BG Pulse Journal USA
ISSN:1947-2536 http://www.bgpulseusa.com/index.php/bg
BG PULSE JOURNAL

UDK 004.912:004.85

THE TOKENIZATION PREMIUM: MEASURING UZBEK TOKENIZATION OVERHEAD IN LARGE LANGUAGE MODELS AND EVALUATING THE 2026 ALPHABET REFORM

Xayrullo Ro'zimatov

Millat Umidi University

Doi: https://doi.org/10.5281/zenodo.21872526

Abstract: Large language models process text as tokens, making tokenization efficiency important for computational cost and context capacity. This study examines Uzbek tokenization efficiency relative to English and other languages and evaluates whether the proposed 2026 Uzbek alphabet reform can reduce the observed overhead. Six widely used tokenizers are compared using a multilingual parallel corpus and a large Uzbek literary corpus. Latin Uzbek requires substantially more tokens than English across the evaluated tokenizers, while Cyrillic Uzbek shows an even greater overhead in some cases. Turkish generally requires fewer tokens than Uzbek under the same tokenizer, suggesting that agglutinative morphology alone does not explain the difference. The proposed alphabet reform produces only modest and tokenizer-dependent reductions in token counts. These findings indicate that orthographic reform can improve tokenization efficiency to a limited extent, but the larger Uzbek tokenization overhead remains primarily a broader tokenizer and language-resource problem.

Keywords: Uzbek; tokenization; large language models; NLP; alphabet reform; low-resource languages; BPE; agglutinative languages.

1. Introduction

Large language models are everywhere now — writing text, translating, summarizing, helping people code. Users type in words; the model, though, doesn't see words. What it sees is a sequence of tokens the tokenizer has cut the input into. That step happens before anything else, and most people never notice it.

This matters in practical terms. APIs bill per token [1], and context windows are measured in tokens rather than words. Two sentences with the same meaning but different token counts are not equivalent to the model or to the user paying the invoice. Prior work has shown this varies substantially across languages [5, 6], with low-resource languages typically receiving the higher costs.

Uzbek is one of those languages, and worth looking at. It's the state language of Uzbekistan, spoken by tens of millions of people, and it's agglutinative — meaning is built up by sticking suffixes onto a root. Take *kitoblarimizdan*, "from our books." That single word contains a root (*kitob*), a plural (*-lar*), a first-person plural possessive (*-imiz*), and an ablative case (*-dan*). Four morphemes glued together. A morphology-aware tool would split them cleanly. A subword

tokenizer, on the other hand, is blind to morphology; it slices the string according to whatever byte-pair statistics its vocabulary happens to remember.

There is also an orthographic wrinkle. In current Latin Uzbek, some sounds are written as digraphs or as a letter plus an apostrophe: *o'*, *g'*, *sh*, *ch*, *ng*. On 7 July 2026, the Legislative Chamber of the Oliy Majlis passed a draft reform that would replace *o'* with ö, *g'* with ğ, *sh* with ş, and *ch* with ç — going from 26 letters plus 3 digraphs to 28 letters plus one apostrophe [12]. The bill has been forwarded to the Senate. The status of the *ng* digraph is not yet settled. One of the stated motivations for the reform is exactly the digitalization of the state language [12]. Which raises a very concrete computational question. Does the proposed new spelling actually change the number of tokens Uzbek text takes up?

This paper addresses four research questions:

- **RQ1.** What is the size of the Uzbek tokenization overhead relative to English in the evaluated corpora and tokenizer families?
- **RQ2.** How does Uzbek tokenization differ between Latin and Cyrillic scripts?
- **RQ3.** How does Uzbek compare with Turkish, another agglutinative Turkic language?
- **RQ4.** Does the proposed 2026 Uzbek Latin alphabet reform reduce tokenization overhead, and if so, by how much?

The contribution is threefold. The paper reports comparative measurements of Uzbek tokenization across six widely used tokenizer families and three corpora of increasing size — from a controlled 30-sentence parallel benchmark to a full novel. It uses Turkish as a linguistic comparison to examine whether agglutinative morphology alone can account for the observed differences. And it provides an empirical evaluation of the proposed July 2026 Uzbek alphabet reform as a tokenization intervention, reporting per-tokenizer reductions on three natural Uzbek corpora.

