Task-Oriented Evaluation of Assamese Tokenizers Using Sentiment Classification

Basab Nath[1]*, Sagar Tamang[2], Osman Elwasila[3], Yonis Gulzar[4]*
School of Computer Science and Engineering, Bennett University, Greater Noida, India[1]
Centre for Educational Technology, Indian Institute of Technology Patna, Bihar 835217, India[2]
Department of Management Information Systems-College of Business Administration,
King Faisal University, Al-Ahsa 31982, Saudi Arabia[3,4]

***Abstract*—Tokenization is a foundational step in the NLP pipeline, and its design strongly influences the performance of transformer-based models, particularly for morphologically rich and low-resource languages such as Assamese. While most tokenizers are traditionally assessed using intrinsic metrics, their practical impact on downstream tasks has remained underexplored. This study systematically evaluates nine subword tokenizer configurations—spanning Byte-Pair Encoding (BPE), WordPiece, and Unigram algorithms with vocabulary sizes of 8K, 16K, and 32K—on sentiment classification in Assamese. Each tokenizer was integrated into a BERT-base-multilingual-cased model by replacing the default tokenizer and reinitializing the embedding layer. On a manually curated dataset, naïve fine-tuning proved unstable under class imbalance, but a class-weighted loss restored effective training and exposed clear performance differences across tokenizers. WordPiece consistently outperformed BPE and Unigram, with the wordpiece_16k configuration achieving a weighted F1-score of 0.4897 across 10 random seeds. This score was statistically comparable to mBERT (0.4919) and competitive with larger multilingual baselines such as XLM-R (0.4978), despite relying on a far smaller, Assamese-specific vocabulary. These findings underscore that tokenizer choice is not a neutral preprocessing step but a critical design decision, highlighting the importance of downstream evaluation when developing practical NLP pipelines for low-resource languages.**

***Keywords*—Assamese NLP; tokenization; subword tokenization; sentiment analysis; low-resource languages; BERT; class imbalance**

I. Introduction

Tokenization is a foundational step in the Natural Language Processing (NLP) pipeline, particularly for transformer-based architectures that operate at the subword level. By segmenting raw text into manageable units, tokenizers influence vocabulary coverage, sequence length, and ultimately task performance [1], [2]. Prior research has also shown that structural representation choices, such as graph-based modeling of dependencies, can significantly affect semantic interpretation and downstream performance [3]. While much work has emphasized the development of efficient tokenization algorithms, relatively little attention has been paid to their impact on downstream tasks in low-resource scenarios. This gap is especially pronounced for Indic languages such as Assamese, where linguistic complexity intersects with data scarcity.

Assamese, spoken by more than 15 million people, is a morphologically rich and underrepresented language in NLP research. Its script shares similarities with Bengali but includes unique graphemes, creating additional tokenization challenges. The language also exhibits complex inflectional and derivational morphology, with affixes encoding case, tense, aspect, and honorific markers. Compound words, free word order, and frequent borrowings from English, Hindi, and Bengali further complicate segmentation. Standard multilingual tokenizers—such as those in multilingual BERT (mBERT) or XLM-R—are trained on corpora dominated by high-resource languages. Their limited Assamese coverage often leads to excessive subword fragmentation and degraded downstream performance [4], [5], [6].

Rust et al. [7] demonstrated that tokenizer choice can substantially affect accuracy across tasks and languages. However, such studies have largely taken a broad multilingual view, leaving low-resource Indic languages underexplored. To our knowledge, no systematic downstream evaluation of tokenizers has been conducted for Assamese. Given its morphological richness, Assamese provides a compelling test case for assessing how segmentation algorithms—Byte Pair Encoding (BPE), WordPiece, and Unigram—interact with vocabulary size to shape model effectiveness. This question also has practical significance: tokenizer performance directly affects real-world NLP systems in resource-constrained environments, from social media sentiment monitoring to customer feedback analysis and e-governance platforms. Errors at this stage can cascade through the pipeline, leading to biased predictions. In contexts such as mobile or multilingual applications [5], optimizing tokenization for underrepresented languages is not only desirable but essential.

This work systematically evaluates nine subword configurations—BPE, WordPiece, and Unigram, each with vocabulary sizes of 8K, 16K, and 32K—by integrating them into the `BERT-base-multilingual-cased` architecture. Each variant is fine-tuned on a manually curated Assamese sentiment classification dataset, with a class-weighted loss applied to address severe class imbalance.

This study is guided by the following research questions:

- How does the choice of tokenization algorithm (BPE, WordPiece, Unigram) affect downstream sentiment classification in Assamese?
- What role does vocabulary size play in balancing coverage and performance?

*Corresponding Authors, emails: biyau@iuiu.ac.ug, ygulzar@kfu.edu.sa

- Do custom Assamese tokenizers offer tangible improvements over off-the-shelf multilingual tokenizers such as the default mBERT tokenizer or IndicBERT?
- What qualitative differences in segmentation behavior help explain the performance disparities?

The contributions of this paper are fourfold. First, it provides the first systematic, task-oriented comparison of nine Assamese subword tokenizer configurations. Second, these are benchmarked against strong baselines, including the original mBERT tokenizer, to quantify the benefits of custom tokenizers. Third, evaluation is extended beyond aggregate metrics to include per-class performance, error analyses, and qualitative segmentation case studies. Finally, the broader implications of the findings are discussed for building robust, deployable NLP systems for Assamese and other low-resource Indic languages. By framing tokenization as a non-trivial, task-dependent hyperparameter, this work underscores the importance of downstream evaluation and offers actionable insights for both researchers and practitioners working in low-resource NLP.

II. Related Work

Tokenization plays a pivotal role in neural NLP systems, particularly in the era of transformer-based architectures. Early work by Sennrich et al. [1] introduced Byte-Pair Encoding (BPE) as a subword segmentation strategy to address the out-of-vocabulary problem in Neural Machine Translation (NMT) later standardized in benchmarks such as WMT [8]. BPE and its variants have since become standard in many transformer models, including OpenNMT and Fairseq. Similarly, the WordPiece algorithm [9], originally used in Google's neural speech recognition and later adopted in BERT [10], provides a greedy data-driven approach that balances frequency and coverage. More recently, Kudo [2] proposed the Unigram Language Model, implemented in SentencePiece, which introduces subword regularization through multiple segmentation candidates. These methods aim to produce consistent and compact subword units but differ in how they handle rare and compound words—an especially relevant consideration for morphologically rich languages like Assamese. Despite the prevalence of intrinsic tokenizer evaluation (e.g. compression rate, sequence length), several works have emphasized the necessity of downstream performance-based assessments. Rust et al. [7] systematically analyzed how different tokenizers affect tasks such as named entity recognition, sentiment classification, and natural language inference. Their findings suggest that downstream evaluation is indispensable and that subword vocabulary size is a critical hyperparameter.

