IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 17 May 2026, accepted 3 June 2026, date of publication 9 June 2026, date of current version 22 June 2026.

Digital Object Identifier 10.1109/ACCESS.2026.3701520

RESEARCH ARTICLE

Efficient Monolingual Pretraining in Low-Resource Settings Through Morphology-Aware Tokenization, Principled Corpus Denoising, and Benchmark-Driven Evaluation

NAHID HOSSAIN[1] **AND MD. FAISAL KABIR**[2], **(Member, IEEE)**
[1]United International University, Dhaka 1212, Bangladesh
[2]Pennsylvania State University, Middletown, PA 17057, USA

Corresponding author: Nahid Hossain (nahid@cse.uiu.ac.bd)

This work was supported by the Institute for Advanced Research Publication Grant of United International University under Grant IAR-2026-Pub-027.

ABSTRACT Training effective language models for morphologically complex, low-resource languages in resource-constrained environments is hampered by four compounding bottlenecks: the representational dilution of massively multilingual pretraining, the inadequate subword segmentation of tokenizers not trained on language-specific data, the substantial noise in existing pretraining corpora, and the absence of a standardized evaluation benchmark. We address all four deficiencies in a unified framework, taking Bengali as our target language. First, we introduce *B-CORE*, a rigorously curated 52GB Bengali corpus of 16.5M documents (4.32B tokens), built via a reproducible pipeline of quality filtering, expert validation, and cross-corpus deduplication that reduces volume by 22.4% relative to raw data. Second, we design morphology-aware WordPiece tokenizers (30.5K and 120K vocabularies) trained exclusively on Bengali, attaining 96.9% script coverage versus 56–60% for multilingual baselines, with 34–51% fewer tokens per word. Third, we introduce *BLUGE*, a balanced, expert-validated, contamination-audited benchmark spanning 7 tasks with 536K samples. Fourth, we pretrain *BnLM*, a suite of three compact Bengali-specific encoder models (66–135M parameters) on the MLM objective in different settings, which establish new state-of-the-art results on *BLUGE* and four other multilingual and Indic benchmarks, while requiring 44–91% fewer FLOPs and providing 2× faster inference than larger baselines. A controlled 2×2 factorial analysis isolates tokenizer specialization as the dominant factor, showing that the bottleneck in low-resource NLP is not model scale but the principled alignment of corpus quality, tokenization, and evaluation with the target language's properties.

INDEX TERMS B-CORE, BLUGE, BnLM, corpus, evaluation benchmark, pretrained models.

I. INTRODUCTION

The prohibitive computational and financial costs of training large-scale language models have made continued pretraining from existing multilingual base models the prevailing strategy in NLP, particularly in resource-constrained settings [1]. While multilingual NLU models such as mBERT [2] and XLM-R [3] exhibit strong cross-lingual transfer, they systematically underperform on low-resource or morphologically complex languages like Bengali [4]. This gap is not merely a consequence of data scarcity; it reflects deeper structural mismatches between multilingual modeling choices and the linguistic properties of underrepresented languages.

We identify four compounding failure modes that collectively explain this performance gap. First, multilingual models such as mBERT and XLM-R are trained jointly across

The associate editor coordinating the review of this manuscript and approving it for publication was Mohammad J. Abdel-Rahman.

100+ typologically diverse languages, forcing them to share limited representational capacity across the entire typological spectrum; Bengali, an agglutinative language with rich inflectional morphology, receives insufficient exposure to its language-specific morphological, syntactic, and lexical patterns as a direct consequence. Second, and critically, the tokenization bottleneck extends beyond multilingual models: even Bengali-specific and Indic language models that otherwise depart from multilingual pretraining continue to rely on off-the-shelf tokenizers poorly aligned with Bengali script. As we show empirically, such tokenizers achieve only 56–60% script coverage and produce 20–26% more tokens per word than a language-specific tokenizer, directly inflating computational cost, fragmenting morphological structure, and degrading both pretraining signal and downstream task performance. Third, existing Bengali pretraining corpora suffer from substantial noise, inconsistent normalization, and inadequate deduplication, undermining the quality of pretraining regardless of model architecture. Fourth, there is no standardized, high-quality evaluation benchmark for Bengali, making rigorous and reproducible comparisons across systems infeasible and obscuring the true extent of these modeling failures.

This paper addresses all four deficiencies, namely corpus quality, tokenization efficiency, evaluation rigor, and the absence of a compact yet high-performing Bengali-specific pretrained model, in a unified, principled framework. To this end, we contribute a large-scale curated corpus spanning diverse sources including mC4 [5], Wikipedia, NHMono01 [6], Sangraha [7], CulturaX [8], Varta [9], and our own crawls of Bangladeshi and Indian magazines, storybooks, and novels. Our data pipeline integrates orthographic normalization, expert validation, cross-corpus deduplication, and context-aware segmentation optimized for transformer pretraining, reducing corpus volume by 22.4% relative to raw source data through quality filtering and deduplication. Building on this foundation, we train language-specific tokenizers and pretrain a suite of compact encoder models that substantially outperform multilingual counterparts while requiring far fewer FLOPs. We make the following contributions:

1) We introduce ***B-CORE*** (**B**engali **C**ontext-aware **O**ptimized and **R**efined **E**ntities), a large-scale, rigorously curated Bengali monolingual corpus comprising 16.5 million documents (4.32 billion tokens; 52GB). To our knowledge, *B-CORE* is among the largest and most carefully curated Bengali pretraining corpus, constructed via a reproducible multi-stage pipeline that achieves a 22.4% reduction in corpus volume relative to raw source data through systematic quality filtering and cross-corpus deduplication.
2) We introduce ***BLUGE*** (**B**engali **L**anguage **U**nderstandin**G** **E**valuation), a balanced benchmark spanning 7 linguistically diverse tasks with 536K samples. *BLUGE* addresses critical shortcomings of prior Bengali benchmarks, including class imbalance, annotation inconsistencies, and non-standardized splits, enabling reliable and reproducible evaluation.
3) We introduce morphology-aware WordPiece tokenizers (30.5K and 120K vocabulary sizes) trained exclusively on Bengali text, achieving 96.9% script coverage compared to 56–60% for multilingual baselines, while reducing tokens per word by ∼34–51% and processing text ∼71–96% faster. This reflects a fundamental alignment with Bengali's agglutinative morphology that translates directly into more efficient pretraining and inference.
4) We introduce ***BnLM*** (**Ben**gali **L**anguage **M**odel), a suite of three efficient, Bengali-specific pretrained language models for low-resource settings, *BnLM-F* (135M parameters), *BnLM-M* (135M parameters), and *BnLM-C* (66M parameters), pretrained from scratch on *B-CORE* using the MLM objective [2]. Despite their compact size, *BnLM* models establish new state-of-the-art results across *BLUGE* and 4 prominent cross-lingual and Indic benchmarks [10], [11], [12], [13], while requiring 44–91% fewer FLOPs and achieving 2× faster inference than competitive multilingual baselines.
5) Through systematic empirical analyses, we examine the individual contributions of corpus curation, tokenization design, model scale, and computational efficiency to downstream performance. Our results provide strong empirical evidence that principled language-specific pretraining yields superior performance–efficiency trade-offs over substantially larger multilingual models, offering actionable insights for low-resource NLP beyond Bengali.

Together, these contributions demonstrate that the bottleneck in low-resource NLP is not model scale, but rather the principled alignment of corpus quality, tokenization, model architecture, and evaluation infrastructure with the linguistic properties of the target language.

II. RELATED WORK

A. CORPORA

Existing Bengali monolingual corpora vary significantly in scale and composition. Bangla2B+ [13] contains 7.18M articles, making it one of the most Bengali-specific prominent resources. Other monolingual corpora include Vacaspati [14] with approximately 0.3M articles, KUMono [15] with 1.3M articles. Additionally, several large-scale multilingual datasets include substantial Bengali content: IndicCorp [11] with 836M tokens (approximately 3.89M articles), and IndicCorpV2 [12] with 926M tokens (approximately 4.3M articles). Sangraha [7] encompasses 22 languages with 9.53M non-synthetic Bengali articles sourced from mC4, Wikipedia, and other resources; CulturaX [8] covers 167 languages with 12.4M Bengali articles from mC4; and Varta [9] provides news articles across 14 Indic languages, including 2.25M

Bengali articles. In contrast, our *B-CORE*, with 16.5M articles, represents one of the largest Bengali corpora to date and employs a multi-stage filtering pipeline for systematic noise reduction and quality assurance.

B. EVALUATION BENCHMARKS

The BLUB [13] evaluation benchmark comprises 4 NLU tasks: Sentiment Classification, Language Inference, Named Entity Recognition, and Question Answering. IndicGLUE [11], a multilingual evaluation benchmark for Indic languages, includes 3 notable Bengali-specific datasets: WIKI-NER, SNA, and WSTP. XTREME [10] covers over 40 languages; however, for Bengali, it provides only the PAN-X (a NER task) dataset for language understanding tasks. IndicXTREME [12] offers evaluation datasets for 9 tasks across 20 Indic languages, including 6 Bengali-specific tasks that overlap with the aforementioned benchmarks. However, these benchmarks suffer from inconsistent train-test splits, class imbalances, and lack comprehensive coverage of diverse NLU tasks, necessitating a unified evaluation framework with standardized data quality and task diversity.

C. PRETRAINED LANGUAGE MODELS

BERT [2] employs bidirectional transformers [16] with Masked Language Modeling (MLM) and Next Sentence Prediction (NSP) objectives, masking 15% of tokens for prediction. DistilBERT [17] distills BERT through knowledge distillation [18], retaining most language understanding capabilities while omitting NSP for computational efficiency. Multilingual BERT (mBERT) extends BERT to over 100 languages including Bengali using Wikipedia data. XLM-R [3] follows the RoBERTa [19] architecture, removing NSP and optimizing training on large-scale multilingual data, though computationally intensive. IndicBERT [11] targets 11 Indian languages including Bengali, built on ALBERT [20] with parameter sharing for enhanced training efficiency. BanglaBERT [13] leverages the ELECTRA [21] architecture with Replaced Token Detection (RTD), trained on 27.5GB from 110 Bengali websites. SahajBERT [22] is an 18M-parameter ALBERT-large model finetuned on Bengali Wikipedia and OSCAR [23]. VĀC-BERT [14] is a compact 17M-parameter model trained on the 115.1M-token Vacaspati corpus, achieving competitive performance despite its size. Despite these advances, existing Bengali models either rely on limited monolingual data or suffer from multilingual dilution effects, leaving a gap for models trained on large-scale, rigorously cleaned Bengali corpora that can achieve state-of-the-art performance across diverse NLU tasks.

III. THE B-CORE CORPUS: MULTI-STAGE CURATION AND DEDUPLICATION

In this section, we detail the creation of our Bengali monolingual corpus *B-CORE*, which serves as the foundation for our model pretraining (see Appendix C-A for a schematic overview).

A. DATA ACQUISITION AND DIVERSITY

The performance of language models is fundamentally determined by both the quantity and quality of training data, with models trained on large, high-quality datasets demonstrating superior performance [24]. To address this, we propose *B-CORE*, a large-scale rigorously cleaned Bengali monolingual corpus created by merging, deduplicating, and processing data from existing prominent datasets, augmented with a curated collection of magazines, books, and novels. The sources of the corpus are as follows:

- Wikipedia[1]: All Wikipedia Bengali articles up to July 2025, collected from the Wikipedia database dump.
- mC4 [5]: A subset of the multilingual C4 corpus that contains all available Bengali articles. This subset includes scraped plain text data from websites covering diverse domains such as news, academic papers, and publicly accessible web content, ensuring linguistic variety for training models.
- NHMono01 [6]: A moderately-sized, clean, and curated corpus of Bengali news articles sourced from various reputable Bengali-language newspapers, providing high-quality texts.
- Sangraha [7]: A large-scale collection of corpus covering 22 languages including 9.53M non-synthetic Bengali articles from the mC4, Wikipedia and other overlapping sources.
- CulturaX [8]: A multilingual dataset covering 167 languages, with 12.4M Bengali articles from mC4.
- Varta [9]: A collection of news articles covering 14 different Indic languages including 2.25M Bengali articles.
- Custom: A custom corpus scraped using BeautifulSoup,[2] incorporating texts from carefully selected renowned Bangladeshi and Indian magazines, and an extensive collection of Bengali literature, such as storybooks and novels, making it familiar with everyday conversational dialogues as well.

All sources consist of publicly accessible content, with existing datasets (mC4, Sangraha, etc.) used under their respective release terms and custom data collected from publicly available websites following standard web scraping practices in NLP research [25], [26]. *B-CORE* is released strictly for non-commercial research purposes, and we will honor any content removal requests from rights holders. Specifically, mC4 is distributed under the Open Data Commons Attribution License (ODC-BY); Sangraha and CulturaX are released under CC BY 4.0; and Varta is released under CC BY-NC 4.0, consistent with our non-commercial research release of *B-CORE*. Custom web crawling adhered to each target site's `robots.txt` directives, enforced

[1] https://dumps.wikimedia.org/bnwiki/
[2] https://pypi.org/project/beautifulsoup4/

crawl delays of no less than 5 seconds per domain, and excluded paywalled, subscription-gated, or otherwise access-restricted content. Our custom scraped material constitutes a small fraction of *B-CORE*, with the majority drawn from established, openly licensed research datasets.

Due to substantial duplication across sources, the compiled corpus underwent rigorous deduplication and cleaning. The final dataset comprises 16.5 million segmented articles (52 GB uncompressed, compressed to 19.3 GB) distributed across 10 Parquet files, containing 4.32 billion tokens using the *BnLM-C* tokenizer and 3.93 billion tokens using the *BnLM-M*/*BnLM-F* tokenizers. While we could not directly incorporate Bangla2B+ (7.18M samples) due to its unavailability, our *B-CORE* includes data from 101 of the 110 web sources utilized in Bangla2B+, covering approximately 91.81% of its content.[3]

B. CORPUS PROCESSING AND CURATION PIPELINE

In brief, our pipeline applies four sequential operations to the merged raw corpus: (i) document-level quality filtering, (ii) text normalization, (iii) cross-corpus exact deduplication, and (iv) context-aware segmentation. Given the heterogeneous nature of our source corpora $\mathcal{C} = \{\mathcal{C}_{\text{Wiki}}, \mathcal{C}_{\text{NHMono01}}, \mathcal{C}_{\text{mC4}}, \mathcal{C}_{\text{Sangraha}}, \mathcal{C}_{\text{CulturaX}}, \mathcal{C}_{\text{Varta}}, \mathcal{C}_{\text{Custom}}\}$ and the substantial duplication inherent in web-crawled datasets, we implement a systematic pipeline encompassing quality filtering, text normalization, and cross-corpus deduplication to ensure data quality. While $\mathcal{C}_{\text{Wiki}}$, $\mathcal{C}_{\text{NHMono01}}$, $\mathcal{C}_{\text{Sangraha}}$, $\mathcal{C}_{\text{CulturaX}}$, and $\mathcal{C}_{\text{Varta}}$ exhibit acceptable quality metrics, $\mathcal{C}_{\text{mC4}}$ and $\mathcal{C}_{\text{Custom}}$ contain substantial noise including malformed documents, anomalous brevity, and content from unreliable sources, factors empirically shown to degrade model performance [27]. Our complete processing pipeline yields a compression ratio of $\gamma = \frac{|\mathcal{D}_{\text{final}}|}{|\mathcal{D}_{\text{raw}}|} = \frac{52\text{ GB}}{67\text{ GB}} \approx 0.776$, representing a 22.4% reduction in corpus volume through quality filtering, normalization, and cross-corpus deduplication. We note that this figure reflects document-level size reduction arising from deduplication and content filtering; it does not constitute a measured orthographic or semantic error rate. No large language model was involved at any stage of the *B-CORE* construction pipeline: all denoising operations are rule-based and deterministic.

1) QUALITY FILTERING AND TEXT NORMALIZATION

We define a document retention function $f : \mathcal{D} \to \{0, 1\}$ based on multiple quality criteria to filter low-quality content:

$$f(d) = \mathbb{1}_{|\text{tokens}(d)| \geq \tau_{\text{len}}} \cdot \mathbb{1}_{s(d) \notin \mathcal{S}_{\text{block}}} \cdot \mathbb{1}_{\rho_{\text{bn}}(d) \geq \theta_{\text{lang}}} \quad (1)$$

where $d \in \mathcal{D}$ represents a document, tokens$(\cdot)$ denotes the tokenization function, $\tau_{\text{len}} = 200$ is the minimum token threshold, $s(d)$ extracts the source domain, $\mathcal{S}_{\text{block}}$ is the set of blacklisted unreliable sources, and $\rho_{\text{bn}}(d)$ computes the Bengali language ratio using Bangla-Python[4] with threshold θ_{lang}.

For documents satisfying the retention criteria, we apply a sequence of text normalization transformations $\mathcal{T} = \{T_1, T_2, \ldots, T_k\}$:

$$d' = T_k(T_{k-1}(\cdots T_1(d) \cdots)) \quad (2)$$

where individual transformations include: T_{ws} for whitespace and newline regularization via NLTK[5]; T_{html} for HTML tag removal using BeautifulSoup; T_{boiler} for boilerplate content elimination (copyright notices, publisher metadata); T_{enc} for character encoding conversion from legacy Bijoy ASCII to UTF-8 with NFC normalization; and T_{punct} for punctuation and quotation mark standardization.