Prior work on Uzbek NLP is limited but growing, and it frames what this paper is and is not doing. Kuriyozov, Vilares and Gómez-Rodríguez [16] released BERTbek, a pretrained Uzbek language model, showing that language-specific pretraining produces measurable gains over multilingual models on Uzbek downstream tasks. Salaev [17] built UzMorphAnalyser, a morphological analyzer for Uzbek based on inflectional endings; his work confirms that the root-plus-suffix decomposition assumed in Section 2.2 of this paper is not a linguistic abstraction but a well-formed and computable operation on real Uzbek text. Safarov [19] applied Morfessor-style unsupervised morphological segmentation as a step toward hybrid Uzbek tokenization — providing a concrete example that morphology-aware tokenization is technically approachable for Uzbek, which supports the remediation options discussed in the Discussion. (The Safarov work is currently under review as a conference submission at MathAI 2026 and should be treated as a preprint rather than an established published study.) Recent work has also examined the linguistic quality of LLM-generated Uzbek text [18] and attributes part of the observed error patterns (suffixation errors, disharmonic vowels) to subword-tokenization mismatches with Uzbek morphology. Against this background, the present paper does something narrower and complementary: it does not build a new tokenizer or a new model, but measures how badly the *current* multilingual tokenizers actually handle Uzbek, and provides an empirical evaluation of how much the proposed July 2026 alphabet reform would move that number on real Uzbek text.

2. Background

2.1 Subword Tokenization

Modern language-model tokenizers use subword segmentation. Byte-Pair Encoding (BPE) [2] and related algorithms — the Unigram Language Model [3], SentencePiece [8] — construct a vocabulary by iteratively merging frequent adjacent byte or character pairs in a training corpus until a target vocabulary size is reached (typically 32,000–256,000 entries). Exact procedures differ across tokenizer families, but the property that matters here is the same: vocabularies are learned from data. Sequences that were frequent in the training corpus receive compact representations; rare or unseen sequences are split into smaller pieces, cascading down to individual bytes as a final fallback.

2.2 Uzbek Morphology and Orthography

Uzbek morphology allows long words assembled from a root and a chain of suffixes. A morphology-aware processor would ideally split *kitoblarimizdan* into four functional units, and existing Uzbek NLP tooling can perform this decomposition: Salaev's UzMorphAnalyser [17] reports over 91% word-level accuracy on stemming and morphological feature extraction using inflectional-ending rules. A statistical subword tokenizer, by contrast, is blind to morphology; trained without significant Uzbek data, it will approximate the word using whichever byte-pair merges its statistics recorded, without preserving the morpheme boundaries a linguistic analyzer would find.

Latin Uzbek orthography adds a further consideration. The sounds represented by *o'* and *g'* are written with a base vowel plus an apostrophe (typically U+0027, U+2019, or U+02BC in digital text). Because English text rarely places apostrophes in the middle of words, BPE merge chains typically do not include vowel–apostrophe pairs, and the apostrophe frequently ends up as its own token in the tokenizers evaluated here. The digraphs *sh* and *ch* may or may not receive dedicated tokens depending on the tokenizer.

2.3 The Proposed 2026 Alphabet Reform

According to the official presentation of the draft reform [12], the current representations *o'*, *g'*, *sh*, and *ch* create spelling and technical problems, including in the digitalization of Uzbek. The reform replaces them with dedicated Latin letters ö, ğ, ş, and ç respectively, on a "one sound, one letter" principle. From a computational perspective, replacing a two-character sequence with a single Unicode character does not automatically reduce token count. A Unicode character is not necessarily a single tokenizer unit; the effect depends on whether the new characters and their surrounding sequences are covered by the tokenizer's vocabulary. The outcome must be measured rather than assumed.

2.4 Turkish as a Comparison

Turkish is here for one reason: to block a predictable objection. If Uzbek costs more tokens than English, a reviewer will say — of course, Uzbek is agglutinative and English is not. Turkish is

also agglutinative and Turkic, structurally similar to Uzbek, but it has been in every major multilingual training corpus for years. If Turkish is cheap and Uzbek is expensive under the same tokenizer, morphology alone can't be the whole story. Turkish is a comparison, not a controlled variable — the two languages differ in vocabulary, orthography, and corpus representation as well as size — so any observed difference is evidence consistent with a role for tokenizer coverage, not proof of it.