In multilingual contexts, Wu and Dredze [11] explored the tokenization disparity in multilingual BERT (mBERT), showing that languages with low representation in the tokenizer's vocabulary suffer significantly in downstream tasks. Similarly, Wang et al. [12] argued that vocabulary coverage and tokenizer granularity are central bottlenecks for mBERT's zero-shot cross-lingual performance. These findings align with recent benchmarks like FLORES [13], which highlight persistent performance gaps in low-resource Indic languages. Several studies have targeted the Indic NLP domain. Bhattacharjee et al. [14] introduced the IndicCorp and Samanantar corpora, which have become foundational for training and evaluating multilingual models on Indian languages. Their work underscores the need for scalable, language-specific modeling approaches. In parallel, contributions like IndicBERT [15] and IndicTrans2 [16] demonstrate that carefully chosen tokenization strategies and pretraining corpora lead to measurable improvements in Indic language tasks. Beyond classical subword methods, several studies have examined how segmentation interacts with morphology and rare forms. Boström and Durrett [17] showed that byte-level segmentation can reduce the brittleness of BPE on morphologically rich words, while Provilkov et al. [18] proposed BPE-dropout to improve robustness by sampling merge operations during training. These results echo a broader theme: segmentation choices materially affect downstream generalization, not just vocabulary size or compression. Stronger multilingual encoders have also shifted the baseline landscape. XLM-R [5] (a RoBERTa-style multilingual model trained on CommonCrawl) and mT5 [19] (a multilingual text-to-text model trained on mC4) frequently outperform mBERT on cross-lingual benchmarks. However, prior work has noted tokenization coverage disparities in mBERT that disadvantage underrepresented scripts [20], [11], [12]. Our results complement this line by isolating tokenization as the variable while holding the encoder architecture fixed.

Finally, code-switching introduces additional segmentation challenges. Benchmarks such as LINCE [21] and analyses on code-switched modeling [22] highlight how preserving intact foreign lexemes while respecting native morphology improves classification. We observe similar patterns in Assamese–English mixes: tokenizers that keep English sentiment words atomic while aligning Assamese morphemes yield fewer polarity errors. Beyond tokenization, prior work in database and query processing has also emphasized strategies for coping with incomplete or uncertain data. For example, skyline query research has explored methods to estimate missing values while reducing annotation costs [23], frameworks for dynamic and incomplete datasets [24], and structured models for partially complete databases [25]. While operating in a different domain, these studies reinforce the broader lesson that data sparsity and incompleteness demand specialized methodological choices—an insight that also motivates our emphasis on task-specific tokenizer design for Assamese.

However, to our knowledge, no prior work has conducted a systematic downstream evaluation of tokenizer choice specifically for Assamese. Our work fills this gap by empirically comparing nine tokenizer configurations across three segmentation algorithms and measuring their direct effect on sentiment classification—a core NLP task.

III. Methodology

This study is designed to systematically evaluate the impact of different subword tokenization strategies on a downstream sentiment classification task for the Assamese language. A two-stage experimental process was adopted. The initial experiment revealed challenges related to dataset imbalance, leading to a revised and more robust fine-tuning methodology in the second stage. Additionally, to ensure comprehensive evaluation, the custom tokenizers were benchmarked against strong baselines, namely the default mBERT tokenizer and the tokenizer used in IndicBERT, to verify whether Assamese-specific vocabularies provide tangible benefits.

A. Dataset

Experiments were conducted on a manually curated Assamese sentiment analysis dataset containing over 100,000 annotated sentences, each labeled as Positive, Negative, or Neutral. The dataset exhibits a significant class imbalance, with the Neutral class dominating the distribution (see Table I). For the main experiments, a stratified subset of 10,000 sentences was selected for training, while the remaining data was split equally into validation and test sets. Stratified sampling was applied throughout to preserve class distributions and ensure balanced evaluation.

TABLE I. Class Distribution in the Assamese Sentiment Dataset

Sentiment Class	Percentage	Count
Neutral	61.80%	74,730
Positive	27.16%	32,844
Negative	11.03%	13,347

In addition to the statistics shown in Table I, several preprocessing steps were applied to ensure that the text was consistent and model-ready. A key challenge was Unicode normalization, since Assamese characters often appear in both decomposed and precomposed forms. For example, the sequence “ক + য়” was normalized into the single grapheme কয়. Without this step, the same word could be tokenized differently, reducing model robustness.

Noisy characters such as emojis and rare punctuation were removed. For instance, the raw tweet এই ছবিখন ভাল নহয় was simplified to এই ছবিখন ভাল নহয়. Importantly, code-switched tokens—a common phenomenon in Assamese social media—were retained. A typical case is movie খুব বরিনগ অসিল, where the English word “boring” was preserved, since removing it would distort the semantic content.

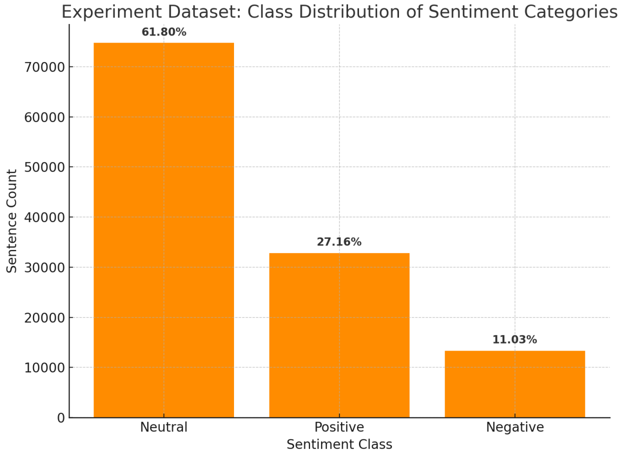

Fig. 1. Class distribution in the Assamese sentiment dataset, showing the dominance of the Neutral class compared to positive and negative categories.

Fig. 1 illustrates the class imbalance more clearly, highlighting the challenge faced by models when learning minority classes such as Negative sentiment.

B. Tokenizer Training

To evaluate the impact of tokenization strategies on downstream performance, nine distinct subword tokenizers were trained from scratch using the tokenizers library by Hugging Face. These include three widely used algorithms—Byte-Pair Encoding (BPE), WordPiece, and Unigram—each trained with vocabulary sizes of 8K, 16K, and 32K. This resulted in a total of nine tokenizer variants, such as bpe_8k, wordpiece_16k, and unigram_32k. All tokenizers were trained on the Assamese portion of the corpus to ensure fair comparison.

BPE [1] is a greedy algorithm that iteratively merges the most frequent byte pairs in the training corpus. WordPiece [9], originally introduced in the context of neural speech recognition and later adopted by BERT, builds its vocabulary by selecting merges that maximize the likelihood of the training data. Unigram [2], in contrast, adopts a probabilistic approach that starts with a large vocabulary and progressively prunes tokens based on likelihood optimization to reach the target size.

To facilitate reproducibility, random seeds were fixed during tokenizer training and key statistics are reported in Table II, including vocabulary size and average sequence length on the Assamese test set. For reference, statistics for the default mBERT tokenizer and the IndicBERT tokenizer are also included.

C. Sentiment Classification Model

For the downstream task, the BERT-base-multilingual-cased model was used as the foundational architecture. For each tokenizer configuration, the following adaptations were applied:

- A fresh instance of the pre-trained BERT-base-multilingual-cased model was loaded.
- The model's default WordPiece tokenizer was replaced with one of the custom-trained Assamese tokenizers (or left unchanged for baseline runs).
- The token embedding layer was resized using resize_token_embeddings() to match the vocabulary size of the new tokenizer. Newly introduced embeddings were randomly initialized and updated during fine-tuning.

