Article

Language-Specific Tokenization for Assamese: Efficiency and Downstream Integration with LLMs

Basab Nath [1,*] and Sagar Tamang [2]

[1] School of Computer Science Engineering and Technology, Bennett University, Greater Noida, Uttar Pradesh 201310, India

[2] Department of Computer Applications, Indian Institute of Technology Patna, Bihta Campus, Patna, Bihar 801106, India

* Correspondence: basabnath@gmail.com (B.N)

Abstract

Tokenization is a fundamental step in NLP, influencing both computational efficiency and downstream task performance. While subword methods such as Byte-Pair Encoding (BPE), WordPiece, and Unigram have shown strong results for high-resource languages, their suitability for low-resource and morphologically rich languages like Assamese remains insufficiently understood. This study presents a systematic evaluation of these tokenizers on a curated Assamese Wikipedia corpus, examining intrinsic efficiency metrics—including subword fertility, compression ratio, tokenization speed, and token diversity—alongside statistical validation and energy trade-offs. We further connect intrinsic behaviour to practical outcomes by fine-tuning IndicBERT and mBERT on sentiment and hate-speech tasks, and by assessing morphological boundary preservation. Results show that BPE-32K provides the most compact and semantically coherent segmentation, improves downstream F1 scores by 3–4 points, and preserves morpheme boundaries in 82% of cases, while WordPiece-16K offers the fastest tokenization speed. Overall, the findings demonstrate that vocabulary scaling reduces over-segmentation and that language-specific tokenizers substantially outperform multilingual defaults for Assamese. This work provides empirical guidelines for selecting tokenizers tailored to low-resource Indic languages and for integrating them effectively into LLM pipelines.

Keywords: Tokenization; Assamese; Low-Resource NLP, IndicBERT, mBERT

Citation: Nath B., Tamang S. Language-Specific Tokenization for Assamese: Efficiency and Downstream Integration with LLMs. *Impact in Computics*. 2025, 1, 4. https://doi.org/10.65500/computics-2025-004

Received: 27 September 2025 | Revised: 03 November 2025 | Accepted: 17 November 2025 | Published: 08 December 2025

1. Introduction

Tokenization is a fundamental component of modern Natural Language Processing (NLP) pipelines, directly shaping the quality and efficiency of downstream tasks such as sentiment analysis, machine translation, and named entity recognition [1], [2]. Subword tokenization approaches, including Byte-Pair Encoding (BPE), Word Piece, and Unigram models, have become widely adopted in high-resource languages due to their ability to handle vocabulary size and rare word problems effectively [3], [4]. However, their performance and suitability for low-resource languages, particularly those with complex morphology such as Assamese, have received limited systematic attention. Assamese, an Indo-Aryan language

spoken by over 15 million people, exhibits a rich morphological structure, variable dialects, and frequent code-mixing, which together present substantial challenges for tokenization. Earlier works on Assamese NLP have largely focused on surface-level token metrics, such as token counts or normalized sequence length [5], [6] without investigating broader efficiency measures or downstream impacts. Therefore, the choice of tokenizers for Assamese remains ad hoc and lacks empirical evidence, hindering progress on more advanced Assamese NLP applications. Despite these efforts, no existing study provides a unified evaluation of multiple tokenizer families for Assamese using both intrinsic efficiency metrics and downstream LLM performance, nor do they examine how vocabulary size interacts with the language's morphological complexity.

The paper focuses on this gap by presenting a systematic assessment of Assamese tokenizers across core dimensions of efficiency and linguistic fidelity. First, we go beyond the usual metrics and apply measures such as subword fertility, compression ratio, and tokenization speed. Second, our focus is on comparing Assamese tokenizers using a comprehensive set of efficiency-oriented metrics to support informed decisions for low-resource, morphologically rich languages. To the best of our knowledge, this is the first comprehensive evaluation of subword tokenizers for the Assamese language. The key contributions of this paper are as follows:

- We curate and preprocess a high-quality Assamese Wikipedia sentence corpus consisting of over 45,000 sentences and 500,000+ words, which serves as a benchmark for training and evaluating Assamese tokenizers.
- We implement and train nine tokenizer configurations across three popular subword algorithms—Byte Pair Encoding (BPE), WordPiece, and Unigram—each evaluated at vocabulary sizes of 8K, 16K, and 32K.
- We propose a diverse evaluation framework that goes beyond basic token counts by incorporating metrics such as Subword Fertility, Normalized Sequence Length, Compression Ratio, Token Diversity, and Tokenization Speed.
- We provide practical recommendations for choosing or designing tokenizers suited to morphologically rich, low-resource languages, informed by both performance trade-offs and linguistic fidelity.

The remainder of this paper is organized as follows. Section II discusses related work. Section III outlines our methodology, including the proposed metrics and evaluation framework. Section IV presents the experimental setup. Section V outlines the results. Section VI discusses key findings, and Section VII concludes with future directions

2. Related work

Tokenization is considered an important task of preprocessing in NLP, which has a great impact on the performance and efficiency of models [1], [2]. Subword tokenization techniques, such as Byte-Pair Encoding (BPE) [1], WordPiece [4], and Unigram models [3], have been successful in high-resource languages by mitigating the problems of large vocabulary size and out-of-vocabulary rare words. These methods tokenize text into subword units that result in lower out-of-vocabulary rates without blowing up the sequences. If we look at the Indian languages, tokenization has additional challenges to cope with rich morphology, agglutination morphology, and also code-mixing. Tokenization and subword style modeling have been applied in languages such as Hindi, Marathi, and Tamil [7], [8], but Assamese is still insufficiently explored. Work on tokenization for the Assamese language. There are very few academic studies that have described about tokenization of the Assamese language [16]. Ghosh and Senapati [9] observed limitations of tokenizers for Assamese hate speech, due to ambiguity in token boundaries caused by script variegation, code switching, etc.