3. Methodology

3.1 Corpora

Three corpora were used, each addressing a different aspect of the research questions:

1. **Parallel corpus (RQ1–RQ3).** Thirty semantically parallel sentences, distributed across five domains (everyday speech, technology, news, academic, legal), translated by the author into English, Latin Uzbek, Cyrillic Uzbek, Russian, and Turkish. Translations preserve meaning without paraphrase.
2. **Short story (RQ4).** Abdulla Qahhor's *Anor* (1936) [14], 1,161 words after basic cleaning. Dialogue-heavy literary prose.
3. **Novel (RQ4).** O'tkir Hoshimov's *Dunyoning ishlari* (1982) [15], 60,390 words after OCR extraction and cleaning. Long-form varied natural prose.

The 30-sentence corpus was chosen small deliberately. It has to be built by hand — every sentence translated across five languages with matched meaning — and thirty was the largest count I could translate carefully and still have the parallelism trustworthy. That corpus answers the cross-language questions (RQ1–RQ3). For the alphabet-reform question (RQ4) I did not need parallelism at all — one language is enough — so I could use large, natural Uzbek prose. *Anor* is a short-form Uzbek classic. *Dunyoning ishlari* is one of the most widely read modern Uzbek novels, and its 60,390 words give a substantially larger sample for estimating the reform's average effect. Two very different registers (dialogue-heavy short story, autobiographical novel) also let me see whether the reform behaves the same way on both.

3.2 Tokenizers

Six tokenizer families were chosen for coverage of the current production landscape rather than for any single research reason. Two OpenAI encodings, o200k_base and cl100k_base, represent widely used tokenizer configurations associated with GPT-4o- and GPT-4-era models. Llama-3 and Gemma represent two widely used open-weight model families. mBERT and XLM-R go back further and are still the standard multilingual encoders in academic pipelines, useful as a baseline against the newer models. Together the six span vocabulary sizes from 100,000 to 256,000 entries, providing a diverse comparison across tokenizer families and vocabulary configurations.

Table 1. Tokenizers evaluated. Vocabulary size is the size reported by the tokenizer implementation used in the experiment.

Tokenizer	Vocabulary	Associated models

Tokenizer	Vocabulary	Associated models
OpenAI o200k_base	200,000	GPT-4o, GPT-4o-mini
OpenAI cl100k_base	100,000	GPT-4, GPT-3.5-turbo
Llama-3	128,000	Llama 3 / 3.1 / 3.2
Gemma	256,000	Gemma family
mBERT	119,547	Multilingual BERT [11]
XLM-R	250,002	XLM-RoBERTa [10]

The tokenizer implementations used are as follows:

- OpenAI encodings (o200k_base, cl100k_base): tiktoken 0.13.0.
- Llama-3 and Gemma vocabularies: HuggingFace transformers 5.14.1, using the community-mirrored repositories *Xenova/llama-3-tokenizer* and *Xenova/gemma-tokenizer*.
- mBERT and XLM-R: the HuggingFace models *google-bert/bert-base-multilingual-cased* and *FacebookAI/xlm-roberta-base*.

Special tokens are excluded from all counts.

3.3 Metrics

- **Token count** — total tokens produced by the tokenizer for the text.
- **Token fertility (F)** — tokens divided by number of words, where a word is a whitespace-delimited unit after Unicode-preserving normalization of apostrophe variants.
- **Relative Token Ratio (RTR)** — target-language tokens divided by English tokens for parallel content.
- **Overhead** = (RTR − 1) × 100%.
- **Reduction** = (current-Latin tokens − revised-Latin tokens) / current-Latin tokens × 100%.

Example. If English requires 350 tokens and Latin Uzbek requires 763 for the same sentences, RTR = 2.18 and overhead = 118%.

3.4 Reform Transliteration

The reform experiment applies the following replacements to Latin Uzbek text, in order, after normalizing apostrophe variants (U+2019, U+02BC, U+2018, U+0060, U+00B4, U+02BB) to ASCII U+0027:

o' → ö, O' → Ö
g' → ğ, G' → Ğ
sh → ş, Sh → Ş, SH → Ş
ch → ç, Ch → Ç, CH → Ç

The digraph *ng* is retained: the draft reform does not replace it with a single Unicode character, and its final treatment is not yet settled [12]. No semantic, lexical, or word-boundary changes were introduced. The transliterated text was then tokenized under the same six tokenizer versions used for the current Latin text.