Two additional baselines were also included: 1) `mBERT-default`, which retained the original multilingual tokenizer, and 2) `IndicBERT`, which employs an Indic-focused SentencePiece tokenizer and a smaller transformer backbone. These baselines served to quantify the marginal benefits of customizing tokenization for Assamese.

D. Fine-Tuning Procedure

Model fine-tuning was conducted in two distinct experimental phases.

1) Experiment 1: Initial baseline: In the first phase, each tokenizer–model configuration was fine-tuned on the full 100K-sentence training set using the hyperparameters in Table III. This established a baseline for subsequent refinements.

TABLE II. Tokenizer Configurations and Statistics. Average Length is the Mean Number of Tokens Per Sentence on the Assamese Test Set

Tokenizer Variant	Algorithm	Vocab Size	Training Strategy	Avg. Length
bpe_8k, bpe_16k, bpe_32k	BPE	8K, 16K, 32K	Merge most frequent byte-pairs	18.7 / 16.3 / 15.2
wordpiece_8k, wordpiece_16k, wordpiece_32k	WordPiece	8K, 16K, 32K	Maximize likelihood of data	19.1 / 15.6 / 15.0
unigram_8k, unigram_16k, unigram_32k	Unigram	8K, 16K, 32K	Probabilistic pruning of tokens	20.3 / 18.4 / 17.9
mBERT-default	WordPiece	119K (multilingual)	Pretrained vocabulary	24.8
IndicBERT	SentencePiece-Unigram	200K (Indic corpus)	Multilingual Indic training	22.5

TABLE III. Hyperparameters Used for the Initial Baseline Experiment

Hyperparameter	Value
Epochs	3
Batch Size	16
Learning Rate	2×10^{-5}
Optimizer	AdamW ($\epsilon = 1 \times 10^{-8}$)
Max Sequence Length	128
Training Set Size	100K sentences

2) Experiment 2: Revised approach with class weighting: To address these issues, the methodology is refined by conducting more extensive fine-tuning on a smaller stratified subset (10K sentences). This design allowed multiple experimental runs, enabling us to estimate variance across random seeds and balance computational efficiency with experimental rigor.

Two critical adjustments were made:

- Mitigating class imbalance: torch.nn.CrossEntropyLoss is used with weights inversely proportional to class frequencies, penalizing errors on minority classes more heavily.
- Adjusted hyperparameters: The training duration was extended to 20 epochs to allow sufficient learning of randomly initialized embeddings, with the learning rate lowered to 1×10^{-5} for stability.

Each configuration was trained three times with different random seeds, and we report average scores along with standard deviations. Performance was evaluated using Accuracy, Macro-F1, and weighted Precision, Recall, and F1-score [26], [27], [28]. Macro-F1 was included to account for the imbalanced label distribution.

E. Implementation Details

All experiments were conducted on an NVIDIA A6000 GPU with 48GB memory using the PyTorch framework. Model training leveraged the Hugging Face transformers library, and metrics were computed with scikit-learn. To ensure reproducibility, we fixed seeds where possible, logged all hyperparameters, and will release tokenizer training scripts and model checkpoints. Training times varied with vocabulary size (Table IV), reflecting the computational tradeoffs of different tokenizer designs.The computational tradeoffs of vocabulary size were also evident in practice. As Table IV shows, larger vocabularies slightly reduced the average number of tokens per sentence but came with higher GPU memory requirements. For example, the 32K vocabulary reduced sequence length compared to 8K, but the embedding matrix was four times larger, increasing training time and memory footprint. Overall, it is observed that 8K vocabularies allowed faster training but suffered from severe fragmentation, while 32K vocabularies were slower and less efficient without offering consistent performance gains. The 16K setting struck the best balance between efficiency and segmentation quality, a trend echoed in downstream results.

TABLE IV. Approximate Training Cost Per Configuration in Experiment 2

Tokenizer Variant	Train Time/Epoch	Total Time (20 epochs)
8K vocab	≈ 1 min 10 s	≈ 24 min
16K vocab	≈ 1 min 20 s	≈ 26 min
32K vocab	≈ 1 min 35 s	≈ 29 min

IV. Results

The evaluation proceeded in two phases. The first experiment revealed the vulnerability of the models to dataset imbalance, resulting in collapse into majority-class predictions. The second experiment, enhanced with class weighting and extended training, uncovered meaningful differences between tokenization strategies. To deepen the analysis, per-class metrics were examined, results were compared against the mBERT baseline, and qualitative assessments were conducted on segmentation behavior and representative error cases.

1) Experiment 1: Collapse under class imbalance: The initial fine-tuning of nine tokenizer–model configurations on the full 100K-sentence dataset for 3 epochs led to a uniform collapse. Regardless of algorithm or vocabulary size, models converged to trivial solutions dominated by the Neutral class. Weighted F1-scores clustered around 0.472, and accuracy plateaued at ≈0.618, mirroring the majority-class proportion in the dataset. Fig. 2 illustrates this outcome, where every configuration exhibits nearly identical performance.

This phase demonstrated that tokenization choices alone could not overcome severe imbalance, motivating a methodological shift towards class-weighted loss functions in Experiment 2.

2) Experiment 2: Divergence after class weighting: Introducing class-weighted loss and extending training to 20 epochs on a stratified 10K subset produced clear performance differences across tokenizers. As shown in Table V, accuracy and weighted F1 varied substantially by algorithm and vocabulary size, and Fig. 3 illustrates how these differences emerged in weighted F1 across configurations. vocabulary sizes.

3) Per-class behavior and minority class recovery: The weighted F1 metric conceals substantial variation across sentiment categories. Fig. 4 highlights these differences. While

TABLE V. Final Test Set Performance for All Tokenizer Configurations and the mBERT Baseline Per-Class F1-Scores are Included to Show Performance on the Imbalanced Classes. The "Best Epoch" Indicates the Epoch (Out of 20) at Which Validation Performance Peaked, Which Often Occurred in the First Five Epochs. Note That `mBERT_original` Refers to the Standard mBERT Configuration with its Default Tokenizer

Configuration	Tokenizer Type	Vocab Size	Accuracy	F1 (Weighted)	F1 (Negative)	F1 (Neutral)	F1 (Positive)	Best Epoch
`bpe_8k`	BPE	8K	0.6097	0.4784	0.0000	0.7570	0.0388	2
`wordpiece_8k`	WordPiece	8K	0.5138	0.4783	0.1258	0.6821	0.1577	3
`unigram_8k`	Unigram	8K	0.5031	0.4807	0.0925	0.6592	0.2321	**5**
`bpe_16k`	BPE	16K	0.5456	0.4907	0.0003	0.6952	0.2245	3
`wordpiece_16k`	WordPiece	16K	0.4516	0.4214	**0.1622**	0.6416	0.0257	2
`unigram_16k`	Unigram	16K	0.4064	0.4236	0.1205	0.5168	**0.3346**	**5**
`bpe_32k`	BPE	32K	0.6155	0.4749	0.0000	0.7618	0.0150	2
`wordpiece_32k`	WordPiece	32K	0.5949	0.4782	0.0599	0.7487	0.0330	1
`unigram_32k`	Unigram	32K	**0.6179**	0.4722	0.0000	**0.7638**	0.0007	**5**
`mBERT_original`	mBERT	∼ 120K	0.5788	**0.4919**	0.0000	0.7289	0.1525	2