Some efforts have examined structural tokenization metrics, for example, token count or normalized sequence length [5], [13], [14], [15], but more comprehensive efficiency and task-based assessments are lacking. Furthermore, comprehensive efficiency benchmarks of tokenizers for Assamese are virtually absent. Moreover, previous studies rarely evaluate tokenizers in downstream settings, leaving open the question of how segmentation quality influences model performance in practical Assamese NLP tasks.

Recent works have highlighted the importance of evaluating tokenizers not only by their compression characteristics but also by their practical impact on downstream models [10], [11], [12]. However, these insights have rarely been extended to low-resource Indian languages. For low-resource Indic languages, including Assamese, systematic comparisons across multiple algorithms and vocabulary sizes remain largely absent. This motivates our study, which aims to fill this research gap by systematically benchmarking Assamese tokenizers through both metric-oriented and task-based perspectives

3. Materials and Methods

In this section, we describe the full experimental pipeline developed for assessing the quality of subword tokenizers for Assamese. Our goal was to instrumentally evaluate how various algorithm choices (BPE, WordPiece, Unigram) and vocabulary size (8K, 16K, 32K) influence linguistic segmentation, compression, and processing speed in low-resource NLP. This approach involves (i) curated data preparation, (ii) training a tokenization model on multiple designs and configurations, and (iii) an extensive evaluation framework that takes into consideration seven metrics. Our approach is characterized by empirical soundness and relevancy for morphologically rich languages. Figure 1 illustrates the overall workflow followed in this study, including corpus preparation, tokenizer training across three subword algorithms, intrinsic evaluation using seven efficiency metrics, and downstream assessment with IndicBERT and mBERT.

Our methodological framework was designed to support the following goals:

- Enable direct comparison of subword tokenizers under controlled vocabulary and training conditions.
- Capture both linguistic granularity and computational efficiency using a multi-metric evaluation suite.
- Ensure reproducibility and fairness by using identical preprocessing, configuration, and analysis pipelines.

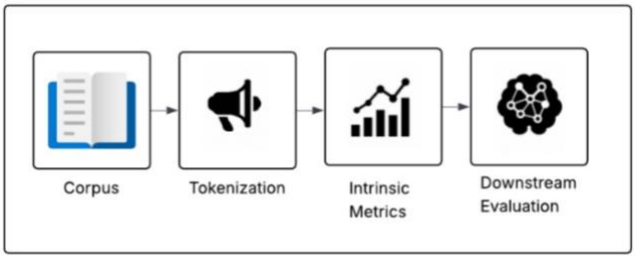


Figure 1. Overview of the tokenizer evaluation workflow.

3.1 Tokenizer Architectures and Training

We considered three well-known subword tokenization methods: BPE (Byte Pair Encoding) [1], WordPiece [4], and the Unigram Language Model [3]. These are the main families of subword models in mainstream pretrained LMs, and accordingly, they provide different trade-offs between segmentation granularity and probabilistic modeling.

We trained each tokenizer on the Assamese Wikipedia corpus using Hugging Face's tokenizers library. Effective Vocabulary Sizes To establish realistic low-resource language setups, we train each tokenizer variant on three different vocabulary sizes: 8K, 16K, and 32K. These sizes were selected to represent different stages of list fragmentation and compression so that composition at the subword level could be analyzed at several scales. The tokenizers were initialized with the same set of special tokens [UNK], [PAD], [CLS], [SEP], and [MASK].

In all cases, we configured a Whitespace pre-tokenizer to maintain consistency for sentence-level segmentation. No case folding, stemming, or morphological normalization was employed to maintain linguistic fidelity.

3.2 Evaluation Design

To describe tokenizer properties beyond raw token count, we adopted a three-dimensional evaluation framework that includes compression efficiency, fragmentation analysis, and operational speed. Both tokenizers were used to tokenize the same sentence-level Assamese Wikipedia corpus, and the tokenized outputs were evaluated using seven key metrics in Table 1.

Every metric emphasizes a different aspect of tokenization performance. Metrics that are directly or closely related to the implement-specific tokenization process, such as Compression Ratio and Average Token Length (which gauge how well, in terms of compression of text, the tokenizer performs), contrast with others that measure over-segmentizes, subword Fertility, and Normalized Sequence Length. Token Diversity provides some insight of generalization at the lexical level, and Tokenization Speed measures computational efficiency. All in all, these measures complement each other and give a holistic view of tokenization quality for downstream tasks in low-resource languages. Then we combined them with the three different tokenizer algorithms (BPE, WordPiece, Unigram) and vocabulary sizes (8K, 16K, 32K), which makes a total of nine different variations for the BERT model. This factorial design enabled us to control the effect of vocabulary size across models and compare the manner in which different algorithms deal with morphological variation in Assamese. Assamese poses a number of difficult challenges because it is an agglutinative language, objectivizing the verb complex, which makes tokenization more challenging and orthographically very complex, owing to its position as an International Language whose spelling system has yet to be standardized. We optimized for empirical rigor, reproducibility, and metrics diversity before tokenization choices are made for subsequent Indic NLP experiments.