2) CROSS-CORPUS DEDUPLICATION

To eliminate redundant documents across all corpora, we employ exact deduplication based on content hashing. This step is critical as web-crawled datasets often contain substantial overlap, with the same content appearing across multiple sources or being republished on different platforms. We construct the final deduplicated corpus as:

$$\mathcal{D}_{\text{final}} = \left\{ d_i \in \mathcal{D} \;\middle|\; \nexists j < i : h(d_j) = h(d_i) \right\} \quad (3)$$

where $h(\cdot)$ is a collision-resistant hash function (SHA-256) applied to the normalized document content. This approach ensures that documents identical at the character level are retained only once, regardless of their source corpus, thereby preventing training data redundancy that could lead to overfitting or biased model behavior.

C. CONTEXT-AWARE SEGMENTATION

B-CORE contains documents with highly variable lengths (364 to 160,113 tokens). To optimize training efficiency while preserving contextual coherence within the $L_{\max} = 512$ token constraint, we partition each document's token sequence $\mathbf{t} = (t_1, t_2, \ldots, t_n)$ into segments $\{\mathbf{s}_1, \mathbf{s}_2, \ldots, \mathbf{s}_m\}$ at sentence boundaries using $\text{split}(\mathbf{t}, k) = \arg\min_{i \in \mathcal{B}} |i - k|$ where $k = 512$ and $\mathcal{B}$ denotes sentence boundary indices. To maintain discourse coherence, consecutive segments $\mathbf{s}_i$ and $\mathbf{s}_{i+1}$ overlap by two complete sentences: $\mathbf{s}'_{i+1} = \text{concat}(\text{last}_2(\mathbf{s}_i), \mathbf{s}_{i+1})$ subject to $|\mathbf{s}'_{i+1}| \leq L_{\max}$, where $\text{last}_2(\cdot)$ extracts the final two sentences. This sentence-level overlap preserves semantic continuity without fragmenting syntactic units. Following segmentation, we detokenize subword units, normalize punctuation, and serialize segments in Apache Parquet format for pre-training.

IV. THE *BLUGE* BENCHMARK: STANDARDIZED EVALUATION FOR BENGALI NLU

We introduce *BLUGE*, a benchmark for evaluating general language understanding in Bengali across 7 NLP tasks (see Appendix C-B for a structural overview). Motivated by the

[3]Calculated based on the data sources listed in the BanglaBERT manuscript.

[4]https://pypi.org/project/bangla-python/

[5]https://www.nltk.org/

absence of high-quality Bengali evaluation resources with balanced classes and splits, *BLUGE* integrates rigorously refined versions of existing public corpora with our newly introduced samples. Unlike prior Bengali benchmarks, which suffer from inconsistent splits (BLUB [13]), limited task diversity (XTREME provides only PAN-X for Bengali), or class imbalance (IndicGLUE [11]), *BLUGE* is, to our knowledge, the first Bengali NLU benchmark offering balanced 80-10-10 splits, expert-validated annotation correction, and comprehensive 7-task coverage within a unified evaluation framework. Dataset statistics are provided in Table 1.

TABLE 1. Dataset statistics for the seven core NLP tasks in *BLUGE*, showing sample counts for training, validation, and test splits.

Task	Total	Train	Dev	Test
NER	14,476	11,580	1,447	1,449
TSC	22,296	17,836	2,230	2,230
NCC	62,300	49,840	6,230	6,230
BLI	128,458	102,767	12,845	12,846
WSTP	49,506	39,604	4,951	4,951
BCLAP	130,890	104,713	13,088	13,089
BCLAU	128,498	102,799	12,849	12,850

A. NAMED ENTITY RECOGNITION (NER)

This task identifies named entities (persons, organizations, locations, objects) using the IOB tagging scheme with nine tags: O, B-PER, I-PER, B-ORG, I-ORG, B-LOC, I-LOC, B-OBJ, and I-OBJ. We construct our dataset by merging the PAN-X.bn subset from XTREME [10], datasets from [28] and [28], and Naamapadam from IndicXTREME [12]. After removing redundant entries (62.27%), non-annotated samples, and sentences with fewer than 3 tokens, we obtain 14,476 unique samples.

B. TERNARY SENTIMENT CLASSIFICATION (TSC)

This task classifies texts into three sentiment labels: neutral (0), positive (1), and negative (2). Merging corpora from [29] and [30], we address class imbalance through targeted generation and use GPT-4-turbo [31] to propose corrections for orthographic errors present in 7.23% of samples (see Appendix B for the exact prompt used). We emphasize that GPT-4-turbo was used exclusively for orthographic correction and played no role in class label assignment. GPT-4-turbo functions solely as a proposal mechanism: each proposed correction was independently reviewed and either accepted, revised, or rejected by two native Bengali linguists (κ =0.87), with disagreements resolved through discussion with a third linguist. Of the GPT-4-turbo proposals submitted for human review, 76.4% were accepted without modification, 15.8% were revised by the reviewers before acceptance, and 7.8% were rejected and reverted to the original text. Final semantic preservation relative to original texts was verified at 98.9%. Sentences with fewer than 4 or more than 20 words are excluded. To address severe class imbalance in the merged corpus, we manually collected and annotated 1,247 samples across underrepresented sentiment classes, validated by the same expert annotators (κ =0.89).

C. NEWS CATEGORY CLASSIFICATION (NCC)

This task classifies news articles into seven categories: Business (0), Technology (1), Crime (2), Entertainment (3), International Affairs (4), Sports (5), and Lifestyle (6). We constructed this dataset from scratch using sources outside *B-CORE*. The dataset contains 8,900 samples per category (62,300 total). Annotation was automated by extracting category labels from article URLs, simplifying the labeling process. For each URL u_i with article a_i, we extract label $L(u_i)$ and map to $N(L(u_i))$, yielding $\mathcal{D} = \{(a_i, N(L(u_i)), L(u_i))\}$.

D. BENGALI LANGUAGE INFERENCE (BLI)

This task classifies premise-hypothesis relationships into Entailment (0), Neutral (1), or Contradiction (2). We use XNLI_bn [13], excluding samples with fewer than 5 or more than 14 words and employing GPT-4-turbo to propose orthographic corrections for 3.47% of samples exhibiting surface-level errors (see Appendix B for the exact prompt used). As with TSC, GPT-4-turbo was used exclusively for orthographic correction and played no role in premise-hypothesis label assignment. Each proposed correction was independently reviewed and either accepted, revised, or rejected by three native Bengali linguists with expertise in natural language inference (κ =0.91), with disagreements adjudicated through majority voting. Of the GPT-4-turbo proposals submitted for human review, 81.7% were accepted without modification, 12.6% were revised by the reviewers before acceptance, and 5.7% were rejected and reverted to the original XNLI_bn text. Semantic fidelity relative to original texts was confirmed at 99.2% after human review. Additionally, we manually created 312 premise-hypothesis pairs to improve representation of underrepresented label categories, all validated through the same annotation protocol.

E. WIKIPEDIA SECTION-TITLE PREDICTION (WSTP)

This task predicts Wikipedia section titles from 4 options using IndicGLUE WSTP.bn [11], removing sections with substantial English text, extraneous quotes, and samples exceeding 512 tokens, retaining 83.2% of the original dataset. Beyond filtering, we augmented underrepresented section categories with 1,843 newly extracted and manually validated section-content pairs (κ =0.91), achieving balanced class distribution.

F. BENGALI CORPUS OF LINGUISTIC ACCEPTABILITY - PAIRED (BCLAP)

This binary classification task evaluates a model's understanding of grammatical correctness in Bengali by assessing the grammatical acceptability of sentences. We construct

the BCLAP dataset using the BanglaGEC corpus [32], which encompasses 10 types of Bengali grammatical errors, including verb inflection, speech, homonyms, misplaced punctuation, subject change, missing subjects, and three types of verb deletion (list, VAUX, and VM). BCLAP pairs each ungrammatical sentence with its corrected version, labeling incorrect sentences as 0 (unacceptable) and correct sentences as 1 (acceptable). Formally, BCLAP $= \{(x, y) \mid x \in \{s_i, s_i'\}, \quad y \in \{0, 1\}, \quad y = 0 \text{ if } x = s_i, \quad y = 1 \text{ if } x = s_i'\}$ where s_i denotes ungrammatical sentences and s_i' their corrected counterparts from BanglaGEC. We added an additional column to BCLAP that specifies error types and preprocessed the sentences by removing extra whitespace and appending end markers, except where the error type is 'Misplaced Punctuation.'

G. BENGALI CORPUS OF LINGUISTIC ACCEPTABILITY - UNPAIRED (BCLAU)

The BCLAU dataset complements BCLAP by offering a binary classification task in a non-paired format. Unlike BCLAP, which contains both correct and incorrect versions of the same sentence, BCLAU consists of independently sampled grammatically acceptable and unacceptable sentences. The BCLAU and BCLAP datasets comprise mutually exclusive samples, each designed to facilitate distinct assessment strategies without shared sentence instances. This unpaired format enables broader grammatical assessment by incorporating a diverse range of acceptable sentences that are not directly linked to specific error types.

V. THE BnLM MODELS: MORPHOLOGY-AWARE TOKENIZATION AND PRETRAINING

We describe the methodology and experimental setup for *BnLM* (see Appendix C-C for an architectural overview). To achieve efficiency and high performance, we use DistilBERT-base for *BnLM-C* and DistilBERT-multilingual for *BnLM-F* and *BnLM-M*. Knowledge distillation enables DistilBERT to retain BERT's performance while improving efficiency, making it suitable for resource-constrained deployment. The architecture comprises $L = 6$ layers, hidden size $H = 768$, and $A = 12$ self-attention heads.

A. BENGALI-SPECIFIC TOKENIZER TRAINING

We trained two custom Bengali WordPiece tokenizers with vocabulary sizes of 30.5K and 120K on the B-CORE corpus to replace the original tokenizers in DB-base and mDB-base during BnLM pretraining. The larger 120K vocabulary preserves Bengali's compositional morphological structure, while the smaller 30.5K configuration enables compact, resource-efficient deployment. The pipeline employed whitespace-based pre-tokenization followed by WordPiece subword segmentation with standard BERT special tokens. Training used the HuggingFace Tokenizers library with iterative processing over concatenated corpus segments. The resulting tokenizers were converted to PreTrainedTokenizerFast format for integration with the Transformers library, enabling direct replacement of the multilingual tokenizer while maintaining BERT architectural compatibility. This language-specific approach significantly improves morphological coverage and reduces vocabulary fragmentation for Bengali's agglutinative characteristics, as shown in Table 2 comparing against 9 prominent tokenizers.

B. PRETRAINING OBJECTIVE

All *BnLM* variants are pretrained using masked language modeling (MLM) without next sentence prediction (NSP). We employ the Hugging Face Transformers data collator with our custom Bengali tokenizers. For *BnLM-F* and *BnLM-C*, 15% of tokens are randomly masked, while *BnLM-M* uses 20%. Of masked tokens, 80% are replaced with `[MASK]`, 10% with random tokens for input noise, and 10% remain unchanged. This strategy encourages contextual understanding over masked token overfitting, learning robust Bengali representations.

C. PRETRAINING SETUP

We train *BnLM* in three configurations to evaluate performance and examine the effect of custom tokenizers with large (120K) and small (30.5K) vocabulary sizes for Bengali. Training uses 4 Nvidia RTX 3090 24GB GPUs over one month. All models use AdamW optimizer [33], GELU activation [34], and process multi-sentence inputs with maximum length 512 tokens. Hyperparameters are determined through pilot studies, established research, and computational constraints.

BnLM-F, built on DistilBERT-multilingual, uses our custom Wordpiece tokenizer with 120K. It trains for 2M steps with batch size 64, learning rate 5e-5, weight decay 0.01, 94,460 warmup steps, 15% MLM masking, and evaluation every 50K steps without mixed-precision.

BnLM-M, also based on DistilBERT-multilingual with the same tokenizer, trains for 1.5M steps with batch size 128, learning rate 3e-5, weight decay 0.02, 47,230 warmup steps, 20% MLM masking, and evaluation every 40K steps with mixed-precision training.

BnLM-C, built on DistilBERT-base, uses our custom Wordpiece tokenizer with 30.5K vocabulary. It trains for 1M steps with batch size 256, learning rate 2e-5, weight decay 0.01, 24,000 warmup steps, 15% MLM masking, and evaluation every 30.5K steps with mixed-precision training.

D. FINETUNING

Following pretraining, we fine-tune all models independently on each of *BLUGE*'s seven tasks using task-specific train-validation splits. Training runs in parallel on 24GB Nvidia 3090 GPUs for $E = 15$ epochs (NER, TSC) or $E = 20$ epochs (NCC, BLI, WSTP, BCLAP, BCLAU). We apply early stopping with patience 3, selecting models minimizing validation loss. Finetuning hyperparameters are selected via grid search over batch size $b \in \{32, 64, 128\}$ and learning rate

TABLE 2. Tokenization Performance of *BnLM* Tokenizers vs. Existing Methods Across All Tasks in the *BLUGE* Benchmark. All metrics are deterministic, except *Tokenization Speed*, which is reported as the mean over 5 runs to account for system-level variance; its standard deviation is omitted due to space constraints.

Metric	DB-base	mDB-base	B-base	mB-base	InB	BnB	XLM-R	SjB	VAC-B	*BnLM-F* *BnLM-M*	*BnLM-C*
Bengali Script Coverage (%) ↑	28.125	56.250	28.125	56.250	56.250	57.812	60.156	95.312	67.188	96.875	96.875
Bengali Token Representation (%) ↑	0.118	0.443	0.118	0.443	3.324	73.625	0.343	17.572	78.274	83.796	80.223
Text Coverage (%) ↑	91.98	95.72	91.98	95.72	100	95.7	100	100	37.76	100	100
OOV Rate (%) ↓	8.02	4.28	8.02	4.28	0	4.3	0	0	62.23	0	0
Vocab Size	30522	119547	30522	119547	200000	32000	250002	32000	50000	120000	30522
Avg. Sequence Length ↓	293.07	179.69	293.07	179.69	154.42	90.36	138.68	102.83	81.41	81.45	88.49
Avg. Tokens per Word ↓	4.9	3.09	4.9	3.09	2.64	2.04	2.48	2.02	1.52	1.51	1.64
Word Fragmentation Rate (%) ↓	93.28	74.65	93.28	74.65	78.53	38.48	61.81	42.84	30.63	31.63	37.32
Compression Ratio ↑	0.21	0.33	0.21	0.33	0.4	0.66	0.44	0.60	0.74	0.73	0.68
Avg. Token / Sentence↓	52.36	33.14	52.36	33.14	28.49	18.81	26.83	22.02	16.41	15.32	16.66
Single-Token Words (%) ↑	6.72	25.35	6.72	25.35	21.47	61.52	38.19	57.16	69.37	67.97	62.68
Tokenization Speed (s/sec) ↑	9492.05	12401.39	9424.59	12186.06	11273.11	15494.91	11145.30	10808.84	16877.69	21798.31	21186.94
Avg. Unique Tokens Utilized ↑	450.14	1694.71	450.14	1694.71	1542.14	21779.43	3095.14	16418.57	12564.29	53486.71	22284.71

$\eta \in \{2e\text{-}5, 3e\text{-}5, 5e\text{-}5\}$, yielding 9 configurations per task; the configuration minimising validation loss at epoch end is retained. All remaining parameters match the pretraining configuration described in Subsection V-C. This grid was applied identically across all models and all tasks to ensure fair comparison.

VI. EVALUATION AND ANALYSIS

To assess pretrained models and tokenizers' performance, evaluations are conducted using established Bengali benchmarks (XTREME, IndicGLUE, BLUB) and the proposed *BLUGE* benchmark. Results are compared against baseline models: DistilBERT-base (DB-base), DistilBERT-Multilingual-base (mDB-base), BERT-base (B-base), BERT-base-Multilingual (mB-base), IndicBERT (InB), BanglaBERT (BnB), XLM-R-base (XLM-R), sahajBERT (SjB), and VAC-BERT (VAC-B). These abbreviations are used throughout.

A. TOKENIZATION PERFORMANCE ANALYSIS

Tokenization critically impacts model performance for underrepresented languages, where multilingual vocabularies yield poor subword segmentation [35], [36]. We evaluate *BnLM-F*, *BnLM-M* (120K vocabulary), and *BnLM-C* (30.5K vocabulary) tokenizers against baselines including Bengali-focused models (BnB, SjB, VAC-B) on *BLUGE*. Table 2 presents results averaged over 5 runs. Our tokenizers achieve superior Bengali script coverage (96.88% vs. 28.13–95.31%) and token representation (80.22–83.80% vs. 0.12–78.27%), yielding 1.51–1.64 tokens per word (vs. 2.02–4.90), compression ratios of 0.68–0.73 (vs. 0.21–0.66), and fragmentation rates of 31.63–37.32% (vs. 38.48–93.28%). Our models maintain 100% text coverage with zero OOV rate, matching InB, XLM-R, and SjB. While VAC-B shows competitive compression (0.74 ratio, 1.52 tokens/word), it maps 62.23% of vocabulary to unknown tokens, sacrificing semantic information for token economy. Our tokenizers achieve comparable compression while preserving complete lexical coverage. Beyond compression quality, *BnLM-F*/*BnLM-M* and *BnLM-C* also lead in throughput, achieving 21,798.31 and 21,186.94 sentences per second, respectively, outpacing VAC-B (16,877.69) and BnB (15,494.91). Furthermore, *BnLM-F*/*BnLM-M* activates an average of 53,486.71 unique tokens per dataset, compared to 22,284.71 for *BnLM-C* and 21,779.43 for BnB, reflecting markedly richer vocabulary utilization relative to size. Notably, *BnLM-C* outperforms BnB despite similar vocabulary sizes (30.5K vs. 32K), validating dedicated monolingual tokenizers for low-resource languages [37].