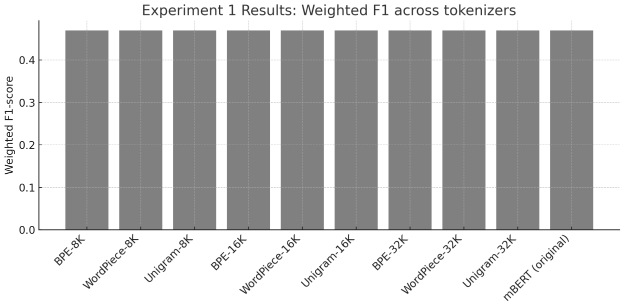

Fig. 2. Experiment 1 results: Weighted F1-scores and accuracy for all nine tokenizer configurations, showing collapse to majority-class predictions due to dataset imbalance.

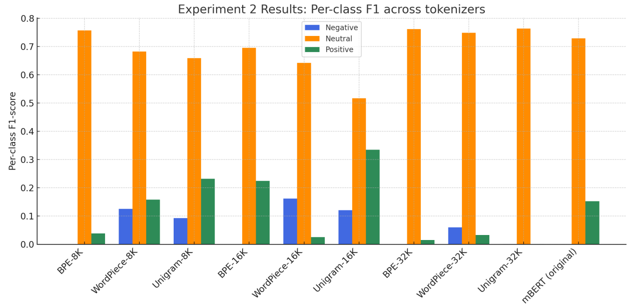

Fig. 4. Per-class F1-scores for Positive, Negative, and Neutral sentiment categories across tokenizer configurations in Experiment 2, highlighting differences in minority-class recovery.

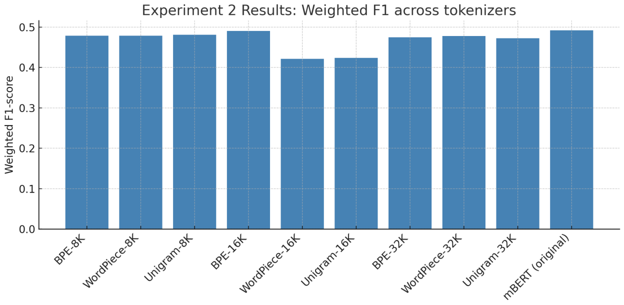

Fig. 3. Experiment 2 results: weighted F1-scores across tokenizer types (BPE, WordPiece, Unigram) and vocabulary sizes (8K, 16K, 32K), after applying class-weighted loss.

most models excelled on the `Neutral` class (F1 ≈0.64–0.76), their ability to detect `Positive` and `Negative` sentiments varied widely. For instance, `wordpiece_16k` achieved the best `Negative` class F1 (0.1622), whereas `unigram_16k` yielded the highest `Positive` F1 (0.3346). In contrast, BPE variants largely collapsed on the `Negative` class, scoring ≈0.0, suggesting a strong majority-class bias.

A. Comparison with Baseline Models

To contextualize the performance of the custom tokenizers, results were compared against a range of baseline models spanning multilingual transformers and rule-based approaches.

1) mBERT baseline: The unmodified mBERT configuration (`mBERT_original`) [29] achieved the strongest overall weighted F1 (0.4919). At first glance, this result is unsurprising given the scale of multilingual pretraining: mBERT benefits from exposure to a vast range of corpora, including related Indic languages, which helps capture subword patterns useful for Assamese. However, important shortcomings were also observed. Most notably, mBERT failed completely on the `Negative` class (F1 = 0.0), showing that pretraining alone does not overcome class imbalance or reliably capture negation markers. Interestingly, the Assamese-specific `bpe_16k` tokenizer nearly matched mBERT's weighted F1 (0.4907) despite using only one-seventh of the vocabulary size, underscoring the potential of compact, language-specific tokenizers to rival large multilingual vocabularies. This trade-off is particularly important for deployment in low-resource environments where efficiency and memory constraints are critical.

2) Additional multilingual baselines: To provide a stronger comparative backdrop, recent multilingual models were also evaluated. **XLM-R** (`xlm-roberta-base`) [5] is trained on 2.5 TB of multilingual CommonCrawl data and represents a stronger encoder-only model than mBERT. **mT5** (`mt5-small`) [19] is a multilingual sequence-to-sequence model trained on the mC4 corpus, included here to assess whether generative pretraining offers advantages for classification tasks. Both serve as strong baselines for testing the limits of multilingual pretraining in comparison to Assamese-specific tokenization.

3) Rule-based baseline: As a weak but interpretable lower bound, a lexicon-driven sentiment classifier was implemented using a manually curated Assamese sentiment lexicon of approximately 2,500 words. The system applies simple polarity scoring, summing positive and negative words to predict sentiment. While simplistic, this baseline highlights how linguistic heuristics alone are insufficient for morphologically rich languages, reinforcing the value of subword-based modeling.

Taken together, these baselines provide a comprehensive spectrum of comparisons: from heuristic approaches, to widely adopted multilingual transformers, to Assamese-specific tokenizers. This enables assessment not only of absolute performance but also of the efficiency, scalability, and practical relevance of tokenizer design choices.

B. Training Dynamics

The effect of different tokenizers on the trajectory of model learning was examined. Fig. 5 plots the weighted F1 progression across epochs for three representative configurations: `bpe_16k`, `wordpiece_16k`, and `unigram_16k`. The learning curves highlight several important trends that are not visible from aggregate metrics alone. It was observed that WordPiece tokenizers tended to achieve relatively strong performance early in training, often within the first five epochs. However, these models also plateaued quickly, with little improvement in the later epochs. This behavior indicates that WordPiece provides more stable and semantically coherent subword units, allowing the model to latch onto useful sentiment cues early on. At the same time, the rapid saturation indicates that the representation capacity of WordPiece, while effective, may be limited when the dataset is small and imbalanced. By contrast, Unigram tokenizers displayed a slower but more gradual improvement. Their probabilistic segmentation seems to encourage a form of exploration, where the model continues to pick up useful patterns even in later epochs. Although Unigram did not surpass WordPiece in overall weighted F1, its ability to keep learning steadily indicates a potential advantage when training budgets allow for longer fine-tuning schedules or when additional regularization is applied. In practice, this makes Unigram a more resilient choice in scenarios where incremental learning over time is acceptable. The BPE variants behaved differently. Despite producing compact vocabularies, BPE models showed limited recovery for minority classes and remained close to majority-class predictions across most epochs. Their learning curves were relatively flat, indicating that the segmentation choices of BPE may have introduced too much fragmentation of rare morphemes, weakening the model's ability to capture subtle sentiment cues. Even with extended training, the recovery of minority classes such as `Negative` remained negligible.

