl-2_mix*) than the minimum redundancy point cut-off (*mireBPE* and *mireBPE_mix* respectively). On the other

hand, *mireBPE* and *mireBPE_mix* achieve slightly lower Lorenz cost over the control points in their test buckets.

However, all of the mutual differences are minimal when compared to the other models, with about 10-15% deterioration (from the baseline) on the Latin test and 40-60% on the mixed script test regarding all basic tokenization and information-theoretic metrics (CR, fertility, token length and Rényi entropy). On the other hand, it achieves some of the highest boundary recall scores (MorphScore) but some of the lowest boundary precision indicates it could be due to over-segmentation.

These findings combined indicate that the *merging phase* is correctly performed by selecting the minimum redundancy point (as evident via comparison against *ctrl-1* and *ctrl-2*), but the *concatenation phase* needs further methodology refinements to produce more optimal results (as evident via comparison with the baseline).

5.3. Srna tokenizers

Since these tokenizers are trained like GPT2-style BPE, and script compression is applied only on the mixed script text, theoretically, *Srna* tokenizer without case compression should match *bpe* results on the first test (confirmed, Table 4) with *SrnaCF* results providing the insight into the case compression strategy. They show (contrary to the hypothesis) that case compression negatively impacts the tokenizer performance on all intrinsic metrics, meaning that the benefits of extra vocabulary slots (provided through lowercasing) are outweighed by the extra capitalisation tags inserted into text.

The results of the second test (Table 4), provide insight in the possible impact of script compression, especially through the comparison of the BPE model trained on mixed script (*bpe_mix*) and BPE model trained on script compressed text (*Srna*). From these results, we see an improvement on the baseline regarding basic tokenization metrics. Higher compression rate and token length than both primary and secondary baseline (and all other tokenizers), with fertility being slightly lower than that of *ts_bpe_mix*. When looking into information-theoretic metrics, specifically Rényi entropy, we notice a slightly lower score on the experimental model, but what is interesting is that the entropy is same for *Srna* on both tests, while the entropy increased from *bpe* to *bpe_mix*, with the addition of Cyrillic tokens positively affecting the score. Another improvement against

the baseline can be seen in the lowest-of-all Lorenz cost indicating that script compression achieves the highest balance of Latin and Cyrillic performance across the board. Comparison between the results of *Srna* and *SrnaCF* indicate that the case compression result in the loss of performance across all metrics, just like in the first test.

It is implied from the results that this tokenizer could be further improved by using trailing space BPE instead of the GPT-2 style, in which case it would match the top performance (set by *ts_bpe*) on the Latin test and it is expected (based on reviewed results that it would extend that lead further in the mixed script testing. All in all, when used without the case compression, *Srna* tokenizer achieves top performance on most metrics, except the open case of Rényi Entropy and the MorphScore where it matches the baseline but lags slightly behind the morphology-aware tokenizer *MorfoTok*.

5.4. MorfoTok tokenizers

When comparing the results achieved by different MorfoTok tokenizers we can infer that the best model within the group is *MorfoTok*, achieving more or less the same results as *MorfoTok_base* on the Latin-only test, but outperforming both *MorfoTok_base_mix* and *MorfoTokCF* on the mixed-script test, reaffirming the advantage of script compression (*MorfoTok_base_mix* vs. *MorfoTok*), and disadvantage of case compression (*MorfoTok* vs. *MorfoTokCF*) under intrinsic tokenizer evaluation. The only exception being MorphScore results achieved by *MorfoTokCF*, which were the highest among all built tokens. However this tokenizer achieved some of the worse scores on base tokenization metrics in this group.

When comparing these tokenizers (primarily *MorfoTok*) against the baseline it is evident it somewhat underperforms on the basic tokenization metrics, scoring around 10-15% lower on CR, fertility and token length on both tests, 10-15% higher Lorenz cost and slightly lower VU. We also measure ~10% lower Shanon entropy (both tests) and Rényi entropy decrease on the mixed script. On the plus side *MorfoTokCF* achieves the best MorphScore overall (both boundary recall and precision), with *MorfoTok* just slightly behind.