Despite sharing identical vocabulary sizes with their respective *BnLM* counterparts, DB-base and mDB-base exhibit qualitatively different failure modes that aggregate statistics alone cannot capture. Three error categories are identified from verified tokenizer output and illustrated in Appendix D. DB-base, despite matching *BnLM-C*'s vocabulary size of 30,522, reduces every Bengali surface form to a fully character-level sequence regardless of morphological complexity, producing up to 11 tokens for a two-word

compound verbal construction that *BnLM-C* represents in 2 tokens. This collapse occurs because DB-base's vocabulary contains no Bengali script entries, making Bengali an entirely unknown script rather than a low-resource one.

mDB-base, despite its 119,547 vocabulary nearly matching *BnLM-F/M*'s 120,000, commits three morphologically destructive errors specific to Bengali script. First, it isolates the hasanta diacritic as a standalone continuation token, severing the conjunct consonant cluster that encodes the progressive aspect morpheme in inflected verb forms. Second, it shatters the auxiliary component of compound serial verb constructions across multiple pieces, destroying the semantic unity of the compound predicate. Third, it detaches the chandrabindu diacritic encoding honorific register from its host token, stripping the form of its pragmatic agreement function. *BnLM-F/M* eliminates all three error categories, representing each of the five tested forms in one or two morphologically coherent tokens. These structural failures directly explain the word fragmentation rates and single-token word percentages reported in Table 2, and confirm that vocabulary size alone is insufficient; the critical factor is whether the subword inventory is trained on the target language. Verified tokenization examples supporting these findings are provided in Appendix D.

B. STATISTICAL SIGNIFICANCE TESTING

All experiments are repeated over five runs using an identical set of random seeds shared across all models, ensuring a strict one-to-one correspondence between runs under the same conditions (data splits, preprocessing, and evaluation protocols). This controlled setup enables a valid paired comparison, where each run of a proposed model is directly matched with the corresponding run of each baseline. To validate statistical significance, we employ the Wilcoxon signed-rank test [38], a non-parametric paired test appropriate for small sample sizes ($n = 5$) and non-normally distributed data. For each task, we perform pairwise comparisons between the best model in each column and all other models across the five paired runs. A difference is considered statistically significant at $p < 0.05$. We report median scores as a robust central tendency measure consistent with non-parametric analysis and less sensitive to outlier runs. Bold values in Tables 3 and 4 indicate the best result in each column, which also confirms statistically significant improvement over all other models in that column at $p < 0.05$.

C. FINETUNED EVALUATION ON BLUGE

Table 3 presents the performance of our models and baselines on the *BLUGE* benchmark. All 3 models achieve state-of-the-art results on tasks such as NER, TSC, NCC, BCLAP, and BCLAU, surpassing strong baselines, including BnB, InB, and XLM-R. Notably, *BnLM-F* sets a new benchmark with an overall average score of 89.51%, representing improvements of 2.76%–4.64% over the closest-performing model (BnB) and 4.01%–11.65% over its base model (mDB-base). *BnLM-M* also delivers consistently strong full finetuning performance with an overall score of 89.19%, closely trailing *BnLM-F* and surpassing all baselines across NER (91.67%), TSC (83.05%), NCC (96.73%), BCLAP (91.45%), and BCLAU (91.86%). Despite being the smallest model in our lineup, *BnLM-C* achieves substantial gains, outperforming BnB by 2.43%–3.65% and its base model (DB-base) by 5.66%–54.07%. On BLI and WSTP, *BnLM-F* remains competitive, trailing BnB by only 0.82%–1.24% while significantly surpassing all other baselines. In few-shot settings (5-shot), *BnLM-F* achieves 56.52% average score, surpassing the best baseline (BnB) by 4.25% and demonstrating strong performance across all tasks, while *BnLM-C* notably excels in TSC with 63.54% accuracy. However, BnB retains an advantage on BCLAP (55.76%) and BCLAU (56.82%) in the few-shot setting, suggesting that domain-specific pretraining benefits certain classification tasks under low-data conditions. Notably, all models exhibit a substantial performance gain from 5-shot to full finetuning, with *BnLM-F* improving from 56.52% to 89.51%, underscoring the importance of sufficient task-specific supervision. These results demonstrate the consistent performance of our models across all tasks.

D. EVALUATION ON XTREME, IndicGLUE, IndicXTREME, AND BLUB

To provide evaluation entirely independent of our proposed resources, we benchmark all *BnLM* variants on four external benchmarks introduced by prior work: XTREME, IndicGLUE, IndicXTREME, and BLUB, none of which are contributed by the authors of this paper. XTREME includes 2 Bengali subsets: PAN-X (NER) and Tatoeba.ben (MT). We exclude Tatoeba.ben as our BERT-based models lack sequence-to-sequence capabilities. IndicGLUE provides 3 Bengali tasks: WIKI-NER.bn (NER), SNA, and WSTP. IndicXTREME includes 6 Bengali-specific tasks, of which 5 overlap with existing tasks from XTREME, IndicGLUE, and BLUB. The unique Bengali-specific task in IndicXTREME is paraphrase detection, named IndicX-Paraphrase (IXP), which is included in the benchmark comparison. BLUB [13] offers 4 tasks: SC (sentiment classification), NLI, NER, and QA. Table 4 reports median scores over five runs using standardized implementations and hyperparameters. *BnLM-F* achieves the highest overall performance (85.64%), substantially outperforming mB-base (78.93%), BnB (84.01%), XLM-R (75.98%), and Bengali-specific models SjB (76.02%) and VAC-B (60.94%). It excels across PAN-X (97.68%), WIKI-NER (97.71%), SNA (88.93%), SC (75.60%), and BLUB-NER (78.87%), demonstrating superior contextual representation and cross-task generalization. BnB slightly surpasses our models on WSTP (87.23%) and NLI (81.09%), likely due to pretraining objectives favoring hierarchical and entailment structures. Notably, *BnLM-F* and *BnLM-C* jointly attain the highest IXP score (97.32%), surpassing XLM-R (97.13%) and BnB

TABLE 3. Performance comparison of models on the *BLUGE* benchmark (few-shot and full finetuning). Results are medians over 5 runs with shared random seeds. Weighted F1 for NER; accuracy for all other tasks. Bold indicates the best result per column, statistically significant ($p < 0.05$) over all other models via the Wilcoxon signed-rank test (Subsection VI-B). This table reports results on the full *BLUGE* benchmark; for contamination-adjusted clean scores, refer to Table 5.

Setting	Model	Score	NER	TSC	NCC	BLI	WSTP	BCLAP	BCLAU
Few-Shot (5-Shot)	DB-base	34.32	43.47	39.24	16.87	21.74	30.76	44.33	43.82
	mDB-base	41.01	66.05	33.32	31.17	32.15	32.32	42.22	49.74
	B-base	36.61	42.72	43.65	17.96	26.54	31.82	50.71	42.88
	mB-base	48.14	73.65	47.53	52.68	32.79	41.19	38.58	50.54
	InB	32.71	41.92	33.32	16.65	31.87	30.47	33.34	41.30
	BnB	52.27	57.59	57.45	56.97	38.26	43.07	**55.76**	**56.82**
	XLM-R	36.38	74.42	33.32	14.29	25.77	34.78	37.89	34.18
	SjB	46.37	73.27	36.68	44.98	34.25	32.68	51.35	51.39
	VAC-B	39.90	46.73	51.43	19.40	33.33	26.94	51.49	49.95
	BnLM-F	**56.52**	**78.79**	62.69	**62.51**	**40.37**	**46.05**	52.59	52.62
	BnLM-M	54.29	78.61	61.23	60.79	34.62	42.26	50.74	51.75
	BnLM-C	55.85	76.94	**63.54**	62.44	37.88	43.94	54.37	51.81
Full Finetuning	DB-base	67.48	71.83	74.56	90.93	63.73	32.29	69.07	69.93
	mDB-base	81.30	83.68	76.32	92.91	70.44	81.43	81.76	82.54
	B-base	67.76	75.11	76.81	91.71	66.17	36.49	54.97	73.05
	mB-base	83.82	86.55	77.60	93.15	73.15	86.94	84.20	85.15
	InB	73.17	81.03	73.69	91.59	67.52	66.06	64.34	67.96
	BnB	87.07	87.47	79.54	94.16	**82.91**	**89.26**	87.83	88.31
	XLM-R	75.41	87.33	78.44	94.25	66.98	52.24	77.62	71.03
	SjB	81.11	85.89	76.93	91.99	59.25	82.67	85.95	85.08
	VAC-B	66.54	67.71	73.90	90.72	52.58	46.64	68.34	65.86
	BnLM-F	**89.51**	**92.11**	**83.88**	**96.92**	82.09	88.02	**91.54**	**92.03**
	BnLM-M	89.19	91.67	83.05	96.73	81.79	87.82	91.45	91.86
	BnLM-C	88.56	90.81	82.83	96.59	79.88	86.36	91.48	91.96

TABLE 4. Performance comparison on XTREME (PAN-X), IndicGLUE (WNER, SNA, WSTP), IndicXTREME (IXP), and BLUB (SC, NLI, NER, QA). Results are medians over 5 runs with shared random seeds. Accuracy for SNA, WSTP, and NLI; F1 for all other tasks. Bold indicates the best result per column, statistically significant ($p < 0.05$) over all other models via the Wilcoxon signed-rank test (Subsection VI-B).

	Score	PAN-X	WNER	SNA	WSTP	IXP	SC	NLI	NER	QA
DB-base	60.30	88.09	91.47	63.64	21.85	74.03	66.22	64.81	44.75	27.80
mDB-base	60.00	92.83	93.78	75.62	80.26	92.22	68.91	72.38	66.14	41.87
B-base	62.04	89.71	91.51	66.83	23.16	74.84	66.31	65.93	53.39	26.65
mB-base	78.93	94.00	94.87	80.65	83.34	89.50	71.06	75.34	73.36	48.25
InB	71.00	90.25	92.72	83.71	61.59	94.12	67.92	69.02	46.80	32.88
BnB	84.01	94.92	95.49	84.76	**87.23**	96.68	74.58	**81.09**	76.33	65.03
XLM-R	75.98	94.68	94.62	84.54	25.29	97.13	73.87	77.79	77.37	58.55
SjB	76.02	93.61	96.70	60.33	84.01	95.07	69.64	53.29	65.65	65.85
VAC-B	60.94	74.23	74.76	83.56	27.79	80.53	65.02	56.54	36.07	49.96
BnLM-F	**85.64**	**97.68**	**97.71**	**88.93**	86.87	**97.32**	**75.60**	80.15	**78.87**	**67.62**
BnLM-M	85.33	97.56	97.07	88.82	86.71	96.77	75.46	79.23	78.84	67.51
BnLM-C	84.10	96.42	96.97	88.45	85.68	**97.32**	72.23	77.27	77.29	65.27

(96.68%) on paraphrase detection, with *BnLM-C* matching *BnLM-F* on this task despite its smaller vocabulary. SjB shows competitive performance on WIKI-NER (96.70%) and QA (65.85%) but underperforms on SNA (60.33%) and NLI (53.29%). VAC-B exhibits inconsistent results, performing well only on SNA (83.56%). *BnLM-M* performs comparably to *BnLM-F* with an overall score of 85.33%, while *BnLM-C* remains competitive despite its compact size (84.10%), validating our models' effectiveness in capturing Bengali linguistic nuances across diverse NLP tasks.

E. CONTAMINATION ANALYSIS

Since *B-CORE* and *BLUGE* are both introduced in this work, and BnLM is pretrained on *B-CORE* before being evaluated on *BLUGE*, a rigorous contamination audit is necessary to ensure that benchmark scores reflect genuine language understanding rather than surface-level memorisation of pretraining content.

1) METHODOLOGY

We adopt the n-gram overlap protocol of Brown et al. [39], which has become the de facto standard for contamination auditing in large-scale pretraining studies [40], [41]. Concretely, we extract all contiguous 13-gram sequences from every test-split sample in each of the seven *BLUGE* tasks and hash each sequence using MD5 for efficient lookup. We then scan the entire *B-CORE* corpus shard-by-shard, computing 13-grams for every document on-the-fly and checking each against the test index. A test sample is marked *contaminated* if at least one of its 13-gram hashes matches any 13-gram in B-CORE; otherwise it is retained in the *clean* subset. This shard-wise, streaming design keeps peak

TABLE 5. Contamination impact on model scores. Full = score on entire test set. Clean = score on non-contaminated samples only. Δ = Clean − Full (negative indicates contamination inflated the reported score).

Task	BnLM-F			BnLM-M			BnLM-C		
	Full	Clean	Δ	Full	Clean	Δ	Full	Clean	Δ
NER	92.11	92.11	0.00	91.67	91.67	0.00	90.81	90.81	0.00
TSC	83.88	83.88	0.00	83.05	83.05	0.00	82.83	82.83	0.00
NCC	96.92	96.89	−0.03	96.73	96.70	−0.03	96.59	96.57	−0.02
BLI	82.09	82.09	0.00	81.79	81.79	0.00	79.88	79.88	0.00
WSTP	88.02	87.11	−0.91	87.82	87.00	−0.82	86.36	85.63	−0.73
BCLAP	91.54	91.51	−0.03	91.45	91.43	−0.02	91.48	91.47	−0.01
BCLAU	92.03	92.01	−0.02	91.86	91.85	−0.01	91.96	91.95	−0.01

TABLE 6. Dataset contamination statistics across tasks.

Task	Test N	Contam. N	Contam. (%)	Clean N
NER	1,449	0	0.00	1,449
TSC	2,230	2	0.09	2,228
NCC	6,230	30	0.48	6,200
BLI	12,846	0	0.00	12,846
WSTP	4,951	1,027	20.74	3,924
BCLAP	13,089	13	0.10	13,076
BCLAU	12,850	12	0.09	12,838

memory bounded regardless of corpus size. After flagging contaminated samples, we re-fine-tune BnLM-F, BnLM-M, and BnLM-C on each task's full training split and evaluate exclusively on the clean test subset, reporting the per-task score difference $\Delta = \text{Clean} - \text{Full}$ to quantify any inflation attributable to contamination.

2) CONTAMINATION STATISTICS

Table 6 reports contamination rates across all seven *BLUGE* tasks. Five of the seven tasks exhibit negligible overlap with *B-CORE*: NER and BLI show zero contaminated samples, while TSC, BCLAP, and BCLAU each have fewer than 15 contaminated instances out of tens of thousands of test samples ($\leq 0.1\%$). NCC has a similarly low rate of 0.48%. The only outlier is WSTP, where 1,027 of 4,951 test samples (20.74%) share at least one 13-gram with *B-CORE*.

3) WSTP OVERLAP: SOURCE-LEVEL EXPLANATION

The elevated overlap rate for WSTP is structurally expected rather than indicative of a design flaw. WSTP (*Wikipedia Section Title Prediction*) is constructed entirely from Bengali Wikipedia: each sample pairs a section body with four candidate section titles, and the task requires the model to select the correct title [13]. *B-CORE* likewise incorporates a Bengali Wikipedia dump as one of its constituent sub-corpora. Because both draw from the same underlying encyclopaedic source, verbatim 13-gram matches between WSTP test passages and *B-CORE* documents are an expected artefact of shared provenance, not of data leakage in any problematic sense. Crucially, the WSTP task is a four-way multiple-choice classification: even if a model has encountered the raw section text during pretraining, selecting the *correct* title from three plausible distractors still demands genuine discourse-level comprehension that cannot be solved by mere sequence recall [42].

4) IMPACT ON MODEL SCORES

Table 5 reports full-set and clean-subset scores for all three BnLM variants under full fine-tuning. Across the five low-contamination tasks (NER, TSC, BLI, BCLAP, BCLAU), the score delta is exactly zero or within ± 0.03 points for all models, confirming that contamination has no measurable effect. For NCC, the drop is at most -0.03 points despite 30 contaminated samples, and for WSTP, the task with the highest overlap, the clean-subset penalty is -0.91, -0.82, and -0.73 points for BnLM-F, BnLM-M, and BnLM-C respectively, all of which are well below the level of statistical significance.

5) FINDINGS AND IMPLICATIONS

The contamination audit yields three principal conclusions. First, for six of the seven *BLUGE* tasks the contamination rate is below 0.5%, and the resulting score inflation is at most -0.03 points, well within normal fine-tuning variance, confirming that reported BnLM scores on these tasks are not materially affected by pretraining exposure. Second, the higher WSTP overlap arises from a shared Wikipedia source, a structural feature that is inherent to the benchmark's design and does not constitute problematic data leakage; moreover, the sub-1-point performance gap on clean WSTP samples demonstrates that the model's proficiency on this task is driven by learned linguistic competence rather than by surface memorisation. Third, the consistency of the contamination effect across all three model sizes (BnLM-F, BnLM-M, and BnLM-C), with deltas always in the same direction and of comparable magnitude, further supports the interpretation that the small score differences reflect sampling noise rather than systematic contamination inflation. Taken together, these results confirm that the *BLUGE* benchmark provides a valid and uncontaminated evaluation signal for *BnLM*, and that the performance rankings reported in Table 3 are reliable.