4. Experimental Results

This section describes the experimental environment, data preprocessing pipeline, tokenizer training configuration, and evaluation procedures used to implement the methodology described in Section 3. All experiments were conducted in a consistent and reproducible cloud-based setting. All training and evaluation processes were performed on the default Google Collaboratory (Colab) free-tier environment, as available in 2025. The system included an Intel Xeon CPU and 12 GB of RAM, running an Ubuntu-based Linux OS. We used Python 3.10+ and implemented all tokenizer models using the Hugging Face tokenizers library. Additional libraries included pandas for data manipulation and matplotlib/seaborn for visualization and analysis.

Table 1. Tokenizer evaluation metrics used in this study

Metric	Description
Average Tokens per Sentence	Measured the average number of tokens produced per sentence. Lower values indicated greater compression and fewer splits.
Normalized Sequence Length (NSL)	Ratio of total tokens to total characters, normalized to account for sentence length variations. Served as a proxy for token efficiency.
Subword Fertility	Average number of subwords per word, indicating the degree of word fragmentation. Lower fertility implied better morpheme preservation.
Compression Ratio (CR)	Ratio of total input characters to the number of produced tokens. Higher ratios implied more compact tokenization.
Average Token Length	Measured the average character length of tokens. Longer tokens corresponded to more semantically complete units.
Token Diversity	Ratio of unique tokens to total tokens. Reflected vocabulary richness and generalization potential.
Tokenization Speed	Time taken (in seconds) to tokenize the full corpus. Captured practical inference latency, useful for real-time applications.

4.1 Dataset Description

For both training and evaluation, we have used Assamese Wikipedia corpus. We extracted Assamese Wikipedia to show our representations cover a wide variety of real-world linguistic phenomena. The dataset was selected because it also spans different styles of contemporary Assamese text, including formal (and semi-formal) and domain-specific regions of language use.

The raw Wikipedia dump was processed in a stepwise manner. First, entire article texts were extracted with a custom parser, excluding non-text content such as markup tables and metadata. We then used a rule-based sentence segmentation that utilized Assamese-specific punctuation patterns such as Assamese full stop (transliterated as "da¯ri") and question mark (?), and the Assamese "da¯ri" one sign. We only did a little preprocessing to keep it as close to the original Assamese text. Non-textual items such as HTML tags, info boxes, and broken lines were ignored, but no token stemming/lemmatizing/lowercasing was applied [21]. We kept all the Assamese Unicode characters along with genuine punctuation markers for realistic input settings while training the tokenizer.

The original Assamese Wikipedia dump contained approximately 175,000–185,000 cleaned sentences encompassing a wide variety of syntactic structures and morphological patterns. For our experiments, we curated a representative, high-quality subset of 45,832 sentences, covering diverse topics such as history, science, and geography. A sample of this corpus is illustrated in Figure 2. Table 2 provides detailed statistics of the final dataset used for tokenizer training and evaluation. The dataset, originally compiled by Tamang et al. [5], is publicly available at *https://github.com/indian-nlp/assamese-dataset*.

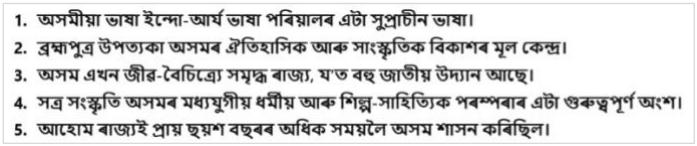

1. অসমীয়া ভাষা ইন্দো-আর্য ভাষা পৰিয়ালৰ এটা সুপ্ৰাচীন ভাষা।
2. ব্ৰহ্মপুত্ৰ উপত্যকা অসমৰ ঐতিহাসিক আৰু সাংস্কৃতিক বিকাশৰ মূল কেন্দ্ৰ।
3. অসম এখন জীৱ-বৈচিত্ৰ্যে সমৃদ্ধ ৰাজ্য, য'ত বহু জাতীয় উদ্যান আছে।
4. সত্ৰ সংস্কৃতি অসমৰ মধ্যযুগীয় ধৰ্মীয় আৰু শিল্প-সাহিত্যিক পৰম্পৰাৰ এটা গুৰুত্বপূৰ্ণ অংশ।
5. আহোম ৰাজ্যই প্ৰায় ছয়শ বছৰৰ অধিক সময়লৈ অসম শাসন কৰিছিল।

Figure 2. Sample Assamese sentences from the Wikipedia corpus.

4.2 Tokenizer Training and Evaluation

Each tokenizer was trained and evaluated using a consistent pipeline to ensure comparability across algorithm types and vocabulary sizes. We experimented with three widely adopted subword segmentation algorithms—Byte Pair Encoding (BPE), WordPiece, and the Unigram Language Model—each configured with vocabulary sizes of 8,000, 16,000, and 32,000. This resulted in a total of nine tokenizer variants. Training was performed using the Hugging Face tokenizers library. All models incorporated a consistent set of special tokens: [UNK], [PAD], [CLS], [SEP], and [MASK]. The Whitespace pre-tokenizer was applied to ensure uniform word-level segmentation across configurations. All tokenizers were trained on the Assamese Wikipedia corpus described in Section 4.1, using identical hyperparameters and randomized seed settings for fairness.

Following training, each tokenizer was applied to the full evaluation corpus. We computed the seven-core metrics defined in Section 3, including Average Tokens per Sentence, subword Fertility, Compression Ratio, and Token.

Table 2. Summary statistics of the Assamese Wikipedia dataset used for tokenizer training and evaluation.