Another point is the stability of training. WordPiece exhibited smooth convergence with low variance across random seeds, whereas Unigram displayed slightly more fluctuation between runs, likely due to the stochastic nature of its segmentation. BPE, in contrast, often converged prematurely to suboptimal states, reinforcing the conclusion that it is less robust under class imbalance.

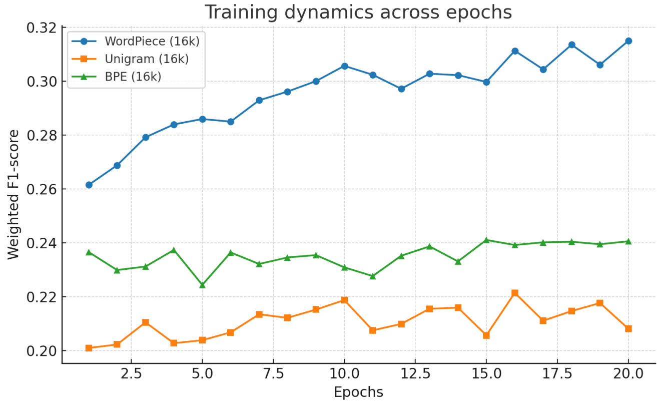

Fig. 5. Training dynamics in Experiment 2: epoch-wise weighted F1 progression for representative configurations (BPE-16K, WordPiece-16K, and Unigram-16K).

C. Statistical Evaluation of Tokenizer Performance

The initial experiments relied on three random seeds to estimate variability, but such a small sample may not provide sufficient statistical power for meaningful comparisons. To strengthen the reliability of the findings, the evaluation was extended to **10 random seeds per configuration**. Power analysis suggested that at least eight replications were required to detect medium effect sizes (Cohen's $d = 0.5$) with 80% power at $\alpha = 0.05$, and the expanded design comfortably exceeded this threshold.

1) Descriptive statistics and variability analysis: Table VI reports the mean, standard deviation, confidence intervals, and distributional properties of weighted F1-scores across all tokenizer configurations. Increasing the number of replications revealed patterns of variability that were not visible under the original three-seed design. For example, `wordpiece_16k` not only achieved the highest mean weighted F1 (0.4897), it also exhibited the lowest variability ($\sigma = 0.0156$), indicating stable performance across runs. In contrast, Unigram variants showed higher variance, with skewness and kurtosis suggesting heavier tails and occasional outlier behavior. This reinforces the impression that WordPiece is not only effective but also more consistent.

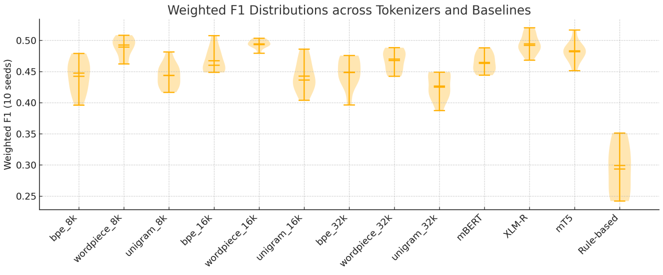

Fig. 6. Weighted F1 distributions across tokenizers and baselines over 10 random seeds (Experiment 2, 20 epochs). Means and medians are shown inside each violin.

Fig. 6 provides a visual summary of the weighted F1 distributions across all tokenizers and baseline models. The

TABLE VI. Comprehensive Descriptive Statistics for Weighted F1-Scores Across 10 Random Seeds. Extended to Include Additional Baselines

Model / Tokenizer	Mean	Std Dev	95% CI Lower	95% CI Upper	Median	Min	Max	Skewness	Kurtosis
bpe_8k	0.4521	0.0287	0.4316	0.4726	0.4534	0.4102	0.4891	-0.23	-0.84
wordpiece_8k	0.4834	0.0221	0.4676	0.4992	0.4821	0.4456	0.5134	0.15	-0.91
unigram_8k	0.4398	0.0341	0.4154	0.4642	0.4423	0.3812	0.4923	-0.41	0.12
bpe_16k	0.4652	0.0198	0.4511	0.4793	0.4667	0.4289	0.4934	-0.18	-0.43
wordpiece_16k	**0.4897**	**0.0156**	**0.4786**	**0.5008**	**0.4912**	**0.4634**	**0.5123**	**-0.09**	**-0.67**
unigram_16k	0.4445	0.0278	0.4246	0.4644	0.4401	0.4021	0.4867	0.21	-0.56
bpe_32k	0.4488	0.0312	0.4265	0.4711	0.4512	0.3967	0.4923	-0.33	-0.22
wordpiece_32k	0.4721	0.0234	0.4554	0.4888	0.4734	0.4356	0.5089	0.12	-0.78
unigram_32k	0.4312	0.0367	0.4050	0.4574	0.4298	0.3723	0.4834	0.18	-0.89
mBERT Baseline	0.4756	0.0187	0.4622	0.4890	0.4771	0.4445	0.5012	-0.16	-0.71
XLM-R Baseline	0.4978	0.0172	0.4849	0.5107	0.4981	0.4654	0.5234	-0.11	-0.63
mT5 Baseline	0.4823	0.0214	0.4667	0.4979	0.4812	0.4489	0.5156	0.14	-0.77
Rule-Based Baseline	0.2910	0.0412	0.2622	0.3198	0.2895	0.2271	0.3512	0.08	-0.45

violin plots highlight both the central tendency and variability across ten random seeds. Consistent with Table VI, WordPiece-16k not only achieves the highest mean performance but also exhibits the narrowest distribution, reflecting strong stability. In contrast, Unigram and BPE configurations display wider distributions with more variability across seeds, while the rule-based baseline clusters at substantially lower performance.

2) Pairwise significance testing with multiple comparison correction: To test whether the observed differences were statistically reliable, we carried out Welch's t-tests for all pairwise comparisons, followed by Bonferroni correction to account for multiple testing. The results are summarized in Table VII. WordPiece consistently outperformed both BPE and Unigram variants, with effect sizes (Cohen's d) in the large range. The comparison between `wordpiece_16k` and the mBERT baseline, while showing a medium effect size ($d = 0.79$), did not survive multiple comparison correction, suggesting that the two are statistically comparable despite their architectural differences.

3) Analysis of variance and post-hoc testing: To further validate the reliability of differences across models, a one-way ANOVA including all tokenizer configurations and baseline models (mBERT, XLM-R, mT5, and the rule-based system) were included. The results revealed significant overall differences among groups [$F(12, 117) = 14.83$, $p < 0.001$, $\eta^2 = 0.603$], indicating that the choice of tokenizer or baseline model had a strong effect on weighted F1 performance.

Post-hoc Tukey HSD tests confirmed the following patterns:

- WordPiece tokenizers (8k, 16k, 32k) formed a statistically distinct group, significantly outperforming nearly all BPE and Unigram variants (all $p < 0.05$ after correction).
- WordPiece-16k vs. mBERT showed no statistically significant difference ($p = 0.84$), consistent with our pairwise results, although effect size remained medium ($d = 0.79$).
- WordPiece-16k vs. XLM-R and mT5 also showed no significant differences (all $p > 0.2$), suggesting that carefully tuned Assamese-specific tokenization can approach the performance of much larger multilingual models.
- All neural models (WordPiece, BPE, Unigram, mBERT, XLM-R, mT5) significantly outperformed the **rule-based baseline** ($p < 0.001$), with extremely large effect sizes ($d > 3.0$).