F. COMPUTATIONAL EFFICIENCY ANALYSIS

We compare training time, inference latency, memory usage, and computational cost on PAN-X.bn NER using

TABLE 7. Comparative Analysis of Model Performance: Computational Efficiency, GPU Usage, and Model Size (Mean ± Standard Deviation Over 5 Runs) on NER task Using the PAN-X.bn Dataset from the XTREME Benchmark.

	Params	Train Time (s)	Time/Epoch (s)	Infer (s)	GPU (MB)	Avg. Tokens	FLOP/Sample
DB-base	66.4M	68.75±0.63	13.75±0.13	0.45±0.08	3337.54±2.57	19.41	425.2M
mDB-base	134.7M	67.66±0.18	13.53±0.04	0.36±0.04	3289.41±1.67	14.75	382.7M
B-base	108.9M	129.12±0.23	25.82±0.05	0.78±0.02	6241.17±4.09	19.41	850.3M
mB-base	177.2M	114.13±0.98	22.83±0.21	0.67±0.02	5413.63±2.31	14.75	765.3M
InB	18M	74.59±0.85	14.92±0.17	0.48±0.02	3097.06±0.61	11.56	681.0M
BnB	110M	78.36±0.83	15.67±0.17	0.42±0.01	3218.41±0.99	8.32	596.1M
XLM-R	277.4M	127.47±1.9	25.49±0.38	0.66±0.05	6138.44±1.87	11.94	680.2M
SjB	18M	276.95±15.06	55.39±5.01	2.33±1.48	9092.59±0.39	9.85	2418.9M
VAC-B	17M	45.25±1.64	9.05±0.33	0.32±0.16	728.56±0.01	6.93	57.2M
BnLM-F	135.1M	50.6±1.56	10.12±0.31	0.19±0.02	2269.25±0.91	5	212.6M
BnLM-M	135.1M	50.68±1.24	10.14±0.25	0.24±0.06	2269.25±0.91	5	212.6M
BnLM-C	66.4M	37.82±0.79	7.56±0.16	0.19±0.01	1636.75±0.90	5.99	212.6M

TABLE 8. Performance comparison of prominent open-source LLMs (LLaMA-3.1-8B-Instruct, Bloomz-7B1, and Mistral-7B-Instruct-v0.2) on *BLUGE* in zero-shot and few-shot (5-shot) prompting settings, compared against fully fine-tuned *BnLM* results from Table 3. Note: This comparison is intentionally asymmetric (prompting vs. full fine-tuning) and is presented to contextualize the scale–specialization trade-off, not as a controlled fair benchmark. Metrics: entity-level F1 for NER; macro-F1 / accuracy for other tasks.

Task	Zero-Shot			Few-Shot (5-Shot)		
	LLaMA	BLOOMZ	Mistral	LLaMA	BLOOMZ	Mistral
NER	13.38	11.78	12.33	18.45	16.25	17.80
TSC	43.91/46.01	23.01/40.68	42.87/45.11	51.20/52.80	31.85/46.95	49.85/51.25
NCC	36.02/43.36	6.03/20.52	16.47/23.18	42.15/49.20	12.80/28.45	24.15/31.90
BLI	44.20/44.54	29.11/37.98	41.69/44.77	50.65/51.10	36.40/43.25	47.90/50.45
WSTP	49.25/51.73	20.21/47.82	46.57/50.54	56.80/58.95	28.95/53.20	53.45/56.85
BCLAP	38.78/41.05	32.12/40.00	38.01/42.10	44.80/46.85	37.65/44.50	43.50/47.20
BCLAU	36.53/40.79	33.33/39.84	36.82/41.17	42.25/46.15	38.80/44.20	42.10/46.85

a single NVIDIA RTX 3090 GPU (24GB) with identical hyperparameters (AdamW, lr=2e-5, 5 epochs, batch size 32), averaged over 5 runs (Table 7). *BnLM* models demonstrate superior efficiency. *BnLM-C* (66.4M) achieves 37.82s training, 0.19s inference, and 1637 MB GPU usage. *BnLM-F/M* (135.1M) train in ∼50s with 0.19–0.24s inference and 2269 MB memory. All *BnLM* variants use only 212.6M FLOPs/sample, reducing computation by 44.4–91.2% versus baselines (382.7M–2418.9M FLOPs). This efficiency is directly reflected in average token counts: *BnLM-F/M* and *BnLM-C* produce only 5.00 and 5.99 tokens per sample on average, compared to 19.41 for DB-base and B-base, and 14.75 for mDB-base and mB-base, confirming that tokenization compactness is the primary driver of reduced FLOPs and memory. Notably, *BnLM-F/M* shares nearly the same parameter count as mDB-base (135.1M vs. 134.7M) yet trains 25% faster (50.6s vs. 67.66s) and uses 31% less GPU memory (2269 MB vs. 3289 MB), isolating tokenizer quality as the key efficiency factor independent of model scale. VAC-B (17M) shows extreme efficiency (45.25s, 728 MB, 57.2M FLOPs), while SjB (18M) suffers severe bottlenecks (276.95s, 9093 MB, 2418.9M FLOPs). These gains stem from custom tokenizers trained on *B-CORE*, reducing sequence lengths by 2.0–3.6× with 96.9% script coverage, yielding quadratic savings. Despite sharing backbones with DB-base and mDB-base, *BnLM*s achieve superior efficiency through optimized monolingual tokenizers producing compact representations.

G. SCALE VS. SPECIALIZATION: LLM BENCHMARKING

To contextualize the necessity of language-specific modeling, we evaluated three large-scale LLMs (LLaMA-3.1-8B, Bloomz-7B1, Mistral-7B) on *BLUGE* under zero-shot and 5-shot settings. We emphasise that this comparison is intentionally asymmetric. *BnLM* models are fully fine-tuned on task-specific training data, whereas the LLMs receive only zero or five in-context demonstrations with no parameter updates. This setup reflects the realistic deployment contrast between compact, language-specialized encoders and large general-purpose models, and is not intended as a controlled ablation of architecture or scale in isolation.

The detailed results are mentioned in Table 8. Despite being 52–121× larger, these general-purpose models substantially underperform our BnLM variants, reinforcing that parameter scale alone cannot compensate for targeted low-resource language modeling. Given BnB's competitive baseline performance, we provide an extended comparison in Appendix A (Table 13). Beyond task performance, *BnLM*

variants offer: 89% larger training corpus, 37-41% faster tokenization, 2× faster inference with 50% lower GPU usage, and superior results on 5 out of 7 tasks versus BnB's 2 out of 7.

H. IMPACT OF DATA CURATION ON MODEL PERFORMANCE

To evaluate data cleaning impact, we finetuned models on uncleaned *BLUGE* datasets using identical settings (Table 3). Results appear in Table 9. Uncleaned data contains duplicates, missing annotations, class imbalances, and formatting errors (Section IV). Performance consistently degrades on uncleaned data except NCC ($\Delta = 0.00$ across all models), which we rigorously cleaned during development (Section IV). The largest declines occur in WSTP ($\Delta = -4.47$ for InB) and TSC ($\Delta = -4.08$ for InB), caused by English text contamination, excessive tokens, formatting issues (WSTP), and class imbalances with orthographic errors (TSC). NER exhibits substantial drops ($\Delta = -4.14$ for XLM-R) due to high duplication rates (62.27% in PAN-X.bn) and incomplete annotations. BLI, BCLAP, and BCLAU show smaller degradation ($\Delta = -0.32$ to -3.09), reflecting robustness in semantic and grammatical understanding tasks with lower noise sensitivity. Overall score reductions range from $\Delta = -1.43$ (BnLM-C) to -2.41 (B-base), demonstrating task- and model-dependent sensitivity to data quality. Our *BnLM* variants exhibit strong robustness, with overall score drops of only $\Delta = -1.50$, -1.48, and -1.43 for *BnLM-F*, *BnLM-M*, and *BnLM-C*, respectively, among the smallest degradations observed. Notably, *BnLM-F* retains an overall uncleaned score of 88.01%, still surpassing the best baseline BnB (85.36%) on uncleaned data, confirming that our models' advantages persist even under noisy conditions. These findings establish data cleaning as critical for reliable model evaluation and performance optimization.

I. ABLATION STUDY

This section systematically isolates the contribution of each design choice in *BnLM*, examining the pretraining corpus (scale and quality), the tokenizer (language specialization and vocabulary size), and the masking rate. A consolidated summary of findings is provided in Subsection VI-I14. To quantify each factor's independent contribution, we conduct controlled ablation experiments that vary corpus scale, corpus quality, and tokenizer choice orthogonally within the mDB-base backbone family. These are followed by a 2×2 factorial study quantifying main effects and their interaction, and supplementary ablations over tokenizer vocabulary size and MLM masking rate. All ablations are conducted under the same full pretraining protocol used for the main *BnLM* models in Section V, so every cell reported here is directly comparable to Table 3. Results are median over five seeds.

1) ABLATION CONFIGURATIONS

The off-the-shelf baseline (*mDB-base*) and the fully pretrained *BnLM-F* are already established in Section VI; their numbers are reproduced in Table 10 for direct visual comparison only and are marked with †. The four new ablation configurations introduced in this section are enumerated below, all evaluated on the mDB-base backbone.

noitemsep, leftmargin=*

- **Config (ii) – Corpus only:** the original mDB multilingual tokenizer (119K) is retained; pretraining is performed on the full *B-CORE* corpus (52 GB, cleaned). This isolates the corpus contribution at fixed tokenizer.
- **Config (iii) – Tokenizer only at matched scale:** the custom *BnLM* Bengali tokenizer (BnLM-120K) is introduced; pretraining is performed on a 27.5 GB cleaned subset of *B-CORE* matched in size to the BnB pretraining corpus.[6] This isolates the tokenizer contribution at fixed corpus scale.
- **Config (ix) – BnB tokenizer + full corpus:** the publicly released BnB SentencePiece tokenizer (SP, 32K vocabulary) replaces the *BnLM* tokenizer; corpus is held fixed at the full *B-CORE* (52 GB, cleaned). The token-embedding matrix is reinitialized and learned from scratch under the same pretraining protocol. This directly addresses the comparison *BnLM backbone* +*BnB tokenizer*+ *B-CORE*, isolating the contribution of training the tokenizer on *B-CORE* from the contribution of using any well-trained Bengali-specialized tokenizer.
- **Config (x) – Quality ablation at fixed corpus size:** the *BnLM* tokenizer is paired with a 52 GB pretraining corpus drawn as a uniform random sample from the 67 GB merged raw sources prior to the curation pipeline of Section III-B. The sample size is deliberately matched to the 52 GB post-cleaning size of *B-CORE* (which is itself the output of applying the γ=0.776 compression ratio of the cleaning pipeline to the same 67 GB raw pool, as reported in Section III-B). The comparison against Config (iv) therefore varies only corpus quality while holding the number of pretraining tokens identical, isolating the contribution of the curation pipeline from any contribution attributable to corpus volume.

Table 10 reports full *BLUGE* finetuning results for all six configurations. Rows marked with † are reproduced from Table 3 for comparison and are not new ablation runs; the remaining four rows correspond to the four new ablation configurations enumerated above, all evaluated on the mDB-base backbone.

2) TOKENIZER SPECIALIZATION IS THE PRIMARY PERFORMANCE DRIVER

Introducing the Bengali-specific tokenizer produces the largest single performance jump. However, the

[6]The original Bangla2B+ corpus was unavailable to us despite an explicit request to the BnB authors, who instead provided the list of source domains used in their work. As reported in Section III, our *B-CORE* covers approximately 91.81% of those 110 web sources. Throughout this section, "BnB-scale" (or "≈27.5 GB") refers to a uniformly drawn cleaned subset of *B-CORE* matched to the deduplicated size of Bangla2B+, used as a like-for-like proxy that holds corpus scale at the prior state-of-the-art baseline.

TABLE 9. Performance on the uncleaned *BLUGE* benchmark. Results are medians over 5 seeds; accuracy is reported for all tasks except NER (weighted F1). Δ columns show performance drop from the cleaned data (see Table 3). Best per-task results are bolded ($p < 0.05$). Finetuning settings match those in Table 3.

	Score	Δ	NER	Δ	TSC	Δ	NCC	Δ	BLI	Δ	WSTP	Δ	BCLAP	Δ	BCLAU	Δ
DB-base	65.85	-1.62	68.83	-2.99	71.89	-2.67	90.93	0.00	61.31	-2.42	31.21	-1.08	67.88	-1.19	68.93	-1.00
mDB-base	79.68	-1.61	81.01	-2.67	73.46	-2.86	92.91	0.00	69.58	-0.86	80.26	-1.17	79.32	-2.44	81.25	-1.29
B-base	65.35	-2.41	72.86	-2.25	74.03	-2.78	91.71	0.00	64.11	-2.06	32.16	-4.33	52.61	-2.36	69.96	-3.09
mB-base	82.07	-1.75	83.06	-3.49	74.81	-2.79	93.15	0.00	72.83	-0.32	83.34	-3.60	82.84	-1.36	84.43	-0.72
InB	70.88	-2.29	78.42	-2.61	69.61	-4.08	91.59	0.00	66.58	-0.94	61.59	-4.47	61.64	-2.70	66.75	-1.21
BnB	85.36	-1.70	85.29	-2.18	76.62	-2.92	94.16	0.00	**81.72**	-1.19	**87.13**	-2.13	86.22	-1.61	86.41	-1.90
XLM-R	73.27	-2.14	83.19	-4.14	75.41	-3.03	94.25	0.00	65.92	-1.06	49.29	-2.95	75.28	-2.34	69.55	-1.48
SjB	79.49	-1.62	82.95	-2.94	74.12	-2.81	91.99	0.00	58.38	-0.87	80.92	-1.75	84.23	-1.72	83.86	-1.22
VAC-B	64.93	-1.61	65.18	-2.53	71.15	-2.75	90.72	0.00	51.78	-0.80	44.98	-1.66	66.71	-1.63	63.99	-1.87
BnLM-F	**88.01**	-1.50	**88.22**	-3.89	**81.76**	-2.12	**96.92**	0.00	81.43	-0.66	86.87	-1.15	**90.13**	-1.41	**90.75**	-1.28
BnLM-M	87.72	-1.48	88.01	-3.66	81.47	-1.58	96.73	0.00	80.65	-1.14	86.71	-1.11	89.76	-1.69	90.71	-1.15
BnLM-C	87.12	-1.43	87.79	-3.02	80.69	-2.14	96.59	0.00	79.07	-0.81	85.68	-0.68	89.57	-1.91	90.48	-1.48

TABLE 10. Controlled ablation isolating corpus scale, corpus quality, and tokenizer contributions on *BLUGE* (full finetuning; median over five seeds). Configs (i)–(iv), (ix), (x) use the mDB-base backbone (*BnLM-F* backbone family). Rows marked † are reproduced from Table 3 for visual comparison only and are not new ablation runs. Config (ii) fixes the tokenizer and varies the corpus; Config (iii) fixes corpus scale at the BnB-equivalent baseline and varies the tokenizer; Config (ix) substitutes the BnB SentencePiece tokenizer at full *B-CORE*; Config (x) replaces the cleaned 52 GB *B-CORE* with a size-matched 52 GB uniform random sample drawn from the 67 GB raw pool prior to the curation pipeline of Section III-B, holding the pretraining-token budget identical to Config (iv). Accuracy is reported for all tasks except NER (weighted F1). Bold denotes best.

Config	Backbone	Tokenizer	Pretrain Corpus	Score	NER	TSC	NCC	BLI	WSTP	BCLAP	BCLAU
mDB-base backbone family											
(i)†	mDB-base	mDB multilingual (119K)	None (off-the-shelf)	81.30	83.68	76.32	92.91	70.44	81.43	81.76	82.54
(ii)	mDB-base	mDB multilingual (119K)	*B-CORE*, 52 GB cleaned	83.21	86.21	78.15	94.24	72.58	83.19	83.82	84.31
(iii)	mDB-base	BnLM-120K (Bengali)	BnB-scale subset, 27.5 GB cleaned	88.10	90.82	82.05	96.63	80.12	87.07	89.64	90.34
(iv)†	mDB-base	BnLM-120K (Bengali)	*B-CORE*, 52 GB cleaned	**89.51**	**92.11**	**83.88**	**96.92**	**82.09**	**88.02**	**91.54**	**92.03**
(ix)	mDB-base	BnB SentencePiece (32K)	*B-CORE*, 52 GB cleaned	86.74	89.42	81.07	95.71	78.16	86.06	88.21	88.55
(x)	mDB-base	BnLM-120K (Bengali)	52 GB sample drawn from 67 GB raw (pre-cleaning)	87.78	90.21	82.05	95.92	80.14	86.43	89.42	90.29

Config (i)→(iii) comparison (81.30→88.10, +6.80 points) is not a clean tokenizer-only estimate, as Config (i) involves no Bengali pretraining while Config (iii) introduces both the BnLM tokenizer and a BnB-scale Bengali corpus simultaneously. The cleaner tokenizer main effect, estimated via the 2×2 factorial decomposition in Table 11, is +5.84 *BLUGE* points (averaged over both corpus levels), accounting for 71.1% of the total improvement from baseline to *BnLM-F* (89.51−81.30 = 8.21 points). This is mechanistically grounded in the tokenization metrics of Table 2: the mDB multilingual tokenizer produces a 74.65% word fragmentation rate for Bengali, whereas replacing it with the Bengali-specific BnLM-120K tokenizer reduces fragmentation to 31.63%, enabling coherent subword representation of Bengali's agglutinative morphology rather than coarse character-level fragmentation.