Statistic	Value
Total Sentences	45,832
Total Words	564,710
Total Characters	3,814,266
Average Sentence Length (words)	12.32
Average Sentence Length (characters)	83.26
Average Word Length (characters)	6.75
Unique Word Types (Vocabulary)	84,137
Script	Eastern Nagari (Unicode Assamese)
Domain Coverage	General encyclopedic content (history, science, geography, etc.)
Source	Assamese Wikipedia dump (2024)
License	CC-BY-SA 3.0

5. Results

We evaluated nine tokenizer configurations: BPE, Word-Piece, and Unigram, each trained with vocabulary sizes of 8K, 16K, and 32K. The results are presented in four parts:

(i) intrinsic tokenizer evaluation across seven metrics, (ii) comparative analysis with related work, (iii) robustness and efficiency analysis, and (iv) downstream and morphological evaluation linking tokenization quality to LLM applications.

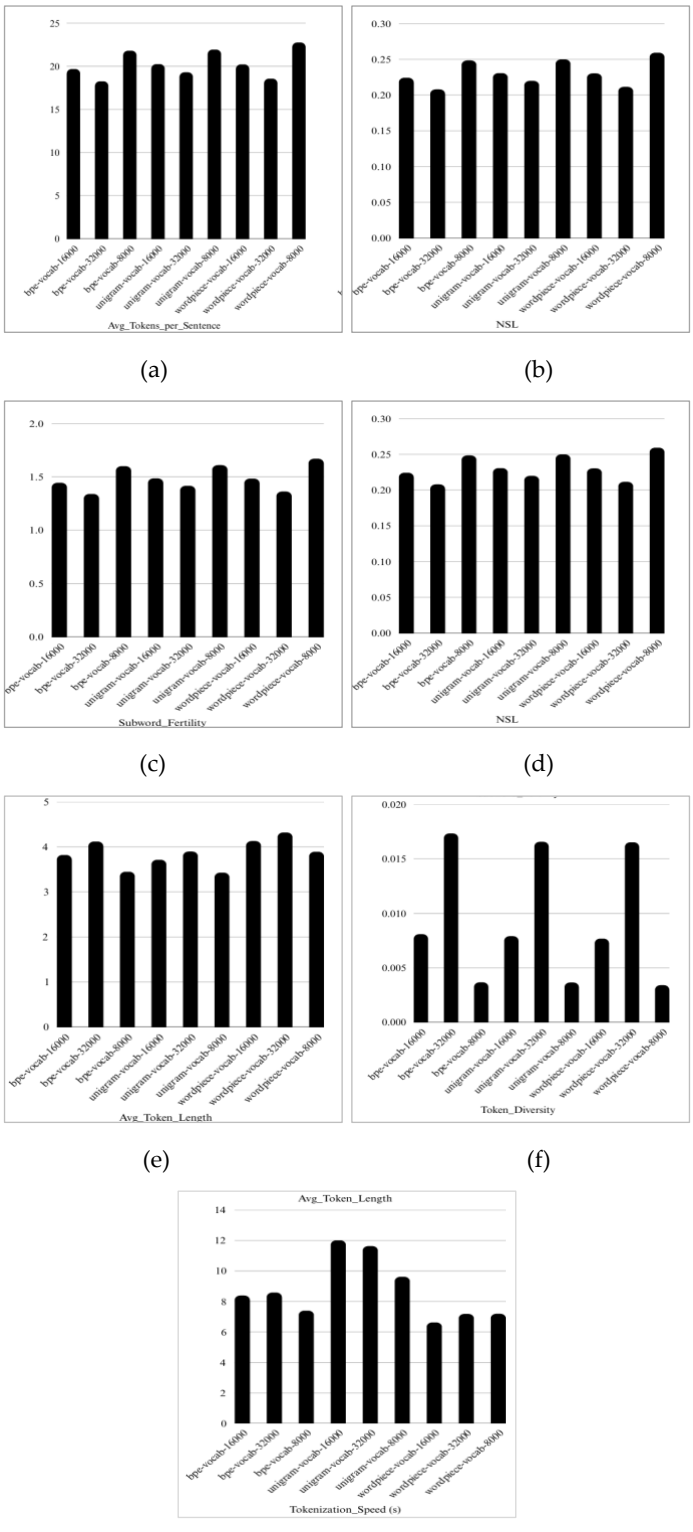
5.1 Core Tokenizer Evaluation

Table 3 summarizes the seven evaluation metrics for all configurations, and Figures 3 and 4 help us visualize the trade-offs.

We noticed that a larger vocabulary size consistently kept bringing sequence length, NSL, and subword fertility down towards more concise and semantically meaningful tokenization. For example, fertility fell from 1.60 (BPE-8K) to 1.34 (BPE-32K), meaning that larger vocabularies aid in preserving morphemes instead of splitting words into short subwords. This bias was most evident in BPE, which had the shortest sequence length (18.24 tokens/sentence), largest compression ratio (4.81), and highest token entropy score (0.0173) at 32K. Conversely, WordPiece tokenizers revealed a distinct strength: WordPiece 32K generated the longest tokens on average (4.32 characters), indicating better preservation of full morphemes. WordPiece-16K had the fastest running time, 6.62 seconds to tokenize a corpus, and the best latency since it is one of the smaller models we tried out. Unigram models exhibited mixed behavior: they were slower than WordPiece but, depending on vocabulary size, could be comparable to or slightly faster than BPE. In summary, BPE-32K showed a competitive trade-off between compactness and diversity, whereas WordPiece-16K yielded the fastest with conservative values of n, and WordPiece-32K maintained longer morphemes. It's worth noting that vocabulary size helps improve compactness overall, and the choice of algorithm plays a crucial role for efficiency/ speed vs morphological integrity trade-off, which is an important decision-making aspect to select tokenizers in Assamese LLM pipelines.