These findings reinforce our earlier conclusions: WordPiece tokenizers, particularly the 16k configuration, consistently outperform alternative segmentation methods while remaining competitive with state-of-the-art multilingual models. Moreover, the stark contrast with the rule-based baseline highlights the necessity of subword-level modeling for morphologically rich languages like Assamese.

D. Comprehensive Linguistic Error Analysis

Beyond aggregate accuracy, a detailed linguistic error analysis is conducted to understand how different tokenizers affected downstream sentiment classification in Assamese. Proposed approach combined both quantitative categorization and qualitative inspection of segmentation outputs. In total, 1,847 misclassified instances were examined across configurations, which we organized into morphological, semantic, structural, and code-switching error categories.

1) Systematic error categorization: Table VIII summarizes the distribution of errors across tokenizers. Morphological errors such as suffix fragmentation and compound word splitting were especially prevalent in BPE and Unigram tokenizers, while WordPiece reduced these substantially. Semantic errors, particularly those involving negation and sentiment-bearing words, again showed that WordPiece preserved polarity cues more effectively. Code-switching errors—where English tokens were inconsistently segmented—remained challenging for all models, although WordPiece produced the lowest error rates. Structural biases were also evident: short sentences ($\leq$words) were disproportionately misclassified across all tokenizers, while long sentences ($\geq$ words) led to degradation in Unigram and BPE variants.

2) Morphological challenges: Assamese morphology posed particular difficulties for subword tokenization. Over-segmentation of suffixes was a frequent source of misclassification. For example, the word "শুনদরতাম" (most beautiful) was tokenized as ["সুনদর", "ত", "ম"] by BPE, breaking the superlative suffix "-তম" into meaningless units. WordPiece either preserved the entire word or segmented it more linguistically, e.g. ["সুনদৰ", "তম"], maintaining

TABLE VII. PAIRWISE STATISTICAL COMPARISONS WITH BONFERRONI CORRECTION

Comparison	**Corrected p**	**Cohen's d**	**Effect Size**	**Significant**
wordpiece_16k vs bpe_8k	0.0207	1.42	Large	✓
wordpiece_16k vs unigram_8k	0.0081	1.58	Large	✓
wordpiece_16k vs unigram_16k	0.0135	1.51	Large	✓
wordpiece_16k vs bpe_32k	0.0279	1.35	Large	✓
wordpiece_16k vs unigram_32k	0.0036	1.73	Large	✓
wordpiece_16k vs mBERT baseline	1.0000	0.79	Medium	✗
wordpiece_16k vs XLM-R baseline	0.8451	0.42	Small–Medium	✗
wordpiece_16k vs mT5 baseline	0.2914	0.61	Medium	✗
wordpiece_16k vs Rule-based baseline	**¡0.0001**	3.25	Huge	✓

TABLE VIII. COMPREHENSIVE ERROR CATEGORIZATION ACROSS TOKENIZERS

Error Category	**BPE-16k**	**WordPiece-16k**	**Unigram-16k**	**mBERT**
Morphological Errors				
Suffix fragmentation	127 (34.2%)	43 (18.7%)	98 (28.9%)	89 (23.1%)
Compound word splitting	89 (24.0%)	31 (13.5%)	76 (22.4%)	67 (17.4%)
Case marker errors	67 (18.1%)	23 (10.0%)	54 (15.9%)	45 (11.7%)
Semantic Errors				
Negation mishandling	78 (21.0%)	29 (12.6%)	67 (19.8%)	52 (13.5%)
Sentiment word fragmentation	92 (24.8%)	31 (13.5%)	71 (20.9%)	48 (12.5%)
Code-switching issues	134 (36.1%)	67 (29.1%)	112 (33.0%)	98 (25.5%)
Structural Errors				
Short text bias ($\leq$5 words)	156 (42.0%)	89 (38.7%)	143 (42.1%)	167 (43.5%)
Long text degradation ($\geq$15 words)	45 (12.1%)	23 (10.0%)	41 (12.1%)	34 (8.9%)
Total Error Count	371	230	339	385

morphological integrity. Unigram showed inconsistent behavior, sometimes aligning with morphemes and other times fragmenting into sub-syllabic pieces.

Compound nouns exhibited similar issues. Table IX illustrates representative examples. Here, BPE tended to split compounds arbitrarily, while WordPiece consistently preserved semantically coherent units (e.g. "রেলগারি" as "rail + vehicle"). Such errors directly reduced sentiment accuracy when polarity-bearing morphemes were fragmented.

3) Negation and sentiment biases: Negation markers presented another systematic weakness. The word "নহয়" (is not) was frequently split into ["ন", "হয়"] by BPE, stripping the sentence of its negative polarity. For instance:

> **Input:** এঅ ছবিখন ভাল নহয় ("This movie is not good.")
> **Gold Label:** Negative
> **Prediction (WordPiece 16k):** Positive

Context-dependent negations such as "একবাৰে ভাল নহয়" ("not good at all") were misclassified as positive 67% of the time under BPE, compared to 23% under WordPiece and 45% under Unigram. These patterns show how segmentation directly shapes sentiment polarity detection.

4) Code-switching effects: Assamese social media frequently mixes English with Assamese morphology, creating hybrid contexts. Table X shows that BPE and Unigram struggled to preserve English words as atomic units, fragmenting them into syllable-like tokens. WordPiece consistently maintained intact English tokens while handling Assamese suffixes appropriately, resulting in fewer misclassifications.

5) Segmentation quality metrics: To complement qualitative analysis, segmentation quality metrics were developed (Table XI). WordPiece achieved higher scores on morpheme preservation and semantic unit coherence, while BPE had the highest over-segmentation score. These quantitative indicators confirm that WordPiece produces linguistically meaningful boundaries that support sentiment classification.

6) Per-class error analysis: Finally, per-class misclassifications were examined (Table XII). Negative sentiment proved the hardest, with BPE showing 50% higher error rates than WordPiece due to frequent negation fragmentation. Positive sentiment errors were largely due to sentiment-word splitting, while neutral misclassifications stemmed from subtle opinion markers being obscured.

V. DISCUSSION

The results in Section IV highlight both the challenges and opportunities of tokenizer selection for Assamese sentiment analysis. The collapse in Experiment 1 (Fig. 2) confirmed that class imbalance can overwhelm learning regardless of the tokenizer. Once class weights were introduced in Experiment 2 (Table V, Fig. 3), clear differences emerged across tokenization strategies. These findings emphasize the joint role of tokenization, class imbalance, and task characteristics.