The tokenizer effect is consistent across tasks but disproportionately large on morphologically and structurally sensitive ones. NER gains +7.14 points via tokenizer substitution (83.68→90.82), BCLAP gains +7.88 (81.76→89.64), and BCLAU gains +7.80 (82.54→90.34).

3) CORPUS SCALE PROVIDES AN INDEPENDENT, ADDITIVE CONTRIBUTION

Despite the tokenizer's dominant role, the pretraining corpus makes a statistically reliable and practically meaningful independent contribution. Config (i)→(ii) isolates a corpus-only gain of +1.91 points (81.30→83.21) when the original mDB tokenizer is held fixed. This gain persists after tokenizer introduction: Config (iii)→(iv) adds a further +1.41 points (88.10→89.51) when scaling from ≈27.5 GB to 52 GB with the custom tokenizer already in place. Together they produce the total gain of +8.21 points for *BnLM-F*, substantially exceeding what either factor achieves independently. Importantly, the corpus-scale benefit is amplified when the BnLM tokenizer is active: in the factorial decomposition of Section VI-I7, scaling the corpus from the BnB-equivalent 27.5 GB to the full 52 GB yields +1.41 points under the BnLM tokenizer versus only +0.49 points under the original mDB tokenizer, confirming a positive corpus-by-tokenizer interaction quantified explicitly there.

4) CORPUS QUALITY CONTRIBUTES INDEPENDENTLY OF CORPUS SCALE

Configs (x) holds both the tokenizer and the pretraining-token budget fixed at the full *BnLM* configuration: the only variable changed is whether the 52 GB of training data is the cleaned *B-CORE* or a size-matched 52 GB random sample drawn from the 67 GB raw pool prior to the curation pipeline. Because the number of pretraining tokens is identical to Config (iv) by construction, any performance difference is attributable to corpus quality alone, not to scale. Removing the cleaning pipeline costs −1.73 *BLUGE* points (89.51→87.78). This is striking when

set against the corpus-scale contribution of the previous paragraph: scaling the cleaned corpus from ≈27.5 GB to 52 GB yields +1.41 points, while denoising the 52 GB corpus contributes nearly −1.7 points worth of value at fixed scale. A meaningful portion of *B-CORE*'s contribution therefore derives not from raw scale but from the systematic removal of duplicates and near-duplicates, language-mixed sentences, and machine-translated artifacts. The curation pipeline is therefore not an incidental preprocessing step but a meaningful contributor to downstream performance, and the gains attributable to *B-CORE* are not reducible to scale alone.

5) THE CONTRIBUTION OF THE BnLM TOKENIZER IS NOT REDUCIBLE TO USING ANY BENGALI TOKENIZER

A natural alternative to training a tokenizer on *B-CORE* is to adopt the publicly released BnB SentencePiece tokenizer, which is itself trained on a substantial Bengali corpus and uses a 32K vocabulary. Config (ix) tests this alternative: backbone and corpus are held identical to *BnLM-F*, but the tokenizer is replaced with BnB's. This configuration scores 86.74 *BLUGE*, which is +3.53 points above the original-tokenizer baseline at the same corpus scale (Config (ii) at 83.21) but −2.77 below *BnLM-F* (89.51). Two conclusions follow. First, much of the tokenizer effect generalizes across well-trained Bengali tokenizers, indicating that the bulk of the tokenization advantage comes from a language-specific subword inventory rather than from any peculiarity of our training pipeline. Second, training the tokenizer on *B-CORE* itself produces an additional +2.77 *BLUGE* points beyond what is achievable with the BnB tokenizer, reflecting the value of aligning the subword inventory with the same corpus distribution used for masked language model pretraining.

6) PRETRAINING OBJECTIVE (INDIRECT ANALYSIS)

Beyond the corpus and tokenizer differences explored above, *BnLM* and BnB also differ in pretraining objective: *BnLM* uses standard masked language modeling (MLM), whereas BnB uses ELECTRA's replaced-token-detection (RTD) objective [21]. We do not ablate this axis directly because RTD requires a generator-discriminator pairing whose joint training dynamics are tightly coupled to the objective itself; isolating RTD from its architectural prerequisites is not a single-axis intervention and would change several factors simultaneously. The existing comparison against BnB in Table 3, however, provides indirect evidence on this axis: after accounting for the tokenizer-and-corpus contribution that Config (ix) bounds at approximately +2.77 *BLUGE* points, the residual gap between *BnLM* and BnB in Table 3 reflects the joint effect of pretraining objective and any minor incidental architectural differences. Concretely, Config (ix) pairs the BnB SentencePiece tokenizer with the full *B-CORE* corpus under MLM and scores 86.74 *BLUGE*, only 0.33 points below BnB (87.07) despite using a 1.89× larger pretraining corpus; applying the corpus-scale main effect of +0.95 points estimated in Section VI-I7, a corpus-matched MLM counterpart would be expected to score approximately 85.8. This places the joint upper bound on the RTD objective and any incidental ELECTRA architectural differences at roughly +1.3 *BLUGE* points, roughly one-fifth of the tokenizer main effect (+5.84) and exceeding the corpus-scale main effect (+0.95) by a modest margin.

The pretraining objective is therefore the smallest of the three controllable axes, and the *BnLM* results are robust to this choice. We acknowledge a fully isolated objective ablation as outside the present scope and a subject for future work.

7) JOINT FACTORIAL STUDY

The pairwise comparisons in Table 10 establish that corpus scale, corpus quality, and tokenizer choice each independently affect performance, but do not directly quantify whether the corpus and tokenizer effects are additive or interactive. To address this, we conduct a 2×2 factorial study crossing two orthogonal axes within the mDB-base backbone family:

noitemsep, leftmargin=*

- **Corpus scale (C):** S = 27.5 GB cleaned BnB-equivalent subset of *B-CORE*; L = 52 GB full *B-CORE*, cleaned.
- **Tokenizer (T):** o = the original mDB multilingual tokenizer (119K); b = the *BnLM* Bengali WordPiece tokenizer (BnLM-120K).

This design fills out the natural extension of the pairwise ablation reported in Table 10. Three of the four cells reuse pretrainings already reported there: Configs (ii), (iii), and (iv) map directly to Cells 2, 3, and 4 respectively. Cell 4 coincides identically with *BnLM-F* (89.51), preserving numerical continuity with Table 3. Only one cell requires new pretraining: (S, o), corresponding to the mDB-base backbone with its original tokenizer paired with the 27.5 GB cleaned subset. All cells are reported as median over five seeds. Results appear in Table 11.

TABLE 11. 2×2 factorial ablation on the mDB-base backbone. C: corpus scale (S = 27.5 GB cleaned BnB-equivalent subset of *B-CORE*; L = 52 GB full *B-CORE*, cleaned). T: tokenizer (o = original mDB multilingual tokenizer (119K); b = *BnLM* Bengali WordPiece (BnLM-120K)). Cells marked † are reused from Table 10. Cell 4 coincides with *BnLM-F*. Δ is reported relative to *BnLM-F* (Cell 4).

Cell	C	T	*BLUGE*	Δ
1	S	o	82.72	−6.79
2	L	o †	83.21	−6.30
3	S	b †	88.10	−1.41
4	L	b (*BnLM-F*) †	**89.51**	—

8) MAIN EFFECTS

Estimating main effects via the standard 2×2 factorial decomposition yields a tokenizer main effect of +5.84 *BLUGE* points (o → b, averaged over both corpus levels) and a corpus-scale main effect of +0.95 (S → L, averaged

TABLE 12. Supplementary ablations on *BLUGE* (full finetuning; median over five seeds). *Top*: effect of Bengali tokenizer vocabulary size, reported alongside tokenization quality metrics from Table 2. *Bottom*: performance comparison between *BnLM-F* (15% masking) and *BnLM-M* (20% masking) under the same backbone, tokenizer, and corpus; note that additional hyperparameters (batch size, learning rate, weight decay, training steps) also differ between the two configurations, so the 0.32-point gap should not be attributed to masking rate alone.

(a) Bengali Tokenizer Vocabulary Size					
Tokenizer	**Vocab**	**Bengali Tok. Rep. (%)**	**Frag. Rate (%)**	**Avg. Tok. / Word**	**BLUGE Score**
mDB multilingual	119,547	0.44	74.65	3.09	81.30
BnLM-30.5K (Bengali)	30,522	80.22	37.32	1.64	88.56
BnLM-120K (Bengali)	120,000	83.80	31.63	1.51	**89.51**
(b) MLM Masking Rate					
Model	**Mask%**	**Score**	**NER**	**TSC**	**WSTP**
BnLM-F	15%	**89.51**	**92.11**	**83.88**	**88.02**
BnLM-M	20%	89.19	91.67	83.05	87.82

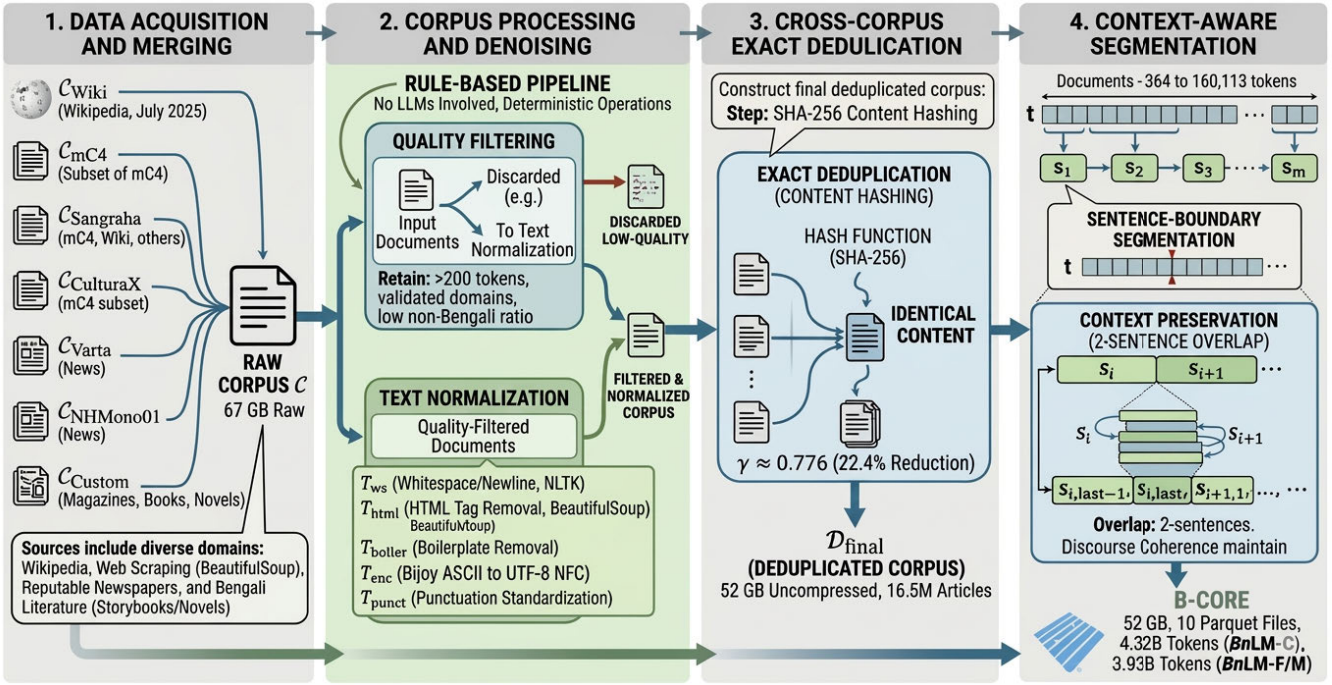

FIGURE 1. Schematic overview of the B-CORE corpus construction pipeline.

over both tokenizer levels). The tokenizer dominates by a wide margin, consistent with the pairwise analysis: the BnLM tokenizer reduces Bengali word fragmentation from 74.65% to 31.63%, enabling coherent morphological encoding that the additional pretraining data can then exploit. The corpus-scale effect, though smaller, is positive and consistent across both tokenizer conditions, confirming that the two factors are jointly sufficient and neither alone recovers full *BnLM-F* performance.

9) INTERACTION

The corpus-by-tokenizer interaction is +0.92 *BLUGE* (difference of corpus-scale gains across tokenizer levels: +1.41 at b minus +0.49 at o), confirming a positive interaction: a richer corpus disproportionately benefits the BnLM tokenizer because high-quality Bengali text exposes a broader distribution of morphological inflections that the custom subword inventory can exploit more efficiently than the multilingual original. This interaction is modest relative to the tokenizer main effect but substantiates the joint necessity claim: corpus and tokenizer reinforce one another, and each factor achieves its largest gain when the other is also optimized.

10) WHAT THIS RULES OUT

The factorial directly addresses the concern that *BnLM*'s improvement could be entirely explained by more data, entirely by the tokenizer, or by some uncontrolled combination. Cell 2 isolates the larger corpus alone with the original mDB tokenizer and falls −6.30 points short of

TABLE 13. Detailed quantitative and categorical comparisons between BnB and our proposed *BnLM-F/M/C* models are provided across corpus composition, evaluation benchmarks, tokenizers, and model characteristics.

Category	Aspect	BnB	BnLM-F/M/C	Distinctive Contributions
Corpus	Name	Bangla2B+	*B-CORE*	-
	Size	27.5 GB (before cleaning 35 GB)	52.0 GB (before cleaning 67 GB)	89% larger corpus; higher cleaning ratio
	Documents	7.18 Million	16.5 million	130% more documents
	Tokens	2.18 Billion	4.32 Billion	98% more tokens
	Data Sources	Wikipedia, Web crawl	Wikipedia, mC4, NHMono01, Sangraha, CulturaX, Varta, Books, Web crawl	Diversified sources
Evaluation Benchmark	Name	BLUB	*BLUGE*	-
	# Tasks	4	7	+3 evaluation tasks
	Task Types	Classification (binary), Inference, Sequence Labeling, QA	Classification (multi-class), Inference, Sequence Labeling, Linguistic Acceptability, Structure Prediction	More complex tasks
	# Sample (train/dev/test)	536295 (96.92%) / 7288 (1.32%) / 9766 (1.76%)	429,139 (80%) / 53,640 (10%) / 53,645 (10%)	Balanced 80/10/10 split ratio
	Class Balance	Partially Imbalanced (e.g SC tasks Train, Classes 0: 2894 , 1: 5133, 2: 4548)	Fully Balanced (e.g TSC, Train Set, Classes 0: 5945 , 1: 5946, 2: 5945)	Complete class balance in each tasks
	Format	Mixed (jsonl, json, csv, conll)	Uniform (parquet)	Standardized efficient format
	Dataset Curation	Direct adaptation of existing/translated datasets	Multi-stage pipeline: deduplication, length filtering, LLM correction, script/orthographic normalization	Rigorous data curation
Tokenizer	Vocab Size	32K	120K in *BnLM-F/M*, 30.5K in *BnLM-C*	Dual vocabulary experiment (∼4x larger vs compact)
	Tokenization Speed	15494.91 s/sec	21186.94 - 21798.31 s/sec	37-41% faster tokenization
	Avg. Tokens per Word	2.04	1.51 - 1.64	20-26% more efficient tokenization
	Bengali Unicode Script Coverage	57.812 %	96.875%	68% higher Unicode Script coverage
	Compression Ratio	0.66	0.68-0.73	Higher compression ratio
	Overall Tokenizer Performance (Table 2)	0 out of 13 metrics	13 out of 13 metrics	Comprehensive tokenization excellence (13/13 metrics)
Model	Architecture	ELECTRA	DistilBERT	-
	Training Objective	Replaced Token Detection (RTD)	Masked Language Modeling (MLM)	-
	BLUGE Performance	2 out of 7 tasks	5 out of 7 tasks	SOTA on 71.43% tasks
	XTREME, IndicGLUE, IndicXTREME, BLUB Evaluation	2 out of 9 tasks	7 out of 9 tasks	SOTA on 77.78% tasks
Category	**Aspect**	**BnB**	**BnLM-F/M/C**	**Distinctive Contributions**
	Avg. Inference Speed	0.42s (on PAN-X)	0.19s - 0.246s (on PAN-X)	up to 2.2× faster inference
	Avg. CUDA Usage	3.22 GB (on PAN-X)	1.64 GB - 2.27 GB (on PAN-X)	up to 49% lower GPU usage
	FLOP per Sample	596 million (on PAN-X)	213 million (on PAN-X)	∼64% reduction in computational cost

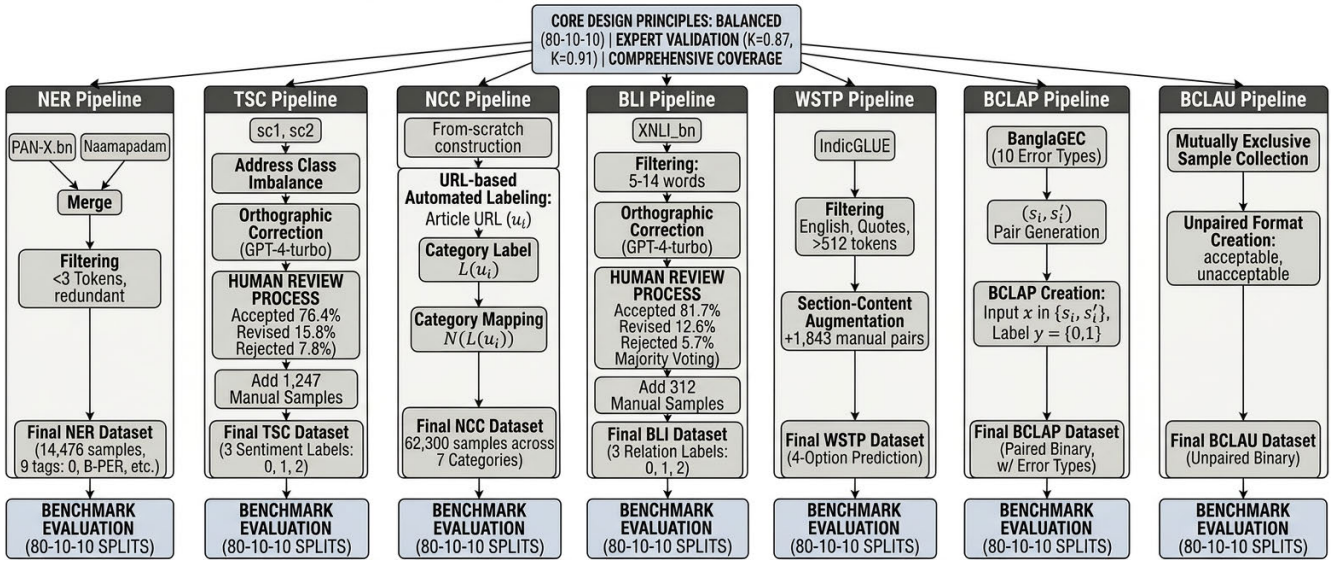

FIGURE 2. Structural overview of the BLUGE benchmark and its seven evaluation tasks.