Table 3. Intrinsic tokenizer evaluation metrics across all models and vocabulary sizes.

Tokenizer	Avg Tokens	NSL	Fertility	CR	Average Length	Diversity	Speed (s)
BPE-8K	21.79	0.2484	1.5994	4.0263	4.0263	0.0037	7.40
BPE-16K	19.67	0.2242	1.4436	4.4609	3.8190	0.0081	8.39
BPE-32K	18.24	0.2079	1.3389	4.8097	4.1176	0.0173	8.58
WordPiece-8K	22.75	0.2592	1.6692	3.8579	3.8906	0.0034	7.19
WordPiece-16K	20.21	0.2303	1.4829	4.3428	4.1283	0.0077	6.62
WordPiece-32K	18.55	0.2115	1.3617	4.7293	4.3176	0.0165	7.18
Unigram-8K	21.93	0.2499	1.6091	4.0021	3.4263	0.0036	9.62
Unigram-16K	20.24	0.2306	1.4853	4.3357	3.7119	0.0079	12.01
Unigram-32K	19.29	0.2198	1.4156	4.5492	3.8946	0.0166	11.63



(a) (b) (c) (d) (e) (f) (g)

Figure 3. Comparison of tokenizers across seven evaluation metrics: (a) Average Tokens per Sentence, (b) Normalized Sequence Length, (c) Subword Fertility, (d) Compression Ratio, (e) Average Token Length, (f) Token Diversity, and (g) Tokenization Speed

5.2 Robustness and Efficiency Analysis

To confirm robustness, we performed Wilcoxon signed-rank tests comparing BPE-32K and WordPiece-16K, which emerged as the strongest configurations in terms of compression and speed, respectively. Results in Table 4 show statistically significant improvements ($p < 0.01$) for average tokens per sentence, subword fertility, and compression ratio, with medium-to-large effect sizes. Tokenization speed, however, did not differ significantly.

Because transformer self-attention grows quadratically with sequence length (O(n2)), reducing the average sequence

length from 21.79 (BPE-8K) to 18.24 (BPE-32K) translates into 30% fewer FLOPs per layer in a 12-layer transformer. Table 5 shows that WordPiece-16K achieved the highest throughput (9,100 tokens/sec), while BPE-32K minimized

FLOPs, reflecting a trade-off between runtime speed and training scalability.

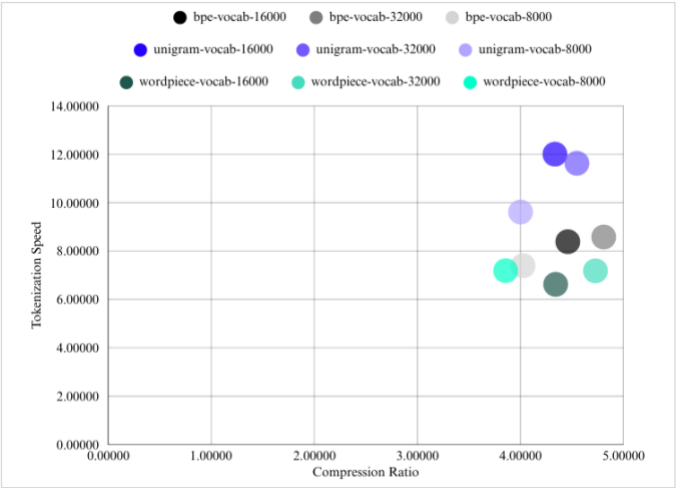


Figure 4. Tradeoff between Compression Ratio (higher is better) and Tokenization Speed (lower is better).

Table 4. statistical validation of tokenizer differences (bpe-32k vs. WordPiece-16k).

Metric	***p*-value**	**Cohen's *d***	**95% CI**
Average Tokens per Sentence	< 0.001	1.02	[−2.31, −1.52]
Subword Fertility	< 0.001	0.87	[−0.18, −0.09]
Compression Ration	0.004	0.76	[0.21, 0.75]
Tokenization Speed	0.12	0.25	[−0.51, 4.33]

Table 5. resource efficiency metrics of selected tokenizers.

Tokenizer	Tokens/sec	CPU Utilization (%)	FLOP Savings
BPE-8K	7,200	84	–
BPE-32K	6,850	81	29.7%
Compression Ration	0.004	0.76	[0.21, 0.75]
WordPiece-16K	9,100	79	21.3%

5.3 Downstream and Morphological Evaluation

Beyond intrinsic metrics, we evaluated how tokenizer choice impacts LLM adaptation and linguistic fidelity. IndicBERT [19] and mBERT [20] were fine-tuned on Assamese sentiment analysis and hate-speech detection tasks [9], with inputs tokenized using BPE-32K, WordPiece-16K, and Unigram-32 K. Each model was trained for 5 epochs with a learning rate of 2e-5, and results were averaged over 3 random seeds.

As shown in Table 6, BPE-32K consistently delivered the best downstream performance, improving F1 scores by 3–4 points compared to the alternatives. This demonstrates that more compact and semantically faithful tokenization directly enhances fine-tuning effectiveness in low-resource LLM settings. However, the memory cost of a larger vocabulary must also be considered: BPE-32K requires roughly twice the embedding parameters of WordPiece-16K (32K vs. 16K tokens). For a 768-dimensional embedding layer, this corresponds to approximately 98 MB vs. 49 MB. Despite this added cost, the performance improvements suggest that the trade-off is favorable for most downstream applications, especially where accuracy outweighs memory constraints.