4. Results

4.1 The Uzbek Tokenization Overhead (RQ1–RQ3)

Table 2. Tokenization efficiency across the 30-sentence parallel corpus.

Tokenizer	Language	Tokens	F	RTR	Overhead
GPT-4o (o200k)	English	348	1.13	1.00×	—
	Uzbek (Latin)	763	2.85	**2.19**×	+119%
	Uzbek (Cyrillic)	814	3.04	**2.34**×	+134%
	Russian	551	2.03	1.58×	+58%
	Turkish	609	2.42	1.75×	+75%
GPT-4/3.5 (cl100k)	English	350	1.14	1.00×	—
	Uzbek (Latin)	979	3.65	**2.80**×	+180%
	Uzbek (Cyrillic)	1572	5.87	**4.49**×	+349%
	Russian	1048	3.85	2.99×	+199%
	Turkish	877	3.48	2.51×	+151%
Llama-3	English	350	1.14	1.00×	—
	Uzbek (Latin)	958	3.57	2.74×	+174%

Tokenizer	Language	Tokens	F	RTR	Overhead
	Uzbek (Cyrillic)	1233	4.60	3.52×	+252%
	Turkish	617	2.45	1.76×	+76%
Gemma	English	346	1.12	1.00×	—
	Uzbek (Latin)	885	3.30	2.56×	+156%
mBERT	English	399	1.30	1.00×	—
	Uzbek (Latin)	828	3.09	2.08×	+108%
XLM-R	English	421	1.37	1.00×	—
	Uzbek (Latin)	610	2.28	1.45×	+45%

In the 30-sentence evaluation, Latin Uzbek required between 1.45× (XLM-R) and 2.80× (cl100k_base) the English token count. Cyrillic Uzbek reached 4.49× under cl100k_base. The result establishes a substantial Uzbek tokenization overhead under the tokenizer settings tested.

XLM-R produced the lowest observed Latin Uzbek RTR in this corpus. XLM-R was trained on the CC-100 corpus, which was constructed to cover a large number of low-resource languages [10]; the result is consistent with the possibility that tokenizer training-data composition contributes to the observed differences.

4.2 Latin and Cyrillic Uzbek (RQ2)

Cyrillic Uzbek generally required more tokens than Latin Uzbek in the evaluated tokenizers. The largest gap appeared under cl100k_base (1,572 vs 979 tokens). Russian, also written in Cyrillic, had considerably lower RTR than either Uzbek variant under GPT-4o (1.58× for Russian vs 2.19× for Latin Uzbek and 2.34× for Cyrillic Uzbek). This pattern is consistent with the view that script alone does not determine tokenizer efficiency, and that vocabulary coverage of the specific language matters as well.

4.3 Uzbek and Turkish (RQ3)

Turkish required fewer tokens than Uzbek under the same tokenizer and the same parallel content. Because both are agglutinative Turkic languages, this observation is consistent with the interpretation that morphological structure alone is not sufficient to explain the full Uzbek

tokenization overhead, and that tokenizer coverage plays a role. This is an association, not a controlled causal identification.

Overhead is relatively stable across the five domains sampled (everyday, technology, news, academic, legal); academic prose shows the highest Latin Uzbek fertility (about 3.15 tokens/word under GPT-4o), everyday speech the lowest (2.77). With only six sentences per domain, this is a descriptive observation rather than a statistically established effect.

4.4 Alphabet Reform Across Corpus Scales (RQ4)

The reform experiment was run at three scales: the same 30-sentence corpus (Latin Uzbek only), the short story *Anor* [14] (1,161 words; character count drops from 8,621 to 8,265 after transliteration, a 4.13% shrink), and the novel *Dunyoning ishlari* [15] (60,390 words; character count drops from 445,939 to 428,709, a 3.86% shrink).

Table 3. Reform reductions across the three corpora (percentage change in token count; positive = fewer tokens after reform).

Tokenizer	30 sentences	*Anor* (1,161 words)	*Dunyoning ishlari* (60,390 words)
GPT-4o	−2.23%	+1.64%	+0.10%
GPT-4/3.5	+1.53%	+2.08%	+0.72%
Llama-3	+3.55%	+5.25%	+2.97%
Gemma	+2.26%	+4.67%	+2.53%
mBERT	+0.36%	+5.39%	+2.66%
XLM-R	−2.79%	+3.73%	+1.25%

Across the three scales, the reform's effect depends on both tokenizer and corpus. On the novel it ranges from +0.10% (GPT-4o) to +2.97% (Llama-3). On the short story, reductions are larger, from +1.64% to +5.39%. On the 30-sentence corpus the direction itself flips for two tokenizers: GPT-4o and XLM-R produce slightly more tokens after transliteration (−2.23% and −2.79%), while the other four produce slightly fewer.