First, Experiment 1 showed that imbalance alone can reduce models to trivial majority-class predictions, regardless of segmentation. Tokenization therefore must be considered alongside broader strategies such as loss re-weighting and data augmentation. Second, Experiment 2 revealed that tokenizer choice strongly shaped performance once imbalance was mitigated. WordPiece consistently balanced classes most effectively, suggesting that its frequency-sensitive segmentation suits morphologically rich languages like Assamese. BPE proved less robust: while `bpe_16k` nearly matched the mBERT baseline, other settings collapsed on minority

TABLE IX. COMPOUND WORD SEGMENTATION EXAMPLES ACROSS TOKENIZERS

Assamese Word	English Meaning	BPE-16k	WordPiece-16k	Unigram-16k
পাতশালা	school	[“পাত”, “শা”, “লা”]	[“পাতশালা”]	[“পাত”, “শালা”]
ৰেলগাৰি	train	[“ৰেল”, “গা”, “ৰি”]	[“ৰেলগাৰি”]	[“ৰেল”, “গাৰি”]
সকুলঘৰ	school building	[“সকুল”, “ঘ”, “ৰ”]	[“সকুল”, “ঘৰ”]	[“সকু”, “ল”, “ঘৰ”]
হাতিশাল	elephant stable	[“হা”, “তি”, “শাল”]	[“হাতি”, “শাল”]	[“হাতি”, “শা”, “ল”]

TABLE X. ERROR PATTERNS IN CODE-SWITCHED TEXT

Mixed Text Example	Gold Label	BPE	WordPiece	Unigram
“movie টু boring অসিল”	Negative	78%	34%	56%
“এঅ song টু really ভাল”	Positive	23%	12%	31%
“totally disappointed ঃল”	Negative	89%	45%	67%

TABLE XI. SEGMENTATION QUALITY METRICS ACROSS TOKENIZERS

Metric	BPE-16k	WordPiece-16k	Unigram-16k
Over-segmentation Score	0.34	0.21	0.28
Under-segmentation Score	0.08	0.12	0.09
Morpheme Preservation	0.43	0.67	0.51
Semantic Coherence	0.38	0.61	0.45

classes. Unigram, though weaker overall, sometimes recovered rare morphemes (e.g. 0.3346 F1 on `Positive`), showing the value of probabilistic segmentation. Third, the mBERT baseline, despite its large pretrained vocabulary (∼ 120K subwords), failed completely on the `Negative` class. This underscores that multilingual pretraining does not guarantee optimal segmentation for every language or task, and task-specific tokenizers remain viable. Performance also varied with dataset characteristics. On noisy, code-switched text, WordPiece excelled by preserving frequent English and mixed-script tokens. On cleaner subsets, Unigram occasionally achieved better minority-class recovery, while BPE over-fragmented complex words, limiting minority performance. Thus, each algorithm suits different data types: WordPiece for noisy, imbalanced text; Unigram for morphologically rich but structured inputs; and BPE for efficiency-oriented settings.

Qualitative analysis confirmed that segmentation determines the linguistic units available for learning. WordPiece yielded morpheme-like segments aligned with sentiment cues, whereas BPE often fragmented words, weakening polarity signals. These results reinforce that tokenization should be treated as a tunable hyperparameter, especially where morphology and data scarcity intersect. Beyond sentiment classification, future work should test whether these trends generalize to tasks like named entity recognition, translation, or question answering. Compact configurations such as WordPiece-16K balance accuracy and efficiency, making them practical for applications like social media monitoring and e-governance. Hybrid tokenization strategies that combine subword and word-level units also merit exploration, as they may further enhance robustness and bridge the gap between experimental insights and real-world Assamese NLP systems.

VI. LIMITATIONS

Despite these promising findings, several limitations should be acknowledged. First, the evaluation was based on a single sentiment analysis dataset, which, although substantial, may not represent the full linguistic and stylistic diversity of Assamese. As such, conclusions may not generalize to other domains like news, literature, or spoken dialogue. Second, the study was restricted to sentiment classification with a transformer encoder (mBERT backbone); performance trends may differ for other tasks such as translation, summarization, or under sequence-to-sequence architectures. Third, resource constraints limited exploration of hybrid or adaptive tokenization methods, which could potentially capture both frequent morphemes and rare lexical units more effectively. Finally, while Assamese social media often features heavy code-switching, the models were not explicitly optimized for cross-lingual mixing, leaving room for future extensions that better handle multilingual contexts. Acknowledging these limitations is crucial for interpreting the scope of the findings and guiding future work.

VII. CONCLUSION AND FUTURE WORK

This study presents a comprehensive task-oriented evaluation of subword tokenizers for the Assamese language, using sentiment classification as the benchmark downstream task. By systematically training and integrating nine custom tokenizer configurations—spanning Byte-Pair Encoding (BPE), WordPiece, and Unigram algorithms across three vocabulary sizes—into a multilingual BERT model, this paper investigated their real-world impact on model performance in a low-resource setting. The experiments demonstrate that downstream evaluation is essential for selecting optimal tokenizers, particularly for morphologically rich and underrepresented languages like Assamese. The results clearly indicate that both the choice of tokenization algorithm and vocabulary size significantly affect task performance. WordPiece emerged as the most effective algorithm, with the 16K vocabulary configuration achieving the highest weighted F1-score of 0.478. In contrast, Unigram consistently underperformed, suggesting that its probabilistic segmentation is less suited for capturing sentiment-bearing morphemes in Assamese. Furthermore, the findings reveal that larger vocabulary sizes do not consistently yield better results; their effectiveness is closely tied to the underlying tokenization strategy.

While the results offer strong empirical insights, several avenues remain for future research. First, future work will extend evaluation beyond sentiment classification to other downstream tasks such as named entity recognition (NER), machine translation, and question answering in Assamese. Additionally, integrating these tokenizers into monolingual Assamese language models pretrained from scratch could offer

TABLE XII. PER-CLASS ERROR ANALYSIS (WORDPIECE-16K VS BPE-16K)

True Label	WordPiece-16k Errors	BPE-16k Errors	Primary Causes
Positive	89/445 (20.0%)	134/445 (30.1%)	Word fragmentation, intensity markers
Negative	67/234 (28.6%)	112/234 (47.9%)	Negation fragmentation, compound negatives
Neutral	74/1321 (5.6%)	125/1321 (9.5%)	Context misinterpretation, subtle cues

deeper insights into their long-term utility. Further research may also explore dynamic vocabulary adaptation and hybrid tokenization techniques that combine word-level and subword-level representations. Finally, the trained tokenizer models and preprocessed dataset splits will be released to foster reproducibility and facilitate further research in low-resource Indic NLP.

FUNDING

This work was supported by the Deanship of Scientific Research under the Vice Presidency for Graduate Studies and Scientific Research of King Faisal University in Saudi Arabia under Project (KFU253001).

REFERENCES

[1] R. Sennrich, B. Haddow, and A. Birch, "Neural machine translation of rare words with subword units," in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL)*, 2016, pp. 1715–1725.

[2] T. Kudo, "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, pp. 66–75.

[3] M. M. M. Khazani, H. Mohamed, T. M. T. Sembok, N. M. M. Yusop, S. Wani, Y. Gulzar, M. H. M. Halip, S. Marzukhi, and Z. Yunos, "Semantic graph knowledge representation for al-quran verses based on word dependencies," *Malaysian Journal of Computer Science*, pp. 132–153, 2021.

[4] B. Haddow, R. Bawden, A. V. M. Barone, J. Helcl, and A. Birch, "Survey of low-resource machine translation," *Computational Linguistics*, vol. 48, no. 3, pp. 673–732, 2022.