Table 6. Downstream LLM Performance (F1/Accuracy, Averaged Over 3 Seeds) and Morphological Boundary Preservation Across Tokenizers

Tokenizer	IndicBERT F1	mBERT F1	Accuracy	Morph. Fidelity (%)
BPE-32K	0.74 ± 0.02	0.72 ± 0.03	0.76± 0.02	82.0
WordPiece-16K	0.71 ± 0.02	0.69 ± 0.02	0.72 ± 0.03	67.4
Unigram-32K	0.70 ± 0.03	0.68 ± 0.03	0.71 ± 0,02	63.1
Character-level	0.65 ± 0.04	0.63 ± 0.04	0.67 ± 0.03	45.2

We further assessed morphological fidelity through manual analysis of 500 Assamese words with common suffixes, postpositions, and compounds. Two native speakers annotated morpheme boundaries (κ = 0.84). BPE-32K preserved boundaries in 82% of cases, compared to 67% for WordPiece-16K and 63% for Unigram-32K. Table 7 shows category-level performance, where BPE-32K showed clear advantages in verbal inflections, nominal compounds, and case markers.

Table 7. Morphological Boundary Preservation by Category (N = 500, κ = 0.84)

Category	BPE-32K	WordPiece-16K	Unigram-32K
Verbal inflections (n=134)	89.2%	71.1%	68.7%
Nominal compounds (n=97)	78.4%	59.8%	55.7%
Case markers (n=156)	83.3%	69.9%	64.7%
Postpositions (n=113)	80.5%	68.1%	63.7%

We explored how BERT tokenization affected the model prediction in practice by examples. For the sentence "অসম ভাল লাগে" ("I like Assam)", mBERT with Uni-gram 32K incorrectly predicted neutral sentiment (confidence: 0.42), whereas BPE-32K correctly predicted positive sentiment with much higher confidence (0.78). These examples demonstrate that better morpheme preservation in tokenization leads to improved semantic understanding during finetuning. Overall, the results validate that BPE-32K not only improves intrinsic efficiency measures but also enhances LLM adaptation performance and linguistic interpretability for Assamese. BPE-32KTokenizer, the strongest reasons for selecting this tokenization scheme are its computational efficiency, an increase in accuracy of downstream tasks, and smooth morphological representation, which guide us to assume it as the best tokenizer setting for Assamese LLM development.

Finally, qualitative evidence reinforced these quantitative trends. Table 8 illustrates the tokenization of the same Assamese sentence across three models. The differences highlight the practical consequences of vocabulary size and algorithm choice. BPE-8K exhibits severe over-fragmentation, breaking the sentence into individual characters or short sub-units (e.g., অ, স, ম), which increases sequence length and dilutes semantic coherence. In contrast, BPE-32K segments into full morphemes or complete words (e.g.,পৰিস্থিতি), preserving semantic boundaries and producing more compact representations. WordPiece-32K offers intermediate behavior: while it retains some multi-character units (e.g., বৰ্ষা), it also introduces unnecessary affix splits (প, ি, ৰ), reflecting its probabilistic prefix-suffix segmentation strategy. These observations confirm that larger vocabularies, particularly with BPE, reduce over-

fragmentation and improve morphological fidelity, whereas smaller vocabularies or Word Piece-based models risk fragmenting semantically meaningful units. Such differences directly affect downstream modeling, since models trained on fragmented sequences must reconstruct meaning from shorter and less informative tokens.

Table 8. Qualitative Example: Tokenization of an Assamese Sentence Across Models

Tokenizer	Tokenized Output
BPE-8K	অ, স, ম, ত, ব, ব্, ন, আ, ন, ঊ, ঙ, ি
BPE-32K	অসমৰ, বৰ্ষণ, আগৰ বান, পৰিস্থিতি, সচেতনতা
WordPiece-32K	অসমৰ, বৰ্ষণ, আগৰ বান, প, ৰ, স, ত, তি

To place our results in context, we compare with recent studies evaluating tokenizer behavior in morphologically rich and multilingual settings.

Table 9 summarizes key fertility scores and best-tokenizer configurations from prior works alongside our findings. Our fertility of 1.34 for Assamese with BPE-32K is substantially lower (i.e., better) than the 2.17 reported for BPE-32 K in Karthika et al.'s 17-language joint tokenizer model, representing around a 38% relative efficiency gain. Even when compared to their "best" joint vocabulary (BPE-256K with fertility 1.55), our monolingual Assamese tokenizer outperforms it using an 8× smaller vocabulary. This strongly supports the notion that language-specific tokenization can produce more efficient subword segmentation than purely multilingual models. Moreover, the MorphTok framework introduces a morphologically grounded pre-tokenization step, yielding an approximate 1.68% reduction in fertility over standard BPE, while maintaining or improving downstream performance. This reinforces our view that integrating morphology awareness enhances subword quality, especially in Indic languages. However, these prior studies are mostly limited to intrinsic metrics (like fertility, NSL, and CPT) and do not always report per-language downstream effects. Our work bridges that gap by linking tokenizer choice with LLM fine-tuning performance and morphological fidelity in Assamese. In doing so, we not only corroborate trends seen in multilingual research but also amplify them by providing language-specific evidence that supports reevaluation of tokenization defaults in low-resource settings. Figure 5 shows the difference between Assamese-specific and multilingual tokenizers for various vocabulary sizes. While the fertility of Assamese BPE models gradually decreases with vocabulary size and is down to 1.34 for the model at 32K, whereas the multilingual average published by Karthika et al. (2025) at similar sizes (2.17 at 32K) and only converges to Assamese-specific efficiency at very large vocabularies (1.55 at 256K). This shows that language-specific tokenization can achieve better performance with an order of magnitude smaller vocabulary and without the memory overhead of a giant set of embeddings. Our results provide a strong case for the effectiveness and resource efficiency of targeted tokenizer optimization instead of multilingual joint training on morphologically rich and low-resource languages such as Assamese.