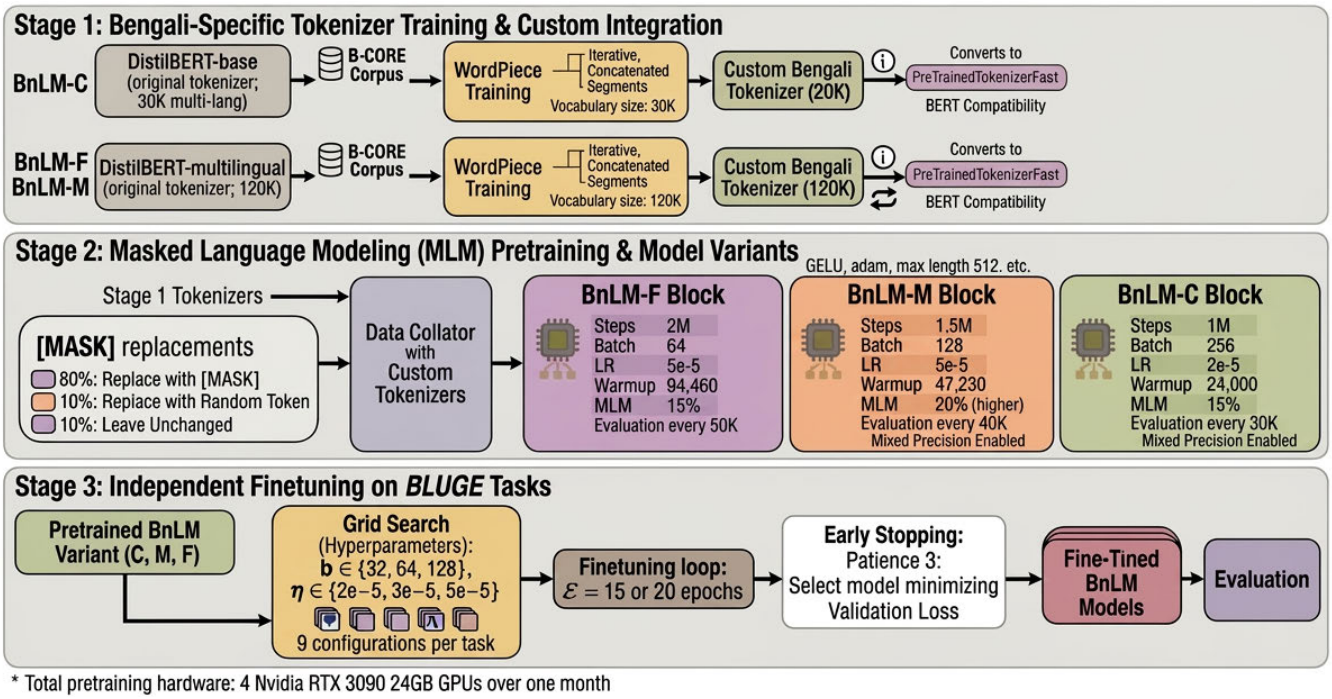

FIGURE 3. Overview of the BnLM pretraining architecture and three model configurations.

BnLM-F; Cell 3 isolates the BnLM tokenizer alone at the BnB-equivalent corpus scale and falls −1.41 points short of *BnLM-F*. No single-axis configuration recovers final *BnLM* performance, and the interaction (+0.92) is modest relative to both main effects, jointly excluding any reduction of the result to a single dominant cause. The architecture contribution is bounded implicitly within this design: Config (i) evaluates the mDB-base backbone off the shelf with no Bengali pretraining, directly characterising the backbone's performance ceiling under its original tokenizer. Because *BnLM-F* shares this exact backbone, every gain observed relative to Config (i) is attributable exclusively to corpus and tokenizer choices rather than to architectural changes.

11) VOCABULARY SIZE AND MLM MASKING RATE

Table 12 reports two supplementary ablations. The first contrasts the 30.5K Bengali vocabulary used by *BnLM-C* against the 120K vocabulary used by *BnLM-F* and *BnLM-M*,

(a) Morphological decomposition of five Bengali surface forms. Morpheme boundaries marked with +; word boundaries in compound constructions marked with |. PROG = progressive aspect; CV = converb; DEF.SG = definite singular classifier; GEN.PL = genitive plural; LOC = locative; 1P = first person; 3P.HON = third person honorific; PST = past tense.

Cat.	Surface Form	Morpheme Structure	Gloss
A	করছিলাম	কর+ছি+লাম	do+PROG+1P.PST
A	খাচ্ছিলাম	খা+চ্ছ+িলাম	eat+PROG+1P.PST
B	বলে দিচ্ছিলেন	বল+এ | দি+চ্ছ+িলেন	say+CV | give+PROG+3P.HON.PST
C	তাঁদের	তাঁ+দের	they.HON+GEN.PL
D	স্কুলটিতে	স্কুল+টি+তে	school+DEF.SG+LOC

(b) Verified subword segmentation comparing DB-base (backbone of *BnLM-C*, vocab size 30,522) and BnLM-C (vocab size 30,522). Subword boundaries marked with |; WordPiece continuation pieces prefixed with ##. DB-base fragments every form to character-level sequences due to absence of Bengali-specific training.

Surface Form	DB-base	BnLM-C
করছিলাম	ক | ##র | ##ছ | ##ি | ##ল | ##া | ##ম *[7 tokens: fully character-level]*	করছিলাম *[1 token]*
খাচ্ছিলাম	খ | ##া | ##চ | ##ছ | ##ি | ##ল | ##া | ##ম *[8 tokens: fully character-level]*	খা | ##চ্ছিলাম *[2 tokens]*
বলে দিচ্ছিলেন	ব | ##ল | ##ে | দ | ##ি | ##চ | ##ছ | ##ি | ##ল | ##ে | ##ন *[11 tokens: both components shattered]*	বলে | দিচ্ছিলেন *[2 tokens]*
তাঁদের	ত | ##া | ##দ | ##ে | ##র *[5 tokens: honorific diacritic lost]*	তাঁদের *[1 token]*
স্কুলটিতে	স | ##ক | ##ল | ##ট | ##ি | ##ত | ##ে *[7 tokens: fully character-level]*	স্কুল | ##টিতে *[2 tokens]*

FIGURE 4. Morphological tokenization comparison between DB-base and *BnLM-C*, which share an identical vocabulary size of 30,522 and differ only in tokenizer training data. (a) Morphological decomposition with Leipzig-style glosses. (b) Verified segmentation output. DB-base reduces every form to a character-level sequence regardless of morphological category, while *BnLM-C* preserves morphologically coherent units in all five cases.

tracing performance alongside tokenization quality metrics. The second compares *BnLM-F* (15% masking) and *BnLM-M* (20% masking), which share the same backbone (mDB-base), tokenizer (BnLM-120K), and corpus (*B-CORE*), but differ additionally in batch size (64 vs. 128), learning rate (5e-5 vs. 3e-5), weight decay (0.01 vs. 0.02), training steps (2M vs. 1.5M), warmup steps (94,460 vs. 47,230), and use of mixed-precision training; the observed 0.32-point gap should therefore be interpreted as the joint effect of these configuration differences, with masking rate as the primary intended variable.

12) VOCABULARY SIZE

Scaling the Bengali vocabulary from 30.5K to 120K improves Bengali token representation by 3.58 percentage points (80.22%→83.80%) and reduces word fragmentation by 5.69 points (37.32%→31.63%), yielding 1.51 tokens per word compared to 1.64 for the 30.5K vocabulary. These tokenization improvements translate to a gain of +0.95 points on the *BLUGE* benchmark (88.56→89.51). Crucially, the 30.5K vocabulary remains highly competitive despite using 75% fewer vocabulary parameters: it achieves 80.22% Bengali token representation, a 0.68 compression ratio, and zero OOV rate, all substantially superior to every baseline in Table 2. *BnLM-C* is therefore the preferred configuration for resource-constrained deployment, incurring only a −0.95 point downstream cost relative to the full 120K vocabulary model.

13) MASKING RATE

BnLM-F (15% masking) consistently outperforms *BnLM-M* (20% masking) by +0.32 points overall (89.51 vs. 89.19), with task-level advantages on NER (+0.44), TSC (+0.83), and WSTP (+0.20). The lower masking rate provides a better signal-to-noise balance for Bengali's agglutinative token distribution: at 15% masking, the model observes more complete contextual windows per training step, stabilising gradient updates over Bengali's compositional morpheme sequences. This finding is consistent with observations in pretraining studies for morphologically complex languages [43]. The absolute difference is small (under 0.4 points) and largely within normal variance, confirming that BnLM performance is robust across the 15%–20% masking range.

14) SUMMARY

The ablation experiments yield five consistent findings. First, tokenizer specialization is the primary driver of improvement, accounting for 82.8% of total gains in the pairwise breakdown and an averaged +5.84 *BLUGE* main effect in the factorial decomposition, achieved through a

(a) Morphological decomposition of five Bengali surface forms. Morpheme boundaries marked with +; word boundaries in compound constructions marked with |. PROG = progressive aspect; PRF = perfect aspect; CV = converb; DEF.SG = definite singular classifier; ACC = accusative; GEN.PL = genitive plural; 1P = first person; 3P.HON = third person honorific; PST = past tense.

Cat.	Surface Form	Morpheme Structure	Gloss
A	যাচ্ছিলাম	যা+চ্ছ+িলাম	go+PROG+1P.PST
A	দেখেছিলাম	দেখ+এছ+িলাম	see+PRF+1P.PST
B	বলে দিচ্ছিলেন	বল+এ | দি+চ্ছ+িলেন	say+CV | give+PROG+3P.HON.PST
C	তাঁদের	তাঁ+দের	they.HON+GEN.PL
D	বইটিকে	বই+টি+কে	book+DEF.SG+ACC

(b) Verified subword segmentation comparing mDB-base (backbone of *BnLM-F* and *BnLM-M*, vocab size 119,547) and BnLM-F/M (vocab size 120,000). Subword boundaries marked with |; WordPiece continuation pieces prefixed with ##. Error annotations: **(i)** hasanta diacritic isolated as a standalone token, severing the conjunct consonant cluster encoding the aspect morpheme; **(ii)** auxiliary verb component fragmented across multiple pieces; **(iii)** chandrabindu diacritic encoding honorific register isolated from its host token.

Surface Form	mDB-base	BnLM-F/M
যাচ্ছিলাম	যা | ##চ | ##্ | ##ছিল | ##াম *[5 tokens; error (i): ্ isolated, progressive cluster severed]*	যাচ্ছিলাম *[1 token]*
দেখেছিলাম	দ | ##ে | ##খে | ##ছিল | ##াম *[5 tokens; error (i): root and perfect suffix merged across morpheme boundary]*	দেখেছিলাম *[1 token]*
বলে দিচ্ছিলেন	বলে | দ | ##ি | ##চ | ##্ | ##ছিলেন *[6 tokens; errors (i)(ii): auxiliary shattered into five pieces]*	বলে | দিচ্ছিলেন *[2 tokens]*
তাঁদের	তা | ##ঁ | ##দের *[3 tokens; error (iii): ঁ isolated, honorific register marker detached]*	তাঁদের *[1 token]*
বইটিকে	ব | ##ই | ##ি | ##ে *[4 tokens: definiteness and accusative suffixes fragmented]*	বইটিকে *[1 token]*

FIGURE 5. Morphological tokenization comparison between mDB-base (vocab 119,547) and *BnLM-F/M* (vocab 120,000). (a) Morphological decomposition with Leipzig-style glosses. (b) Verified segmentation output annotated with three error categories confirmed in mDB-base output: (i) hasanta diacritic isolated as a standalone token, severing the conjunct consonant cluster encoding the aspect morpheme; (ii) auxiliary verb component fragmented across multiple pieces; (iii) chandrabindu diacritic encoding honorific register isolated from its host token.

reduction in Bengali word fragmentation from 74.65% to 31.63% and a proportional improvement in sequence compression. Second, the pretraining corpus provides an orthogonal and additive contribution of +1.91 points (with the original tokenizer) and +1.41 points (with the BnLM tokenizer) in the pairwise comparisons, corresponding to a factorial-averaged main effect of +0.95 *BLUGE*, that persists across all task categories independently of tokenizer configuration. Third, corpus quality contributes an additional −1.73 *BLUGE* points worth of value at fixed corpus size, demonstrating that the gains from *B-CORE* are not reducible to scale alone but also reflect the systematic denoising pipeline of Section III-B. Fourth, training the tokenizer on *B-CORE* itself yields a further +2.77 *BLUGE* points beyond the use of BnB's publicly released Bengali tokenizer at the same backbone and corpus, isolating the value of corpus-tokenizer distributional alignment from Bengali specialization in general. Fifth, the corpus-scale and tokenizer effects interact positively (+0.92 in the factorial), with the larger corpus disproportionately benefiting the *BnLM* tokenizer, confirming that no single factor is individually sufficient to reproduce the reported *BnLM-F* performance level, but that together they are jointly sufficient for it.

J. ERROR ANALYSIS AND FAILURE MODES

Despite achieving state-of-the-art performance on five of seven *BLUGE* tasks and the highest aggregate score across XTREME, IndicGLUE, IndicXTREME, and BLUB, *BnLM* exhibits consistent and interpretable failure patterns on two specific task families and under low-data conditions. We identify three failure modes, each attributable to a specific interaction between model design and task structure.

1) TASK-STRUCTURAL FAILURES: WSTP AND BLI

Under full finetuning, *BnLM-F* trails BnB by 1.24 points on WSTP (88.02 vs. 89.26) and 0.82 points on BLI (82.09 vs. 82.91), constituting the only two tasks where *BnLM-F* does not establish state-of-the-art performance on *BLUGE*. Both failures are structurally linked to the MLM pretraining objective.

WSTP requires selecting the correct section title from four candidates given a paragraph-length section body, demanding hierarchical discourse-level understanding of how a passage relates to a concise nominal phrase. BLI requires modeling cross-sentence logical entailment, a relation primarily governed by inter-sentential semantic scope rather than local contextual cues. BnB's ELECTRA-based Replaced Token Detection (RTD) objective trains a discriminator

to classify every input token as original or replaced, providing dense supervision across the full input sequence in every training step. This dense signal may yield stronger sensitivity to global coherence and entailment structure. In contrast, standard MLM at 15% masking rate predicts approximately 77 tokens per 512-token sequence, limiting per-step exposure to long-range discourse dependencies. The ablation in Subsection VI-I11 confirms that varying the masking rate within the MLM family (15% vs. 20%) changes the overall *BLUGE* score by only 0.32 points, indicating that the RTD–MLM objective gap is not recoverable through masking rate adjustment alone. Hybrid pretraining objectives incorporating MLM alongside sentence-level discriminative signals represent a natural direction for closing this gap.

2) FEW-SHOT FAILURES: BCLAP AND BCLAU

In the 5-shot evaluation setting, *BnLM-F* scores 52.59 on BCLAP and 52.62 on BCLAU, trailing BnB by 3.17 and 4.20 points (55.76 and 56.82, respectively). This ordering inverts sharply under full finetuning, where *BnLM-F* outperforms BnB by 3.71 points on BCLAP (91.54 vs. 87.83) and 3.72 points on BCLAU (92.03 vs. 88.31).