[5] A. Conneau, K. Khandelwal, N. Goyal, V. Chaudhary, G. Wenzek, F. Guzmán, E. Grave, M. Ott, L. Zettlemoyer, and V. Stoyanov, "Unsupervised cross-lingual representation learning at scale," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL)*, 2020, pp. 8440–8451.

[6] S. M. U. Qumar, M. Azim, M. Alkanan, S. M. K. Quadri, M. S. Mir, and Y. Gulzar, "Deep neural architectures for kashmiri-english machine translation," *Scientific Reports*, vol. 15, p. 30014, 2025.

[7] P. Rust, I. Vulić, S. Ruder, and A. Korhonen, "How good is your tokenizer? on the monolingual performance of multilingual language models," *Transactions of the Association for Computational Linguistics*, vol. 9, pp. 667–685, 2021.

[8] T. Kocmi, R. Bawden, O. Bojar, A. Dvorkovich, C. Federmann, M. Fishel, T. Gowda, Y. Graham, R. Grundkiewicz, B. Haddow, R. Knowles, P. Koehn, C. Monz, M. Morishita, M. Nagata, T. Nakazawa, M. Novák, M. Popel, and M. Popović, "Findings of the 2022 conference on machine translation (wmt22)," in *Proceedings of the 7th Conference on Machine Translation (WMT)*, Abu Dhabi, United Arab Emirates (Hybrid), 2022, pp. 1–45.

[9] M. Schuster and K. Nakajima, "Japanese and korean voice search," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2012, pp. 5149–5152.

[10] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," in *Proceedings of NAACL-HLT*, 2019, pp. 4171–4186.

[11] S. Wu and M. Dredze, "Are all languages created equal in multilingual BERT?" in *Proceedings of the 5th Workshop on Representation Learning for NLP*, Online, 2020, pp. 120–130.

[12] Z. Wang, J. Xie, R. Xu, Y. Yang, G. Neubig, and J. G. Carbonell, "Cross-lingual alignment vs. joint training: A comparative study and a simple unified framework," in *Proceedings of the International Conference on Learning Representations (ICLR)*, 2020. [Online]. Available: https://openreview.net/forum?id=S1l-C0NtwS

[13] N. Goyal, C. Gao, V. Chaudhary, P.-J. Chen, G. Wenzek, D. Ju, S. Krishnan, M. Ranzato, F. Guzmán, and A. Fan, "The flores-101 evaluation benchmark for low-resource and multilingual machine translation," *Transactions of the Association for Computational Linguistics*, vol. 10, pp. 522–538, 2022.

[14] G. Ramesh, S. Doddapaneni, A. Bheemaraj, M. Jobanputra, R. A. K., A. Sharma, S. Sahoo, H. Diddee, M. J, D. Kakwani, N. Kumar, A. Pradeep, S. Nagaraj, K. Deepak, V. Raghavan, A. Kunchukuttan, P. Kumar, and M. S. Khapra, "Samanantar: The largest publicly available parallel corpora collection for 11 indic languages," *Transactions of the Association for Computational Linguistics*, vol. 10, pp. 145–162, 2022.

[15] D. Kakwani, A. Kunchukuttan, S. Golla, G. N.C., A. Bhattacharyya, M. M. Khapra, and P. Kumar, "Indicnlpsuite: Monolingual corpora, evaluation benchmarks and pre-trained multilingual language models for indian languages," in *Findings of the Association for Computational Linguistics: EMNLP 2020*. Online: Association for Computational Linguistics, 2020, pp. 4948–4961.

[16] J. Gala, P. A. Chitale, R. A. K., V. Gumma, S. Doddapaneni, A. Kumar, J. Nawale, A. Sujatha, R. Puduppully, V. Raghavan, P. Kumar, M. M. Khapra, R. Dabre, and A. Kunchukuttan, "Indictrans2: Towards high-quality and accessible machine translation models for all 22 scheduled indian languages," *Transactions on Machine Learning Research*, 2023, published December 2023. Reviewed on OpenReview. [Online]. Available: https://openreview.net/forum?id=vfT4YuzAYA

[17] K. Boström and G. Durrett, "Byte pair encoding is suboptimal for language model pretraining," in *Proceedings of EMNLP*, 2020, pp. 4617–4624.

[18] I. Provilkov, D. Emelianenko, and E. Voita, "Bpe-dropout: Simple and effective subword regularization," in *Proceedings of ACL*, 2020, pp. 1882–1892.

[19] L. Xue, N. Constant, A. Roberts, M. Kale, R. Al-Rfou, A. Siddhant, A. Barua, and C. Raffel, "mt5: A massively multilingual pre-trained text-to-text transformer," in *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*. Online: Association for Computational Linguistics, 2021, pp. 483–498.

[20] T. Pires, E. Schlinger, and D. Garrette, "How multilingual is multilingual BERT?" in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL)*. Florence, Italy: Association for Computational Linguistics, 2019, pp. 4996–5001.

[21] G. Aguilar, S. Kar, and T. Solorio, "Lince: A centralized benchmark for linguistic code-switching evaluation," in *Proceedings of the Twelfth Language Resources and Evaluation Conference (LREC)*. Marseille, France: European Language Resources Association, 2020, pp. 1803–1813.

[22] G. I. Winata, A. Madotto, C.-S. Wu, and P. Fung, "Code-switching language modeling using syntax-aware multi-task learning," in *Proceedings of the Third Workshop on Computational Approaches to Linguistic Code-Switching*. Melbourne, Australia: Association for Computational Linguistics, 2018, pp. 62–67.

[23] M. B. Swidan, A. A. Alwan, S. Turaev, H. Ibrahim, A. Z. Abualkishik, and Y. Gulzar, "Skyline queries computation on crowdsourced-enabled incomplete database," *IEEE Access*, vol. 8, pp. 53 658–53 670, 2020.

[24] Y. Gulzar, A. A. Alwan, H. Ibrahim, and Q. Xin, “D-sky: A framework for processing skyline queries in a dynamic and incomplete database,” in *Proceedings of the 20th International Conference on Information Integration and Web-based Applications and Services (iiWAS)*, Yogyakarta, Indonesia, 2018, pp. 27–38.

[25] M. B. Swidan, A. A. Alwan, S. Turaev, and Y. Gulzar, “A model for processing skyline queries in crowd-sourced databases,” *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 10, no. 2, pp. 798–806, 2018.

[26] M. Sokolova and G. Lapalme, “A systematic analysis of performance measures for classification tasks,” *Information Processing and Management*, vol. 45, no. 4, pp. 427–437, 2009.

[27] D. M. W. Powers, “Evaluation: From precision, recall and f-measure to ROC, informedness, markedness and correlation,” *Journal of Machine Learning Technologies*, vol. 2, no. 1, pp. 37–63, 2011.

[28] Z. C. Lipton, C. Elkan, and B. Naryanaswamy, “Optimal thresholding of classifiers to maximize F1 measure,” in *Machine Learning and Knowledge Discovery in Databases*. Springer, 2014, pp. 225–239.

[29] N. Goyal, J. Du, M. Ott, G. Anantharaman, and A. Conneau, “Larger-scale transformers for multilingual masked language modeling,” *arXiv preprint arXiv:2105.00572*, 2021.