At the character level the two literary corpora become 3.86% and 4.13% shorter, but the token reductions are between 0.10% and 5.39%. Character compactness does not translate proportionally into token compactness.

5. Discussion

5.1 Character Reduction Is Not Token Reduction

Character count and token count move together only loosely. Under reform, the two literary corpora shrink by 3.9% and 4.1% in written characters, but token counts move by anywhere from near zero to about 5%. Fewer letters on the page does not automatically translate into fewer tokens for the model. A tokenizer that already learned *lasht* as a single piece will keep encoding it compactly. Confronted with a new *ş*, it has no such shortcut and has to fall back to whatever fragments its vocabulary happens to contain. Policy discussion of the reform's computational benefit needs to keep the two apart — the reader sees one gain, the model sees another, and the two numbers can disagree.

5.2 Why the Effect Varies Across Tokenizers

GPT-4o barely reacts to the reform on the novel-scale corpus (+0.10%). A reasonable reading is that its 200,000-entry vocabulary already carries reasonably compact segmentations for many of the affected Uzbek sequences, so there is little slack left for orthographic normalization to remove. Llama-3, Gemma, and mBERT show larger reductions on the same text — 2.5–3% — consistent with tokenizers whose segmentations are more sensitive to how *o'*, *g'*, *sh*, and *ch* appear on the page. These are inferences from segmentation behavior, not measurements of training data: the training corpora underlying these tokenizers are not publicly available, and this study makes no direct claim about how much Uzbek text each one has actually seen.

5.3 Why the Effect Varies Across Corpora

The three scales do not report the same result, and this divergence is particularly informative. On the 30-sentence corpus, two tokenizers actually produce more tokens under reform. On the two literary corpora, all tokenizers produce fewer. The alphabet did not change — the text did. A plausible explanation is that the literary corpora contain a different frequency distribution of the sequences the reform touches (*o'*, *g'*, *sh*, *ch*), and that dialogue and inflected forms in particular increase the reform's benefit. The 30-sentence corpus was deliberately built with a wide domain spread, including technical words like *kompyuterlashtirilgan* where the existing merge *lasht* is broken by *sh* → *ş* and the reform therefore costs a token. A direct test would be to count the occurrences of the affected sequences in each corpus and correlate them with the observed reduction; this study did not perform that measurement, and it is a natural next step.

5.4 Case Study: Word-Level Behavior

Table 4. Three representative Uzbek words tokenized under GPT-4o o200k, before and after reform. The three cases cover the full range of reform outcomes.

Word (current)	Tokens (current)	Word (revised)	Tokens (revised)	Outcome
shahar	2: [sh, ahar]	şahar	2: [ş, ahar]	unchanged
o'zgartirmoqda	7: [o, ', zg, art, irm, oq, da]	özgartirmoqda	**5**: [öz, gart, irm, oq, da]	reform helps

Word (current)	Tokens (current)	Word (revised)	Tokens (revised)	Outcome
kompyuterlashtirilgan	6: [kom, py, uter, lasht, iril, gan]	kompyuterlaştirilgan	**7**: [kom, py, uter, la, şt, iril, gan]	reform hurts

The three examples illustrate the full range of possible reform effects. *shahar* is unchanged: two tokens before, two after, with different bytes inside. *o'zgartirmoqda* is a clear reduction — 7 down to 5 — because the revised *öz* prefix already exists in GPT-4o's vocabulary as a compact unit. *kompyuterlashtirilgan* is the case where the reform costs a token: the existing BPE merge *lasht* is broken by *sh* → *ş*, the tokenizer has no matching merge for the new *laşt*, and the word grows by one token. The same paragraph of Uzbek text can contain all three outcomes at once, depending on which merges the tokenizer happens to have learned.

6. Implications

Token counts turn straight into bills. At GPT-4o's rate of USD 2.50 per million input tokens [1], one million English input tokens cost USD 2.50; the same content in Latin Uzbek, at RTR = 2.19, is about 2.19 million tokens and roughly USD 5.48 — more than double for the same meaning. Real costs depend on prompt structure, output length, caching, and workload mix, but the mechanism is straightforward.