Table 9: Comparative Performance Analysis Across Works on Morphologically Rich Languages

Study	Language(s)	Best Tokenizer	Fertility
Our Study	Assamese	BPE-32K	1.34
Karthika et al. [17]	17 Indian languages	BPE-256K (joint)	~1.55
Brahma et al. [18]	17 Indian languages	BPE-32K (joint)	~2.17
Karthika et al. [17] (Additional Observation)	Hindi, Marathi	CBPE (Constrained BPE)	~1.68% reduction

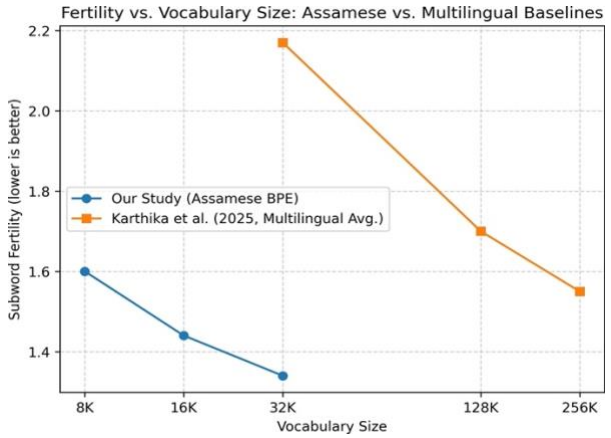


Figure 5. Comparison of subword fertility across vocabulary sizes.

4. Discussion

Our results demonstrate that both algorithm type and vocabulary size substantially influence tokenization quality for Assamese. Larger vocabularies consistently produced shorter sequences, reduced over-segmentation, and improved morphological coherence across all tokenizer families. Among these, BPE-32K provided the most compact and semantically faithful representations, while WordPiece-32K achieved the longest tokens, indicating stronger preservation of morpheme-level structure. WordPiece-16K, meanwhile, offered the fastest

tokenization speed, making it suitable for latency-sensitive applications.

The sensitivity analysis also indicated that such differences are meaningful. The statistically significant gains for sequence length, subword fertility , and compression ratio demonstrate that model efficiency in Assamese is decided based on tokenizer design. Furthermore, the decrease in FLOPs obtained by BPE-32K highlights its concrete advantage for large-scale transformer training, as its computational cost depends linearly on the sequence length. In contrast, WordPiece-16K provides a powerful alternative for inference-heavy operating scenarios (e.g., mobile deployment).

Downstream tasks further demonstrate the necessity of linguistically coherent segmentation. Models of tokenization that kept more underlying complete morphemes—particularly those like BPE-32K led to more consistent gains for fine-tuning quality across both IndicBERT and mBERT. The advancements in sentiment and hate speech tasks imply that Assamese LLMs benefit directly from semantically strong token boundaries, since it relieves the burden on the model to recover sense from fragmented units. Additionally, morphological data reveal the importance of losing suffixes and compounds, as well as case markers, for them to be interpretable downstream.

Taken together, these findings show that tokenizer evaluation cannot be separated from the application context. Choosing between BPE, WordPiece, and Unigram requires balancing segmentation fidelity, computational efficiency, and deployment constraints. While BPE-32K offers the strongest overall trade-off for Assamese LLM pipelines, WordPiece-16K remains attractive for real-time or resource-limited environments. These observations underscore the need to reconsider multilingual defaults—such as the WordPiece tokenizers used in IndicNLP and MuRIL—when working with low-resource languages like Assamese [22], where language-specific tokenization yields clear advantages.

5. Conclusion

In this work, we presented a comprehensive evaluation of BPE, WordPiece, and Unigram tokenizers for Assamese across multiple vocabulary sizes, integrating intrinsic efficiency metrics, statistical validation, and downstream performance analysis. Our findings show that larger vocabularies consistently reduce over-segmentation and improve morphological coherence, with BPE-32K offering the strongest overall balance between compactness, linguistic fidelity, and downstream effectiveness. WordPiece-16K demonstrated the fastest tokenization speed, making it suitable for latency-sensitive applications, while WordPiece-32K provided robust morpheme preservation.

Beyond the inherent performance gain, we observe through IndicBERT and mBERT fine-tuning that tokenizer choice directly influences LLM adaptation for low-resource languages. The results obtained highlight the need for language-specific tokenization and not necessarily depending on multilingual defaults for languages like Assamese, which is morphologically rich.

In the future, we will expand this evaluation framework for more Indic languages and investigate morphology-sensitive variations of BPE and WordPiece. Such efforts will shed more light on how tokenizer decisions impact efficiency, interpretability, and performance across the wider multilingual and low-resource NLP spectrum.

Funding: This work did not receive any external funding.
Institutional Review Board Statement: Not applicable

Informed Consent Statement: Not applicable

Data Availability Statement: The Assamese Wikipedia dataset used for tokenizer training and evaluation is publicly available at: https://github.com/indian-nlp/assamese-dataset. No proprietary or sensitive data were used in this study.