5.5. Evaluation dataset size impact

When repeating the tests with a smaller evaluation dataset (30 thousand documents or 1% of the original) we came to the fol-

lowing conclusions: Regarding fertility, CR, and token length, the results are constantly worse for all tokenizers by a similar scaling factor ($\sim 10\%$) i.e. the bigger dataset provides a harsher challenge, but the comparative results hold up. VU differences are much higher on the smaller dataset, providing a better insight into this metric, but TTR still gravitates towards 0. The only metric effectively influenced by dataset size seems to be Rényi entropy, where models diverge into three groups: entropy decreasing with bigger dataset (all GPT2-style BPEs); entropy stagnating (*ts_bpe* and *Srna* models); and entropy increasing with bigger dataset (*mireBPE* and *MorfoTok*, i.e. heavily experimental models). The implication of these findings should be looked into with more detail in the future.

5.6. Conclusion

Across all experiments, GPT-2 style BPE tokenizers established themselves as a strong baseline, consistently outperforming external models on core tokenization metrics. The trailing-space BPE variants provided improvements, particularly in compression rate (due to the fact that Serbian is highly inflective language where the suffixes outnumber the prefixes), though at the cost of slightly lower vocabulary usage and entropy. *MireBPE* tokenizers demonstrated that minimum redundancy cut-off can be applied to Serbian, outperforming manually chosen control points in basic metrics, but their concatenation phase requires refinement, as overall performance lagged behind the baseline, especially on mixed-script data. With the elimination of some of the intermediate merges, this tokenizer creates room for most of the needed vocabulary, but fails at tokenizing less frequent words, falling back to bytes, as it does not have required character bigrams or trigrams in the vocabulary. *Srna* tokenizers confirmed that script compression is beneficial for Serbian, yielding superior compression rates, token lengths, and Lorenz cost compared to baselines, though case compression consistently degraded performance. *MorfoTok* tokenizers highlighted the trade-off between morphology-aware segmentation and intrinsic efficiency: although they underperformed in compression, fertility, and entropy, they achieved the highest MorphScore, particularly boundary recall and precision. Taken together, these findings suggest that while BPE remains the most robust general approach, script compression strate-

gies such as *Srna* offer promising improvements, and morphology-aware models like *MorfoTok* excel in linguistic boundary detection.

Although the strategies evaluated in this work were tailored to Serbian, their underlying design principles directly extend to two distinct language classes: dual-Script and digraphic languages and morphologically rich languages. The script-compression mechanism in *Srna* is universally applicable to any language operating under active digraphia or multi-script contexts. Key candidate languages include Hindustani (identical spoken vocabulary suffers from severe cross-script representation divergence), Central Asian Languages, namely Kazakh, Uzbek and Tajik (transitioning between Cyrillic, Latin, and Arabic orthographies) and Japanese. In all such cases, projecting orthographically distinct representations of identical underlying phonemes into a unified canonical space before subword vocabulary learning eliminates script bias and improves Lorenz fairness. Similarly, the core bottleneck addressed by *MorfoTok* is the trade-off between arbitrary subword segmentation and grammatical morpheme preservation—applies directly to: other Slavic languages: (e.g., Croatian, Bosnian, Bulgarian, Russian, Ukrainian), which share similar inflectional paradigms (case, gender, etc); and agglutinative Languages: (e.g., Turkish, Hungarian, Finnish, Estonian), where long chains of suffixes cause standard BPE to either over-fragment words or bloat the vocabulary with rare inflected forms.

Future work should refine concatenation in *MireBPE*, integrate trailing-space strategy into *Srna* and refine *MorfoTok* and case compression to increase fertility, while also looking into extrinsic/downstream evaluations through experimental model training.

Acknowledgements

This research was supported by the Science Fund of the Republic of Serbia, #7276, Text Embeddings - Serbian Language Applications - TESLA and the armasuisse S&T project BM-TOK.

Language Resources

Mistral AI. 2024. Mistral medium 3.5 128b. <https://huggingface.co/mistralai/Mistral-Medium-3.5-128B>. Accessed: 2026-05-27.

- Vuk Batanović, Nikola Ljubešić, Tanja Samardžić, and Tomaž Erjavec. 2023. [Serbian linguistic training corpus SETimes.SR 2.0](#). Slovenian language resource repository CLARIN.SI.
- Cvetana Krstev, Duško Vitas, Ivan Obradović, and Ranka Stanković. 2020. [Srpmd4tagging](https://live.european-language-grid.eu/catalogue/lcr/9294). <https://live.european-language-grid.eu/catalogue/lcr/9294>. European Language Grid resource.
- Mitar Perovic. 2026. [ModernBERTiC: A modern encoder for BCMS languages](#). Recrewty, EU-funded grant.
- Milutin Smiljanić. 2019. [srdict_chromium](https://github.com/msmiljan/srdict_chromium): Dictionaries for chromium. https://github.com/msmiljan/srdict_chromium. GitHub repository.
- Qwen Team. 2026. Qwen3. 5: Accelerating productivity with native multimodal agents, february 2026. *URL* <https://qwen.ai/blog>.
- Philipp Wasserscheidt. 2023. [Serbian web corpus PDRS 1.0](#). Slovenian language resource repository CLARIN.SI.
- Mihailo Škorić and Cosimo Palma. 2026. [Wiki dumps to training corpora: South slavic case](#).