The same overhead shrinks the effective context window: at a Llama-3 Uzbek fertility of 3.57, a 128,000-token window fits about 35,900 Uzbek words, compared with roughly 113,000 English words at English fertility — a corpus-derived approximation, not a guarantee. Longer input sequences also drive up compute through the quadratic scaling of self-attention [9], though the end-to-end effect on inference depends heavily on implementation and caching.

The reform reduces the API-cost figure only modestly: 0.10% on GPT-4o and 2.97% on Llama-3 on the novel, small next to the base overhead itself.

7. Limitations

The 30-sentence corpus is a small controlled sample; it supports a clear cross-language comparison but is not representative of Uzbek as a whole. Because the parallel benchmark was translated by a single author, translation choices may introduce language-specific lexical or syntactic differences that affect token counts. A follow-up should use a much larger multilingual parallel benchmark, ideally with independent translators, and report uncertainty estimates. The two literary corpora each come from one author and one register (Qahhor 1936 dialogue-heavy short prose; Hoshimov autobiographical novel), and reform reductions on other text types —

contemporary news, technical documentation, social media, spoken transcripts — are not measured here.

The study measures tokenization efficiency, not model quality. Fewer tokens is not automatically better if it comes at the cost of downstream performance; future work should pair token counts with perplexity or task-specific evaluation.

The reform transliteration follows the mapping in the Legislative Chamber's own presentation [12]; the *ng* digraph was left unchanged because its final treatment is not yet settled [12], and the adopted text may differ from what is tested here. Statements about tokenizer coverage of Uzbek are also inferences from segmentation behavior — the training corpora underlying the six tokenizers are not publicly available for direct measurement.

Corpus files, scripts, and raw per-tokenizer results are available at *https://github.com/kruzimatov/uzbek-tokenization-premium* for verification and re-analysis.

8. Conclusion

This study measured Uzbek tokenization in six widely used tokenizers across three corpus scales, with English as the baseline and Turkish and Russian as comparative reference languages, and evaluated the proposed 2026 Uzbek Latin alphabet reform as a tokenization intervention.

The primary result is a substantial Uzbek tokenization overhead relative to English in every tokenizer tested. Latin Uzbek ranges from 1.45× to 2.80× the English token count in the 30-sentence corpus; Cyrillic Uzbek reaches 4.49× under cl100k_base. The overhead is relatively stable across the five evaluated domains.

A second observation concerns the source of that overhead. It cannot be reduced to agglutinative morphology, because Turkish — agglutinative like Uzbek but much better represented in mainstream multilingual training corpora — costs consistently less under the same tokenizers. Script does not explain it either: Russian in Cyrillic is more compact than Uzbek in either script. Together these observations are consistent with a role for tokenizer vocabulary coverage in addition to linguistic structure.

The reform question yields a more nuanced answer. On the evaluated Uzbek texts, replacing *o'*, *g'*, *sh*, *ch* with ö, ğ, ş, ç does reduce token counts, but modestly and unevenly. In the 60,390-word novel, reductions range from just 0.10% under GPT-4o to 2.97% under Llama-3. Character count decreases by 3.9% in the same text — so a reader looking at the pages sees a real compression, while the tokenizer barely notices. Orthographic compactness and tokenization efficiency are related, but they are not the same property. The reform addresses one component of the tokenization overhead; it does not eliminate it.

For Uzbek language technology, the practical implication is that orthographic reform alone is unlikely to close the gap with high-resource languages. Direct tokenizer interventions remain the more direct route: vocabulary extension with Uzbek morphemes [7]; morphology-aware pre-tokenization along the lines already prototyped by Safarov's Morfessor-based Uzbek segmenter [19], or built on top of Salaev's morphological analyzer [17]; or training an Uzbek-focused tokenizer on a curated corpus, in the spirit of the BERTbek pretraining effort [16]. Each of these

approaches directly addresses vocabulary coverage — the factor most consistent with the observed overhead — rather than the surface representation of the text. The 2026 reform is a useful and cheap partial optimization within a larger infrastructure question, aligned with the broader national goals set out in the "Digital Uzbekistan 2030" strategy [13].