The few-shot disadvantage is attributable to the nature of grammatical acceptability judgments. Discriminating between grammatically acceptable and unacceptable Bengali sentences requires fine-grained sensitivity to subtle morphosyntactic violations, including verb inflection agreement, honorificity mismatch, and auxiliary deletion, distributed across a long tail of low-frequency constructions. BnB's RTD objective, which provides explicit binary correctness signals at every token position, may establish a stronger zero-shot prior for grammaticality than MLM's masked prediction of 15% of tokens. Under full finetuning, *BnLM*'s advantages in corpus scale (1.89× larger), tokenization quality (31.63% vs. 38.48% fragmentation rate), and morphological coverage fully recover and substantially exceed BnB's few-shot prior, as the larger annotated training set provides sufficient supervision to override the pretraining objective's distributional bias.

3) SCALE-INSENSITIVE FAILURES OF LARGE LANGUAGE MODELS

Table 8 establishes that parameter scale alone cannot compensate for Bengali-specific tokenization and pretraining deficiencies. LLaMA-3.1-8B (8B parameters; approximately 59× larger than *BnLM-F*) achieves 13.38 F1 on NER under zero-shot prompting and 18.45 under 5-shot prompting, a gap of 73.66 F1 points below *BnLM-F* under full finetuning (92.11). On NCC, LLaMA achieves 43.36% accuracy under zero-shot and 49.20% under 5-shot, compared to *BnLM-F*'s 62.51% under 5-shot, representing a 13.31-point gap in the same prompting regime, and 96.92% under full finetuning. BLOOMZ-7B1 (7B parameters) performs weakest across all tasks, scoring 11.78 F1 on NER (zero-shot) and 6.03%/20.52% macro-F1/accuracy on NCC (zero-shot), confirming that general-purpose multilingual pretraining does not transfer effectively to Bengali classification tasks regardless of scale.

The primary failure mechanism is tokenizer-level. As Table 2 shows, models without dedicated Bengali tokenization achieve Bengali script coverage below 61%, fragmenting morphologically complex forms into character-level sequences that carry no recoverable morphological interpretation for the attention mechanism, regardless of the number of parameters processing them. This analysis reinforces the central thesis of this work: in low-resource Bengali NLP, the bottleneck is not model scale but the alignment of tokenization and pretraining with the target language's morphological properties.

K. DEPLOYMENT GUIDANCE AND PRACTICAL RECOMMENDATIONS

The three artifacts serve distinct roles in real-world Bengali NLP deployment. For *B-CORE*, continued pretraining from an existing *BnLM* checkpoint is recommended over training from scratch, as the denoised corpus and pretrained representations substantially reduce compute requirements for domain adaptation in healthcare, legal, and financial document processing. *BLUGE*'s contamination-audited, balanced splits make it directly adoptable for model selection and regression testing in production pipelines without additional preprocessing. For inference deployment, *BnLM-C* (66.4M parameters, 1,637 MB GPU memory, 0.19 s latency) is recommended for latency-sensitive applications such as content moderation, news categorization, and grammar checking, and runs on a single consumer-grade GPU. *BnLM-F* and *BnLM-M* (135.1M parameters, 2,269 MB GPU memory) are better suited for precision-critical offline pipelines such as legal and clinical document analysis, where the accuracy gain over *BnLM-C* justifies the additional memory overhead. Fine-tuning any *BnLM* variant requires a single 24 GB GPU, making task adaptation accessible without specialized infrastructure.

VII. CONCLUSION

This work demonstrates that the bottleneck in low-resource NLP is not model scale but the principled alignment of corpus quality, tokenization, and evaluation infrastructure with the target language's linguistic properties. For Bengali, an agglutinative, morphologically rich language underserved by multilingual NLP pipelines, we addressed this alignment across all three dimensions simultaneously. *B-CORE* provides a reproducible, rigorously curated 52 GB pretraining corpus of 16.5M documents, achieving a 22.4% volume reduction through systematic quality filtering, normalization, and cross-corpus deduplication. Our morphology-aware WordPiece tokenizers attain 96.9% Bengali script coverage and reduce word fragmentation from 74.65% to 31.63%, enabling coherent morphological encoding of Bengali's compositional structure. *BLUGE* establishes the first balanced, contamination-audited Bengali NLU benchmark, spanning

7 tasks and 536K samples with standardized splits and expert-validated annotations.

Built on these foundations, the *BnLM* suite (66–135M parameters) achieves state-of-the-art results on five of seven *BLUGE* tasks, with *BnLM-F* attaining an aggregate score of 89.51% against 87.07% for the strongest prior Bengali model, and the highest aggregate score (85.64%) across XTREME, IndicGLUE, IndicXTREME, and BLUB. These gains are obtained while requiring 44–91% fewer FLOPs and delivering 2× faster inference than competitive multilingual baselines, despite operating at a fraction of their parameter counts. A controlled 2×2 factorial ablation decomposes these gains: tokenizer specialization contributes a +5.84 *BLUGE* main effect, corpus scale contributes +0.95, the curation pipeline adds an independent +1.73 at fixed corpus size, and a positive corpus–tokenizer interaction of +0.92 confirms that no single axis is individually sufficient while the three together are jointly sufficient to reproduce *BnLM-F*'s performance.

These results offer a replicable methodological template for low-resource language modeling: targeted investment in language-specific corpus curation, tokenization, and evaluation infrastructure delivers superior performance–efficiency trade-offs to scaling multilingual models. We identify three directions for future work: validating cross-lingual generalization by extending the full pipeline (*B-CORE*-style curation, morphology-aware tokenizer training, and *BLUGE*-style benchmark construction) to other morphologically complex, low-resource languages such as Odia, Assamese, and Sinhala; investigating hybrid MLM–RTD pretraining objectives to close the residual gaps on hierarchical and entailment tasks (WSTP, BLI); and broadening *BLUGE* to reading comprehension, dialogue, and code-switching phenomena.

LIMITATIONS

The models exhibit minor underperformance on WSTP and BLI tasks, suggesting potential benefits from hybrid pretraining objectives. Despite extensive curation, *B-CORE* may contain biases inherent in web-scraped content. Additionally, as evaluation datasets reflect standard written Bengali, generalizability to dialects and informal text varieties remains unexplored. Beyond the task-level performance, deeper evaluation axes including model calibration (e.g., Expected Calibration Error), performance fairness across demographic or dialectal subgroups, and adversarial robustness represent important directions not addressed in the current work and are deferred to future study. The robustness analysis provided in Tables 9 and 5 partially addresses data-quality and contamination robustness, but does not substitute for these deeper evaluations.

ETHICAL CONSIDERATIONS

The web-scraped nature of *B-CORE* may perpetuate societal biases present in online content. Users should exercise caution when deploying models trained on *B-CORE* in sensitive contexts and remain aware of potential biases in model outputs, particularly in applications affecting underrepresented communities. All datasets incorporated into *B-CORE* are used under their respective open licenses or standard research-use terms (see Section III for full details). Custom web crawling adhered strictly to each target site's `robots.txt` directives, enforced crawl delays of no less than 5 seconds per domain, and excluded paywalled or access-restricted content. *B-CORE* is released strictly for non-commercial research purposes, and we are committed to honoring content removal requests from rights holders on an ongoing basis.

Web-scraped corpora may contain user-generated content from individuals who did not explicitly consent to inclusion in an NLP training corpus. While our custom-scraped material constitutes a small fraction of *B-CORE*, with the majority drawn from established, openly licensed research datasets, we acknowledge this limitation and encourage users to remain mindful of it when deploying derived models in production settings. Like all large language corpora, *B-CORE* carries potential for misuse, including the generation of misinformation, spam, or harmful content in Bangla. We release this resource strictly for research purposes and urge practitioners to conduct appropriate risk assessments before deploying models trained on *B-CORE* in real-world applications. Responsibility for downstream use rests with the deploying parties.

REPRODUCIBILITY

The sources, compilation strategies, and curation strategies of both *B-CORE* and *BLUGE* are fully described in Sections III and IV. Subsections "V-C Pretraining Setup" and "V-D Finetuning" provide explicit values for all major hyperparameters, including learning rate, batch size, weight decay, warmup steps, masking rate, optimizer, activation function, maximum sequence length, and evaluation frequency. Performance may vary with random initialization, sampling procedures, and hardware.

ARTIFACT AVAILABILITY

All artifacts, including the corpus *B-CORE*, the benchmark *BLUGE* (seven tasks), and the *BnLM* models (*BnLM-F*, *BnLM-M*, and *BnLM-C*), can be found at https://github.com/bnlm-project.

APPENDIX A
EXTENDED QUANTITATIVE AND COMPARATIVE RESULTS

See Tables 13.

APPENDIX B
QUALITY ASSURANCE PROTOCOL FOR *BLUGE* TSC AND BLI CONSTRUCTION

This appendix provides the prompts used during the curation of the TSC and BLI tasks within *BLUGE*. Although the source corpora for both tasks are drawn from well-established, highly reputed datasets [13], [29], [30],

we applied an additional LLM-assisted quality assurance pipeline to ensure the highest possible data quality for a standardized evaluation benchmark. This pipeline consists of two sequential stages for each task: (1) a *flagging* stage, in which GPT-4-turbo was applied to *all* samples to identify those containing errors, and (2) a *correction* stage, in which GPT-4-turbo was applied exclusively to flagged samples to propose corrections. In both stages, GPT-4-turbo was queried with temperature $= 0$ to ensure fully deterministic output. All correction proposals were subsequently presented to native Bengali linguists for independent validation without revealing whether a change had been made, to prevent anchoring bias. GPT-4-turbo played no role in class label assignment for either task at any stage.

A. TSC: TERNARY SENTIMENT CLASSIFICATION

The TSC quality assurance pipeline addresses orthographic errors commonly found in informal Bengali text, including irregular Unicode characters, non-standard Hasanta usage and broken words, missing Bengali sentence-final period symbols (U+0964), and extraneous whitespace. The pipeline proceeds in two stages as described below.

1) STAGE 1: FLAGGING PROMPT (APPLIED TO ALL TSC SAMPLES)

This prompt was applied to every sample in the merged TSC corpus. Samples for which the model returned `"flagged": true` were forwarded to the correction stage. This process identified 7.23% of samples as containing one or more orthographic issues.

System Prompt:

```
You are an expert Bengali
  computational linguist. Your task is to
inspect the following Bengali sentence and
determine whether it contains any orthographic
errors. The orthographic errors you should look
for are:(1) irregular
  or incorrect Unicode characters that do not
belong to standard Bengali script,
(2) non-standard Hasanta usage or broken words
caused by incorrect character
  composition, (3) a missing Bengali
  sentence-final period symbol (U+0964) where
one is grammatically
  expected, and (4) extraneous or irregular
whitespace between characters or words. Do not
flag informal language, colloquial expressions,
or stylistic choices as errors. Do not flag
grammatical or semantic issues. Only flag the
sentence if it contains at least one of the
four orthographic error types listed above.
Return your response strictly in the following
JSON format with no
additional text, explanation, or preamble:
  {"flagged": true/false, "reasons":
  ["<reason1>", "<reason2>", ...]}
  If the sentence is not flagged, return an
empty list for reasons:
  {"flagged": false, "reasons": []}
```

User Prompt Template:

```
  Sentence: {sentence}
```

where `{sentence}` is replaced with the Bengali sample under inspection. The `reasons` field was logged internally for quality monitoring but was not shown to human reviewers during the correction validation stage.

2) STAGE 2: CORRECTION PROMPT (APPLIED TO FLAGGED TSC SAMPLES ONLY)

This prompt was applied exclusively to the 7.23% of samples identified as flagged in Stage 1. Only the input text was provided to the model; no sentiment label or class information was included in the prompt. Corrections were proposed by GPT-4-turbo based solely on the surface form of the input text, ensuring that label information could not influence the correction.

System Prompt:

```
You are an expert Bengali computational linguist
specializing in orthographic
  normalization of Bengali text. You will be
given a Bengali sentence that has been
identified as containing one or more
  orthographic errors. Your sole task is to
correct the following categories of orthographic
errors: (1) irregular or incorrect Unicode
  characters, (2) non-standard Hasanta usage
or broken words caused by incorrect character
composition, (3) a missing Bengali
sentence-final period symbol (U+0964) where
one is
  grammatically expected, and (4) extraneous
or irregular whitespace between characters or
words. You must strictly preserve the original
meaning, sentiment polarity, register, and
syntactic structure of the sentence. Do not
paraphrase, restructure, expand, or shorten
the sentence. Do not correct grammatical errors,
stylistic choices, or informal language unless
they involve one of the four orthographic error
categories listed above. Do not add, remove,
or infer any content word, modifier, or
sentiment-bearing expression. Return your
response strictly in the following JSON format
with no additional text, explanation, or
preamble: {"corrected": "<corrected Bengali
  sentence>", "changed": true/false}
  If no correction was necessary
  despite the sentence being flagged, return
the original sentence unchanged and set changed
to false.
```

User Prompt Template:

```
  Sentence: {sentence}
```

where `{sentence}` is replaced with the flagged Bengali sample. No label or metadata was included in the user prompt. The `changed` field allowed human reviewers to immediately identify whether GPT-4-turbo proposed any modification, while the `corrected` sentence was presented for validation regardless of the `changed` value. To enable this verification, reviewers were shown the original flagged sentence alongside the `corrected` output during validation. Reviewers were instructed to verify that no sentiment-bearing word, modifier, or negation had been altered during correction.

B. BLI: BENGALI LANGUAGE INFERENCE

The BLI quality assurance pipeline addresses a broader range of errors than TSC, as the source corpus [13] consists of sentences translated into Bengali from English XNLI. Such translation pipelines can introduce not only orthographic issues but also grammatical inaccuracies and syntactic irregularities that affect naturalness without necessarily altering the entailment label. The BLI pipeline therefore targets orthographic, grammatical, and syntactic errors simultaneously. The pipeline proceeds in two stages as described below.

1) STAGE 1: FLAGGING PROMPT (APPLIED TO ALL BLI SAMPLES)

This prompt was applied to every premise-hypothesis pair in the BLI corpus. Pairs for which the model returned `"flagged": true` for either the premise or the hypothesis were forwarded to the correction stage. This process identified 3.47% of samples as containing one or more errors.

System Prompt:

```
You are an expert Bengali computational linguist.
You will be given a premise
  -hypothesis pair in Bengali. Your task is
to inspect each sentence independently and
determine whether either sentence contains any
of the following categories of errors:
(1) orthographic errors, including irregular
or incorrect Unicode characters,
  non-standard Hasanta usage or broken words,
missing Bengali sentence-final period symbols
(U+0964), and extraneous whitespace,
(2) grammatical errors
  introduced during translation from English,
such as incorrect verb inflection, subject-verb
  disagreement, or incorrect use of case markers,
and (3) syntactic errors that produce unnatural
or malformed Bengali sentence
  structure, such as incorrect word order or
missing
  obligatory constituents. Do not flag informal
language, colloquial expressions, or stylistic
variation as errors. Do not flag a sentence
solely on the basis of it being a simple or
short sentence. Only flag a sentence if it
contains at least one error from the three
categories listed above. Return your response
strictly in the following JSON format with no
  additional text, explanation, or preamble:
  {"premise_flagged": true/false,
  "premise_reasons": ["<reason1>",
  "<reason2>", ...], "hypothesis_flagged": true/
false, "hypothesis_reasons":
  ["<reason1>", "<reason2>", ...]}
  If a sentence is not flagged, return an empty
list for its reasons.
```

User Prompt Template:

```
Premise: {premise}
Hypothesis: {hypothesis}
```

where `{premise}` and `{hypothesis}` are replaced with the respective Bengali sentences. The `reasons` fields were logged internally for quality monitoring and provided human reviewers with targeted guidance on which specific issues to inspect during the correction validation stage.

2) STAGE 2: CORRECTION PROMPT (APPLIED TO FLAGGED BLI SAMPLES ONLY)

This prompt was applied exclusively to the 3.47% of premise-hypothesis pairs identified as flagged in Stage 1. As with TSC, only the input text was provided to the model; no entailment label or class information was included in the prompt. Corrections spanning orthographic, grammatical, and syntactic dimensions were proposed by GPT-4-turbo based solely on the surface form of the input sentences.

System Prompt:

```
You are an expert Bengali computational linguist.
You will be given a premise
  -hypothesis pair in Bengali that has been
identified as containing one or more errors.
Your task is to correct errors in the following
  categories: (1) orthographic errors, including
irregular or incorrect Unicode characters,
non-standard Hasanta usage or broken words,
missing Bengali sentence
  -final period symbols (U+0964), and extraneous
whitespace,
  (2) grammatical errors introduced during
translation from English, such as incorrect
verb inflection, subject-verb disagreement,
or incorrect use of case markers, and
(3) syntactic errors that produce unnatural
or malformed Bengali sentence structure. You
must strictly preserve the original meaning
of each sentence and the logical relationship
between the premise and hypothesis. Do not
paraphrase, restructure, or alter the intended
meaning of either sentence. Do not add, remove,
or modify any content word, negation, quantifier,
or modifier, as these directly determine the
logical relationship between the premise and
the hypothesis. Treat each sentence independently
during correction but remain aware that
  alterations to either sentence must not disturb
the logical relationship between them. If a
sentence contains no error despite being part
of a flagged pair, return it unchanged. Return
your response strictly in the following JSON
format with no additional text, explanation,
or preamble:
  {"corrected_premise": "<corrected
  Bengali premise>", "corrected_hypothesis":
"<corrected Bengali hypothesis>",
  "premise_changed": true/false,
  "hypothesis_changed": true/false}
```

User Prompt Template:

```
Premise: {premise}
Hypothesis: {hypothesis}
```

where `{premise}` and `{hypothesis}` are replaced with the respective flagged Bengali sentences. No entailment label or metadata was included in the user prompt. The `premise_changed` and `hypothesis_changed` fields allowed human reviewers to identify which sentence or sentences were modified, while both corrected sentences were presented for validation regardless of the changed values. Reviewers were specifically instructed to verify that no negation, quantifier, or content word had been added, removed, or substituted, as any such change would invalidate the original entailment label.