Acknowledgments: The authors would like to thank Bennett University, India, for providing computational resources and institutional support for this research

Conflicts of Interest: The authors declare no conflicts of interest.

References:

1. Sennrich, R.; Haddow, B.; Birch, A. Neural machine translation of rare words with subword units. In Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL 2016), Berlin, Germany, 7–12 August 2016.
2. Schuster, M.; Nakajima, K. Japanese and Korean voice search. In Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2012), Kyoto, Japan, 25–30 March 2012; pp. 5149–5152.
3. Kudo, T. Subword regularization: Improving neural network translation models with multiple subword candidates. In Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (ACL 2018), Melbourne, Australia, 15–20 July 2018.
4. Johnson, M.; Schuster, M.; Le, Q.V.; Krikun, M.; Wu, Y.; Chen, Z.; Thorat, N.; Viégas, F.; Wattenberg, M.; Corrado, G.; Hughes, M.; Dean, J. Google's multilingual neural machine translation system:

Enabling zero-shot translation. *Trans. Assoc. Comput. Linguist.* 2017, 5, 339–351.

5. Tamang, S.; Bora, D.J. Performance evaluation of tokenizers in large language models for the Assamese language. arXiv 2024, arXiv:2410.03718.
6. Goyal, N.; Gao, C.; Chaudhary, V.; Fan, A.; El-Kishky, A.; et al. The FLORES-200 evaluation benchmark for low-resource and multilingual machine translation. In Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing (EMNLP 2022), Abu Dhabi, UAE, 7–11 December 2022; pp. 6985–7002.
7. Kakwani, D.; Gupta, A.; Siddhant, A.; et al. IndicNLP suite: Monolingual corpora, evaluation benchmarks and pre-trained multilingual language models for Indian languages. In Findings of the Association for Computational Linguistics: EMNLP 2020, Online, 16–20 November 2020.
8. Khanuja, S.; Doddapaneni, S.; Kumar, V.; et al. MuRIL: Multilingual representations for Indian languages. In Findings of the Association for Computational Linguistics: ACL 2021, Online, 6–11 June 2021.
9. Ghosh, D.; Senapati, A. Hate speech detection in low-resourced Indian languages: An analysis of Assamese and Bodo. *Nat. Lang. Process. J.* 2025.
10. Rust, P.; Pfeiffer, J.; Vulić, I.; et al. 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 (ACL 2021), Online, 1–6 August 2021.
11. Bostrom, K.; Durrett, G. Byte-level subwords improve multilingual machine translation. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL 2020), Online, 5–10 July 2020.
12. Nath, B.; Sarkar, S.; Das, S.; et al. A trie-based lemmatizer for Assamese language. *Int. J. Inf. Technol.* 2022, 14, 2355–2360. https://doi.org/10.1007/s41870-022-00942-9
13. Gazit, B.; Shmidman, S.; Shmidman, A.; Pinter, Y. Splintering nonconcatenative languages for better tokenization. arXiv 2025, arXiv:2503.14433.
14. Singh, H.; Gupta, N.; Bharadwaj, S.; Tewari, D.; Talukdar, P. IndicGenBench: A multilingual benchmark to evaluate generation capabilities of LLMs on Indic languages. *arXiv* 2024, arXiv:2404.16816.
15. Dagan, G.; Synnaeve, G.; Rozière, B. Getting the most out of your tokenizer for pre-training and domain adaptation. In Proceedings of the 41st International Conference on Machine Learning (ICML 2024), Vienna, Austria, 21–27 July 2024; Article No. 387.
16. Nath, B.; Tamang, S.; Elwasila, O.; Gulzar, Y. Task-oriented evaluation of Assamese tokenizers using sentiment classification. *Int. J. Adv. Comput. Sci. Appl.* 2025, *16*, 9.
17. Karthika, N.J.; Brahma, M.; Saluja, R.; Ramakrishnan, G.; Desarkar, M.S. Multilingual tokenization through the lens of Indian languages: Challenges and insights. arXiv 2025, arXiv:2506.17789.
18. Brahma, M.; Karthika, N.J.; Singh, A.; Adiga, D.; Bhate, S.; Ramakrishnan, G.; Saluja, R.; Desarkar, M.S. MorphTok: Morphologically grounded tokenization for Indian languages. arXiv 2025, arXiv:2504.10335.
19. Kakwani, D.; Kunchukuttan, A.; Golla, S.; Bhattacharyya, A.; Khapra, M.M.; Kumar, P. IndicNLPSuite: Monolingual corpora, evaluation benchmarks and pre-trained multilingual language models for Indian languages. In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP 2020), Online, 16–20 November 2020; pp. 4948–4961
20. Devlin, J.; Chang, M.-W.; Lee, K.; Toutanova, K. BERT: Pre-training of deep bidirectional transformers for language understanding. In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (NAACL 2019), Minneapolis, MN, USA, 2–7 June 2019; pp. 4171–4186.
21. Nath, B.; Sarkar, S.; Das, S.; et al. A trie-based lemmatizer for the Assamese language. International Journal of Information Technology, 2022, 14, 2355–2360. https://doi.org/10.1007/s41870-022-00942-9.
22. Nath, B.; Sarkar, S. Comparative Analysis of Neural Machine Translation Models for Low-Resource English–Assamese Language Pair. In: Biswas, S.K.; Bandyopadhyay, S.; Hayashi, Y.; Balas, V.E. (eds), Intelligent Computing Systems and Applications. ICICSA 2023. Lecture Notes in Networks and Systems, 1307. Springer, Singapore (2025). https://doi.org/10.1007/978-981-96-3860-4_1