References

- Orevaoghene Ahia, Sachin Kumar, Hila Gonen, Jungo Kasai, David Mortensen, Noah Smith, and Yulia Tsvetkov. 2023. [Do All Languages Cost the Same? Tokenization in the Era of Commercial Language Models](#). In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 9904–9923.
- Mehdi Ali, Michael Fromm, Klaudia Thellmann, Richard Rutmann, Max Lübbering, Johannes Leveling, Katrin Klug, Jan Ebert, Niclas Doll, Jasper Buschhoff, Charvi Jain, Alexander Weber, Lena Jurkschat, Hammam Abdelwahab, Chelsea John, Pedro Ortiz Suarez, Malte Ostendorff, Samuel Weinbach, Rafet Sifa, Stefan Kesselheim, and Nicolas Flores-Herr. 2024. [Tokenizer Choice For LLM Training: Negligible or Crucial?](#) In *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 3907–3924.
- Project Apertus. 2025. [Apertus: Democratizing open and compliant llms for global language environments](#).

- Catherine Arnett and Benjamin Bergen. 2025. [Why do Language Models Perform Worse for Morphologically Complex Languages?](#) In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 6607–6623.
- Duygu Ataman, Matteo Negri, Marco Turchi, and Marcello Federico. 2017. [Linguistically Motivated Vocabulary Reduction for Neural Machine Translation from Turkish to English](#). *The Prague Bulletin of Mathematical Linguistics*, 108:331–342.
- Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. [Unsupervised Cross-lingual Representation Learning at Scale](#). In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451.
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. [BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding](#). *CoRR*, abs/1810.04805.
- Dávid Držík and František Forgáč. 2024. [Slovak morphological tokenizer using the Byte-Pair Encoding algorithm](#). *PeerJ Computer Science*, 10:e2465.
- Ximena Gutierrez-Vasques, Christian Bentz, Olga Sozinova, and Tanja Samardžić. 2021. [From Characters to Words: the Turning Point of BPE Merges](#). In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 3454–3468.
- Matthias Huck, Simon Riess, and Alexander Fraser. 2017. [Target-side Word Segmentation Strategies for Neural Machine Translation](#). In *Proceedings of the Second Conference on Machine Translation*, pages 56–67, Copenhagen, Denmark. Association for Computational Linguistics.
- Vani Kanjirangat, Tanja Samardžić, Ljiljana Dolamic, and Fabio Rinaldi. 2025. [Tokenization and Representation Biases in Multilingual Models on Dialectal NLP Tasks](#). In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 23992–24010.
- Taku Kudo. 2018. [Subword Regularization: Improving Neural Network Translation Models with Multiple Subword Candidates](#). In

- Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 66–75.
- Davis Liang, Hila Gonen, Yuning Mao, Rui Hou, Naman Goyal, Marjan Ghazvininejad, Luke Zettlemoyer, and Madian Khabsa. 2023. [XLM-V: Overcoming the Vocabulary Bottleneck in Multilingual Masked Language Models](#). In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 13142–13152.
- Thomas F. Magner. 2001. [Digraphia in the Territories of the Croats and Serbs](#). *International Journal of the Sociology of Language*, 2001(150):11–26.
- Clara Meister. 2025. [TokEval: A Tokenizer Analysis Suite](#).
- Sabrina J. Mielke, Ryan Cotterell, Kyle Gorman, Brian Roark, and Jason Eisner. 2019. [What Kind of Language Is Hard to Language-Model?](#) In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4975–4989.
- Benjamin Minixhofer, Ivan Vulić, and Edoardo Maria Ponti. 2025. [Universal Cross-Tokenizer Distillation via Approximate Likelihood Matching](#). In *Advances in Neural Information Processing Systems*, volume 38, pages 79297–79326. Curran Associates, Inc.
- NLLB-Team. 2022. [No Language Left Behind: Scaling Human-Centered Machine Translation](#).
- Hyunji Hayley Park, Katherine J Zhang, Coleman Haley, Kenneth Steimel, Han Liu, and Lane Schwartz. 2021. [Morphology Matters: A Multilingual Language Modeling Analysis](#). *Transactions of the Association for Computational Linguistics*, 9:261–276.
- Sachin Pawar, Manoj Apte, Kshitij Jadhav, Girish Keshav Palshikar, and Nitin Ramrakhiyani. 2025. [Broken Words, Broken Performance: Effect of Tokenization on Performance of LLMs](#). In *Proceedings of the 14th International Joint Conference on Natural Language Processing and the 4th Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics*, pages 372–385.
- Aleksandar Petrov, Emanuele La Malfa, Philip Torr, and Adel Bibi. 2023. [Language Model Tokenizers Introduce Unfairness Between Languages](#). In *Thirty-seventh Conference on Neural Information Processing Systems*.

- Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. [Language Models are Unsupervised Multitask Learners](#). *OpenAI blog*, 1(8):9.
- 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](#). In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3118–3135.
- Jonne Saleva and Constantine Lignos. 2021. [The Effectiveness of Morphology-aware Segmentation in Low-Resource Neural Machine Translation](#). In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Student Research Workshop*, pages 164–174.
- 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.
- Mihailo Škorić and Nikola Janković. 2024. [New Textual Corpora for Serbian Language Modeling](#). *INFOTHECA: Faculty of Philology, University of Belgrade*, 24(1):71–96.
- Chaofan Tao, Qian Liu, Longxu Dou, Niklas Muennighoff, Zhongwei Wan, Ping Luo, Min Lin, and Ngai Wong. 2024. [Scaling Laws with Vocabulary: Larger Models Deserve Larger Vocabularies](#). In *Advances in Neural Information Processing Systems*, volume 37, pages 114147–114179.

Komparativna intrinzična evaluacija LLM tokenizatora na srpskom tekstu

Mihailo Škorić, Nikola Janković, Tanja Samardžić

Sažetak

Ovaj rad predstavlja uporednu intrinzičnu evaluaciju tokenizatora za velike jezičke modele, obučenih posebno za srpski jezik, koji karakteriše bogata morfologija i upotreba dva pisma. Konstruisali smo nekoliko tokenizatora koristeći različite strategije, uključujući standardno BPE u GPT2 stilu i BPE sa pratećim razmakom, kao i neke eksperimentalne pristupe: istraživanje minimalne redundantnosti, kompresiju pisam i kapitalizacije, i tokenizaciju potpomognutu morfologijom (na osnovu srpskog morfološkog rečnika). Pored toga, upoređujemo tokenizatore sa savremenim modelima kao što su Qwen3.5 i Mistral3.5. Obučavanje svih tokenizatora se sprovodi na srpskom korpusu od približno milijardu reči, a evaluacija se vrši na odvojenom, tri puta manjem skupu. Intrinzične osobine se analiziraju pomoću softvera Tokenizer Analysis Suite, izračunavajući metrike kao što su fertilitet, stepen kompresije, Renijeva entropija i morfoskor. Naši rezultati ističu sistematske razlike u rezultatima postignutim različitim strategijama izgradnje tokenizatora: pristupi vođeni morfologijom postižu veći morfskor, dok strategije kompresije postižu niži fertilitet i poboljšanu kompresiju, posebno pri radu sa mešovitim ćirilčno-latiničnim korpusima. U poređenju, postojeći tokenizatori opšte namene pokazuju da su konstantno po većini intrinzičnih metrika inferiorniji u odnosu na tokenizatore specijalizovane za jedan jezik, što naglašava važnost prilagođenog dizajna tokenizatora.

Ovaj rad pruža početnu uporednu studiju srpskih tokenizatora, nudeći uvid u to kako izbori strategija utiču na intrinzična svojstva. Sistematskim upoređivanjem postojećih i novoobučenih modela, postavljamo osnovu za buduća istraživanja razvoja tokenizatora za morfološki bogate jezike i doprinosimo benčmarkingu koji će pomoći u dizajniranju prilagođenih tokenizatora za specifične namene u budućnosti.

Ključne reči: Tokenizacija, Srpski, Intrinzična evaluacija, VJM

Works are licensed under the terms and conditions of the Creative Commons Attribution 4.0 International License.

<https://creativecommons.org/licenses/by/4.0/>