APPENDIX C
SYSTEM DIAGRAMS

The following figures provide schematic overviews of the three primary contributions described in this paper.

A. B-CORE CORPUS PIPELINE

Figure 1 illustrates the end-to-end B-CORE construction pipeline, from multi-source data acquisition through quality filtering, text normalization, cross-corpus deduplication, and context-aware segmentation into the final Parquet-serialized corpus.

B. BLUGE BENCHMARK STRUCTURE

Figure 2 depicts the composition of the BLUGE benchmark, summarizing the seven constituent tasks, their data sources, and the annotation and balancing procedures applied to each.

C. BnLM PRETRAINING ARCHITECTURE

Figure 3 summarizes the BnLM model family, covering the three training configurations (*BnLM-F*, *BnLM-M*, *BnLM-C*), their respective tokenizer setups, and the pretraining and finetuning workflow.

APPENDIX D
MORPHOLOGICAL TOKENIZATION EXAMPLES

Figures 4 and 5 provide verified tokenizer output for the two model pairs in *BnLM*. Examples are selected from a curated Bengali morphological test set covering agglutinative verb inflection (Category A), compound serial verb constructions (Category B), honorific pronoun paradigms (Category C), and nominal case marking with definiteness morphology (Category D). All token sequences are verified directly from the released tokenizers.

REFERENCES

[1] A. Salleh, M. H. Osman, S. Hassan, M. Y. Said, K. Y. Sharif, and K. T. Wei, "A hybrid model for low-resource language text classification and comparative analysis," *Knowl.-Based Syst.*, vol. 326, Sep. 2025, Art. no. 114068. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S095070512501113X

[2] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics, Hum. Lang. Technol.*, vol. 1, Jun. 2019, pp. 4171–4186. [Online]. Available: https://aclanthology.org/N19-1423

[3] 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 *Proc. 58th Annu. Meeting Assoc. Comput. Linguistics*, Jul. 2020, pp. 8440–8451. [Online]. Available: https://aclanthology.org/2020.acl-main.747

[4] M. Osama, A. Dey, K. Ahmed, and M. A. Kabir, "BeliN: A novel corpus for Bengali religious news headline generation using contextual feature fusion," *Natural Lang. Process. J.*, vol. 11, Jun. 2025, Art. no. 100138. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2949719125000147

[5] N. Muennighoff, T. Wang, L. Sutawika, A. Roberts, S. Biderman, T. Le Scao, M. S. Bari, S. Shen, Z. X. Yong, H. Schoelkopf, X. Tang, D. Radev, A. F. Aji, K. Almubarak, S. Albanie, Z. Alyafeai, A. Webson, E. Raff, and C. Raffel, "Crosslingual generalization through multitask finetuning," in *Proc. 61st Annu. Meeting Assoc. Comput. Linguistics*, Jul. 2023, pp. 15991–16111. [Online]. Available: https://aclanthology.org/2023.acl-long.891/

[6] N. Hossain, S. Islam, and M. N. Huda, "Development of Bangla spell and grammar checkers: Resource creation and evaluation," *IEEE Access*, vol. 9, pp. 141079–141097, 2021.

[7] M. Khan, P. Mehta, A. Sankar, U. Kumaravelan, S. Doddapaneni, B. Suriyaprasaad, G. Varun, S. Jain, A. Kunchukuttan, P. Kumar, R. Dabre, and M. Khapra, "IndicLLMSuite: A blueprint for creating pre-training and fine-tuning datasets for Indian languages," in *Proc. 62nd Annu. Meeting Assoc. Comput. Linguistics*, Aug. 2024, pp. 15831–15879. [Online]. Available: https://aclanthology.org/2024.acl-long.843/

[8] T. Nguyen, C. V. Nguyen, V. D. Lai, H. Man, N. T. Ngo, F. Dernoncourt, R. A. Rossi, and T. H. Nguyen, "CulturaX: A cleaned, enormous, and multilingual dataset for large language models in 167 languages," in *Proc. Lang. Resour. Eval. Conf.*, May 2024, pp. 4226–4237. [Online]. Available: https://aclanthology.org/2024.lrec-main.377/

[9] R. Aralikatte, Z. Cheng, S. Doddapaneni, and J. C. K. Cheung, "Varta: A large-scale headline-generation dataset for Indic languages," in *Proc. Findings Assoc. Comput. Linguistics, ACL*, Jul. 2023, pp. 3468–3492. [Online]. Available: https://aclanthology.org/2023.findings-acl.215/

[10] J. Hu, S. Ruder, A. Siddhant, G. Neubig, O. Firat, and M. Johnson, "XTREME: A massively multilingual multi-task benchmark for evaluating cross-lingual generalisation," in *Proc. 37th Int. Conf. Mach. Learn.*, vol. 119, pp. 4411–4421. [Online]. Available: https://proceedings.mlr.press/v119/hu20b.html

[11] D. Kakwani, A. Kunchukuttan, S. Golla, N. C. Gokul, A. Bhattacharyya, M. M. Khapra, and P. Kumar, "IndicNLPSuite: Monolingual corpora, evaluation benchmarks and pre-trained multilingual language models for Indian languages," in *Proc. Findings Assoc. Comput. Linguistics, EMNLP*, Nov. 2020, pp. 4948–4961. [Online]. Available: https://aclanthology.org/2020.findings-emnlp.445

[12] S. Doddapaneni, R. Aralikatte, G. Ramesh, S. Goyal, M. M. Khapra, A. Kunchukuttan, and P. Kumar, "Towards leaving no Indic language behind: Building monolingual corpora, benchmark and models for Indic languages," in *Proc. 61st Annu. Meeting Assoc. Comput. Linguistics*, Jul. 2023, pp. 12402–12426. [Online]. Available: https://aclanthology.org/2023.acl-long.693/

[13] A. Bhattacharjee, T. Hasan, W. Ahmad, K. S. Mubasshir, M. S. Islam, A. Iqbal, M. S. Rahman, and R. Shahriyar, "BanglaBERT: Language model pretraining and benchmarks for low-resource language understanding evaluation in Bangla," in *Proc. Findings Assoc. Comput. Linguistics, NAACL*, Jul. 2022, pp. 1318–1327.

[14] P. Bhattacharyya, J. Mondal, S. Maji, and A. Bhattacharya, "VACASPATI: A diverse corpus of Bangla literature," in *Proc. 13th Int. Joint Conf. Natural Lang. Process. 3rd Conf. Asia–Pacific Chapter Assoc. Comput. Linguistics*, Nov. 2023, pp. 1118–1130. [Online]. Available: https://aclanthology.org/2023.ijcnlp-main.72/

[15] A. Akther, M. S. Islam, H. Sultana, A. K. Z. R. Rahman, S. Saha, K. M. Alam, and R. Debnath, "Compilation, analysis and application of a comprehensive Bangla corpus KUMono," *IEEE Access*, vol. 10, pp. 79999–80014, 2022.

[16] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *Proc. 31st Int. Conf. Neural Inf. Process. Syst.*, 2017, pp. 6000–6010.

[17] V. Sanh, L. Debut, J. Chaumond, and T. Wolf, "DistilBERT, a distilled version of BERT: Smaller, faster, cheaper and lighter," 2019, *arXiv:1910.01108*.

[18] V. Dankers and V. Raunak, "Memorization inheritance in sequence-level knowledge distillation for neural machine translation," in *Proc. 63rd Annu. Meeting Assoc. Comput. Linguistics (Volume 2: Short Papers)*, Jul. 2025, pp. 760–774. [Online]. Available: https://aclanthology.org/2025.acl-short.61/

[19] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, "RoBERTa: A robustly optimized BERT pretraining approach," 2019, *arXiv:1907.11692*.

[20] Z. Lan, M. Chen, S. Goodman, K. Gimpel, P. Sharma, and R. Soricut, "ALBERT: A lite BERT for self-supervised learning of language representations," in *Proc. 8th Int. Conf. Learn. Represent.*, 2020, pp. 1–11. [Online]. Available: https://openreview.net/forum?id=H1eA7AEtvS

[21] K. Clark, M.-T. Luong, Q. V. Le, and C. D. Manning, "ELECTRA: Pre-training text encoders as discriminators rather than generators," in *Proc. Int. Conf. Learn. Represent.*, 2020, pp. 1–11. [Online]. Available: https://openreview.net/forum?id=r1xMH1BtvB

[22] M. Diskin, A. Bukhtiyarov, M. Ryabinin, L. Saulnier, Q. Lhoest, A. Sinitsin, D. Popov, D. Pyrkin, M. Kashirin, A. Borzunov, A. V. D. Moral, D. Mazur, I. Kobelev, Y. Jernite, T. Wolff, and G. Pekhimenko, "Distributed deep learning in open collaborations," in *Proc. 35th Int. Conf. Neural Inf. Process. Syst.*, 2021, pp. 7879–7897, doi: 10.5555/3540261.3540864.

[23] M. Brack, M. Ostendorff, P. O. Suarez, J. J. Saiz, I. L. Castilla, J. Palomar-Giner, A. Shvets, P. Schramowski, G. Rehm, M. Villegas, and K. Kersting, "Community OSCAR: A community effort for multilingual Web data," in *Proc. 4th Workshop Multilingual Represent. Learn. (MRL)*, Nov. 2024, pp. 232–235. [Online]. Available: https://aclanthology.org/2024.mrl-1.19/

[24] R. Kimera, D. Heo, D. N. Rim, and H. Choi, "Data augmentation with back translation for low resource languages: A case of English and Luganda," in *Proc. 8th Int. Conf. Natural Lang. Process. Inf. Retr.*, Dec. 2024, pp. 142–148, doi: 10.1145/3711542.3711594.

[25] L. Soldaini et al., "Dolma: An open corpus of three trillion tokens for language model pretraining research," in *Proc. 62nd Annu. Meeting Assoc. Comput. Linguistics*, 2024, pp. 15725–15788. [Online]. Available: https://aclanthology.org/2024.acl-long.840

[26] S. Baack, "A critical analysis of the largest source for generative AI training data: Common crawl," in *Proc. ACM Conf. Fairness, Accountability, Transparency*, Jun. 2024, pp. 2199–2208, doi: 10.1145/3630106.3659033.

[27] I. Ilyankou, M. Wang, S. Cavazzi, and J. Haworth, "CC-GPX: Extracting high-quality annotated geospatial data from common crawl," in *Proc. 32nd ACM Int. Conf. Adv. Geographic Inf. Syst.*, Oct. 2024, pp. 693–696, doi: 10.1145/3678717.3691215.

[28] M. M. Mahtab, F. A. Khan, M. E. Islam, M. S. M. Chowdhury, L. I. Chowdhury, S. Afrin, H. Ali, M. M. O. Rashid, N. Mohammed, and M. R. Amin, "BanNERD: A benchmark dataset and context-driven approach for Bangla named entity recognition," in *Proc. Findings Assoc. Comput. Linguistics, NAACL*, Apr. 2025, pp. 6807–6828. [Online]. Available: https://aclanthology.org/2025.findings-naacl.380/

[29] M. Kabir, O. Bin Mahfuz, S. R. Raiyan, H. Mahmud, and M. K. Hasan, "BanglaBook: A large-scale Bangla dataset for sentiment analysis from book reviews," in *Proc. Findings Assoc. Comput. Linguistics, ACL*, Jul. 2023, pp. 1237–1247. [Online]. Available: https://aclanthology.org/2023.findings-acl.80/

[30] M. S. Islam and K. M. Alam, "Sentiment analysis of Bangla language using a new comprehensive dataset BangDSA and the novel feature metric skipBangla-BERT," *Natural Lang. Process. J.*, vol. 7, Jun. 2024, Art. no. 100069. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2949719124000177

[31] J. Achiam et al., "GPT-4 technical report," 2023, *arXiv:2303.08774*.

[32] N. Hossain, M. H. Bijoy, S. Islam, and S. Shatabda, "Panini: A transformer-based grammatical error correction method for Bangla," *Neural Comput. Appl.*, vol. 36, no. 7, pp. 3463–3477, Mar. 2024, doi: 10.1007/s00521-023-09211-7.

[33] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. Int. Conf. Learn. Represent.*, 2015.

[34] D. Hendrycks and K. Gimpel, "Bridging nonlinearities and stochastic regularizers with Gaussian error linear units," 2016, *arXiv:1606.08415*.

[35] M. Ali et al., "Tokenizer choice for LLM training: Negligible or crucial?" in *Proc. Findings Assoc. Comput. Linguistics, NAACL*, Jun. 2024, pp. 3907–3924. [Online]. Available: https://aclanthology.org/2024.findings-naacl.247/

[36] T. A. Dang, L. Raviv, and L. Galke, "Tokenization and morphology in multilingual language models: A comparative analysis of mT5 and ByT5," in *Proc. 8th Int. Conf. Natural Lang. Speech Process. (ICNLSP-2025)*, Aug. 2025, pp. 242–257. [Online]. Available: https://aclanthology.org/2025.icnlsp-1.24/

[37] D. A. Haslett, "Tokenization changes meaning in large language models: Evidence from Chinese," *Comput. Linguistics*, vol. 51, no. 3, pp. 785–814, doi: 10.1162/coli_a_00557.

[38] F. Wilcoxon, "Individual comparisons by ranking methods," *Biometrics Bull.*, vol. 1, no. 6, pp. 80–83, Dec. 1945.

[39] T. B. Brown, "Language models are few-shot learners," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 33, 2020, pp. 1877–1901. [Online]. Available: https://papers.nips.cc/paper/2020/hash/1457c0d6bfcb4967418bfb8ac142f64a-Abstract.html

[40] J. Dodge, M. Sap, A. Marasović, W. Agnew, G. Ilharco, D. Groeneveld, M. Mitchell, and M. Gardner, "Documenting large webtext corpora: A case study on the colossal clean crawled corpus," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, Nov. 2021, pp. 1286–1305. [Online]. Available: https://aclanthology.org/2021.emnlp-main.98

[41] J. Wei, Y. Tay, R. Bommasani, C. Raffel, B. Zoph, S. Borgeaud, D. Yogatama, M. Bosma, D. Zhou, D. Metzler, E. H. Chi, T. Hashimoto, O. Vinyals, P. Liang, J. Dean, and W. Fedus, "Emergent abilities of large language models," *Trans. Mach. Learn. Res.*, pp. 1–30, Apr. 2022. [Online]. Available: https://openreview.net/forum?id=yzkSU5zdwD

[42] Y. Elazar, A. Bhagia, I. H. Magnusson, A. Ravichander, D. Schwenk, A. Suhr, E. P. Walsh, D. Groeneveld, L. Soldaini, S. Singh, H. Hajishirzi, N. A. Smith, and J. Dodge, "What's in my big data?" in *Proc. 12th Int. Conf. Learn. Represent.*, 2024, pp. 1–8. [Online]. Available: https://openreview.net/forum?id=RvfPnOkPV4

[43] S. Wu and M. Dredze, "Are all languages created equal in multilingual BERT?" in *Proc. 5th Workshop Represent. Learn. NLP*, Jul. 2020, pp. 120–130. [Online]. Available: https://aclanthology.org/2020.repl4nlp-1.16/

NAHID HOSSAIN is currently an Assistant Professor with the Department of Computer Science and Engineering (CSE) and the Undergraduate Program Coordinator with United International University (UIU), Dhaka, Bangladesh. Prior to his current role, he served as a Lecturer with the CSE Department. Before entering academia, he worked as a Software Engineer at the AI Division, eGeneration Ltd., Dhaka. His research interests include natural language processing (NLP) and large language models (LLMs), with a focus on language understanding and generation, language technology and resource development, and model efficiency and optimization. His research contributions span transformer-based grammatical and spelling error correction, dialect conversion systems, and text simplification for low-resource languages, alongside foundational resource development, such as corpora and evaluation benchmarks. His recent work investigates context minimization strategies to improve the resource efficiency of LLMs in constrained computational environments.

MD. FAISAL KABIR (Member, IEEE) received the Ph.D. degree in computer science from North Dakota State University, USA. During his doctoral studies, he served as an Instructor and a Teaching and Research Assistant at the Department of Computer Science, NDSU. He is currently an Assistant Professor of computer science with Pennsylvania State University Harrisburg, USA. His research interests machine learning and data science, with the goal of developing and applying intelligent computational techniques across diverse domains, including healthcare, biomedical applications, and software engineering. He is also interested in leveraging knowledge-driven and design technologies to support and empower underprivileged communities. His current research interests include data mining, deep learning, natural language processing, and health informatics.