References

1. OpenAI. API Pricing. — https://openai.com/api/pricing/ (accessed August 2026).
2. Sennrich R., Haddow B., Birch A. Neural Machine Translation of Rare Words with Subword Units // Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL). — Berlin, 2016. — Pp. 1715–1725. arXiv:1508.07909.
3. Kudo T. Subword Regularization: Improving Neural Network Translation Models with Multiple Subword Candidates // Proceedings of the 56th Annual Meeting of the ACL. — Melbourne, 2018. — Pp. 66–75. arXiv:1804.10959.
4. Common Crawl Foundation. Statistics of Common Crawl Monthly Archives — Language Distribution. — https://commoncrawl.github.io/cc-crawl-statistics/plots/languages.html (accessed August 2026).
5. Petrov A., La Malfa E., Torr P., Bibi A. Language Model Tokenizers Introduce Unfairness Between Languages // Advances in Neural Information Processing Systems (NeurIPS). — New Orleans, 2023. arXiv:2305.15425.
6. Ahia O., Kumar S., Gonen H., Kasai J., Mortensen D. R., Smith N. A., Tsvetkov Y. Do All Languages Cost the Same? Tokenization in the Era of Commercial Language Models // Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing (EMNLP). — Singapore, 2023. — Pp. 9904–9923. arXiv:2305.13707.
7. Rust P., Pfeiffer J., Vulić I., Ruder S., Gurevych I. How Good is Your Tokenizer? On the Monolingual Performance of Multilingual Language Models // Proceedings of the 59th Annual Meeting of the ACL. — 2021. — Pp. 3118–3135. arXiv:2012.15613.
8. Kudo T., Richardson J. SentencePiece: A Simple and Language Independent Subword Tokenizer and Detokenizer for Neural Text Processing // Proceedings of the 2018 EMNLP: System Demonstrations. — Brussels, 2018. — Pp. 66–71. arXiv:1808.06226.
9. Vaswani A., Shazeer N., Parmar N., Uszkoreit J., Jones L., Gomez A. N., Kaiser L., Polosukhin I. Attention Is All You Need // Advances in Neural Information Processing Systems (NeurIPS). — Long Beach, 2017. — Pp. 5998–6008. arXiv:1706.03762.
10. Conneau A., Khandelwal K., Goyal N., Chaudhary V., Wenzek G., Guzmán F., Grave E., Ott M., Zettlemoyer L., Stoyanov V. Unsupervised Cross-lingual Representation Learning at Scale // Proceedings of the 58th Annual Meeting of the ACL. — 2020. — Pp. 8440–8451. arXiv:1911.02116.

11. Devlin J., Chang M.-W., Lee K., Toutanova K. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding // Proceedings of NAACL-HLT. — Minneapolis, 2019. — Pp. 4171–4186. arXiv:1810.04805.

12. Gazeta.uz. Deputies pass law to modify Uzbek alphabet. — 9 July 2026. — https://www.gazeta.uz/en/2026/07/09/alphabet/ (accessed August 2026).

13. Presidential Decree of the Republic of Uzbekistan No. UP-6079 (PF-6079) “On the Approval of the Strategy ‘Digital Uzbekistan — 2030’ and Measures for its Effective Implementation.” — Tashkent, 5 October 2020.

14. Qahhor A. Anor (hikoya). — First published 1936. Text used in this study obtained from the ziyouz.uz digital library of Uzbek classical literature.

15. Hoshimov O’. Dunyoning ishlari (qissa va hikoyalar). — First pieces published 1981; complete collected edition Toshkent: Sharq nashriyoti, 2005. Text used in this study obtained from a scanned edition (60,390 words after OCR extraction and cleaning).

16. Kuriyozov E., Vilares D., Gómez-Rodríguez C. BERTbek: A Pretrained Language Model for Uzbek // Proceedings of SIGUL 2024. — 2024.

17. Salaev U. UzMorphAnalyser: A Morphological Analysis Model for the Uzbek Language Using Inflectional Endings. — arXiv:2405.14179, 2024.

18. Investigating linguistic errors in large language model generation of Uzbek text // Cogent Arts & Humanities. — Taylor & Francis, 2025. DOI: 10.1080/23311983.2025.2600519.

19. Safarov R. Unsupervised Morphological Segmentation for Hybrid Tokenization in Uzbek Language. — Under review as a conference paper at MathAI 2026, OpenReview, 2026. https://openreview.net/forum?id=cwUDQn6jt2.