From Bias to Balance: How Multilingual Dataset Composition Affects Tokenizer Performance Across Languages

Aishwarya Selvamurugan[1], Raj Dandekar[2], Rajat Dandekar[2], Sreedath Panat[2]

[1]Department of Computer Science,
Sri Eshwar College of Engineering,
Coimbatore, India
aishh2305@gmail.com

[2]Vizuara AI Labs,
Pune, India
raj@vizuara.com, rajatdandekar@vizuara.com, sreedath@vizuara.com

Abstract

Tokenization serves as a crucial preprocessing step in multilingual language models, affecting performance in both high-resource and low-resource languages. However, current tokenizers seem to adopt language biases due to unbalanced training datasets, leading to a poorly optimized tokenizer for underrepresented languages. This research examines the impact of balanced multilingual datasets on the performance of tokenizers trained with the Byte Pair Encoding, WordPiece, and Unigram Language Model algorithms. We build balanced corpora from various sources to study the impact of vocabulary size on 15k, 30k, 50k dataset scales. The trained tokenizers are assessed through intrinsic metrics, including Subword Fertility and Normalized Sequence Length, as well as through extrinsic performance on downstream tasks like Part-of-Speech tagging, Named Entity Recognition, and Machine Translation. We build custom data sets along with customized evaluation pipelines to enable consistent comparisons across nine languages using models built into standard NLP frameworks. Our observations reinforce the importance of a balanced dataset when training tokenizers and, in turn, advance the development of equitable and robust multilingual NLP systems.

Code —
https://github.com/Aishwarya-Selvamurugan/Tokenizer

Introduction

Tokenization is a fundamental step in natural language processing (NLP), serving as the bridge between raw text and model input. It enables diverse linguistic structures to be converted into standardized forms that can be effectively processed by deep learning models [Sennrich, Haddow, and Birch 2016]. This conversion becomes particularly critical in multilingual settings, where vocabulary overlap across languages is often limited [Conneau et al. 2020]. Subword-based tokenization strategies have emerged as a dominant solution because they alleviate the out-of-vocabulary (OOV) problem by segmenting unseen or rare words into smaller known units [Kudo and Richardson 2018]. Such an approach is especially advantageous for morphologically rich and low-resource languages, where word formation processes generate a vast number of infrequent word forms.

Different tokenization schemes optimize various trade-offs between vocabulary compression, sequence length, and downstream model efficiency. Importantly, tokenization quality has been shown to directly affect both model efficiency and fairness, with higher subword fertility in low-resource languages often resulting in longer token sequences and reduced accuracy [Lindsey et al. 2024].

Multilingual pretrained models such as mBERT [Devlin et al. 2019] and XLM-R [Conneau et al. 2020] rely on shared subword vocabularies across languages. However, this approach can result in token collisions, inconsistent granularity, and biased performance, particularly disadvantaging underrepresented languages. Recent studies have highlighted that tokenizer design and vocabulary allocation can encode systemic bias, resulting in token inflation or inadequate coverage for certain scripts [Xiang Zhang 2024].

Despite their widespread use, existing tokenization strategies disproportionately favor high-resource and Latin-script languages. This leads to over-segmentation of low-resource or non-Latin languages, thereby inflating sequence lengths and reducing efficiency [Petrov et al. 2023]. Empirical evidence shows that metrics such as subword fertility, normalized sequence length (NSL), and parity vary significantly across languages and directly impact downstream tasks [Rust et al. 2021]. Furthermore, tokenizer biases can impose up to 68% additional training costs when multilingual models are trained on skewed datasets dominated by European languages [Ali et al. 2023]. Beyond computational concerns, these disparities introduce broader societal and economic consequences. Languages that undergo excessive token fragmentation incur higher API usage costs and slower processing times, disproportionately affecting communities that already face digital marginalization [Rust et al. 2021], [Orevaoghene Ahia 2023]. Such disparities not only undermine fairness and representation but also exacerbate social and linguistic inequalities in multilingual NLP systems.

In this work, we aim to systematically investigate how balanced multilingual corpora can mitigate such disparities in tokenizer performance. Specifically, we analyze nine ty-

pologically diverse languages (Yoruba, Arabic, Mandarin Chinese, Russian, Hindi, Japanese, Swahili, Bengali, and Turkish) selected to represent different language families, writing systems, and morphological complexity levels. Our evaluation combines intrinsic metrics (e.g., subword fertility, normalized sequence length) with extrinsic performance on downstream tasks such as part-of-speech (POS) tagging, named entity recognition (NER), and machine translation. By adopting balanced datasets derived from Wikipedia and OSCAR, we seek to assess whether equitable data representation can yield fairer and more efficient tokenizers across languages.

The scope of this study is deliberately focused on subword-based tokenizers, excluding character-level and neural tokenization approaches. Additionally, while we evaluate downstream tasks using curated datasets, the experiments do not extend to full-scale pretrained large language models. Data limitations, particularly for low-resource languages such as Yoruba in NER tasks, further shape the boundaries of our investigation. Nonetheless, this work contributes toward a deeper understanding of multilingual tokenization fairness and provides empirical evidence for the importance of balanced data in tokenizer design.

The remainder of this paper is organized as follows: Section 2 reviews related work on multilingual tokenization bias and fairness. Section 3 presents our methodology, including dataset construction, tokenizer training, and evaluation frameworks across POS tagging, NER, and machine translation tasks. Section 4 reports experimental results comparing tokenizer performance using both intrinsic metrics and downstream task evaluations. Finally, Section 5 discusses our findings, acknowledges limitations, and outlines directions for future research.

Related Works

Tokenization Algorithms

The three primary subword tokenization algorithms examined in this study optimize different linguistic properties. BPE [Sennrich, Haddow, and Birch 2016] uses frequency-driven merging of character pairs, making it effective for capturing morphological patterns but potentially inconsistent across diverse languages. WordPiece [Devlin et al. 2019] extends BPE with likelihood-based merging, offering balanced performance particularly suitable for morphologically aware tasks. Unigram Language Model [Kudo 2018] employs a probabilistic framework, starting with comprehensive vocabularies and pruning based on likelihood, often yielding more linguistically motivated segmentations for morphologically rich languages.

Fairness, Dataset Composition and Evaluation metrics

Fairness in multilingual NLP has been extensively studied, with tokenization identified as a fundamental source of bias [Blodgett et al. 2020]. Conneau et al. (2020) demonstrated that balanced multilingual training data improves cross-lingual transfer performance [Conneau et al. 2020] , while Ahia et al. (2023) found substantial improvements in African languages when using balanced corpora [Ahia, Kreutzer, and Hooker 2023]. However, most studies focus on model training rather than tokenizer development itself. Tokenization quality assessment employs both intrinsic metrics like Subword Fertility [Lindsey et al. 2024] and Normalized Sequence Length, and extrinsic performance on downstream tasks. Rust et al. (2021) expanded evaluation frameworks to include parity metrics measuring cross-lingual consistency [Rust et al. 2021].

Alternative Approaches

Character-level approaches like CANINE [Clark et al. 2022] and ByT5 [Xue et al. 2021] eliminate tokenization bias by operating directly on Unicode characters, though at increased computational cost [Lindsey et al. 2024]. Recent adaptive methods such as MAGNET propose learnable segment boundaries to reduce over-segmentation [Blasi, Anastasopoulos, and Neubig 2024], representing a shift toward dynamic, context-aware tokenization. This work addresses existing gaps by systematically evaluating tokenization algorithms across nine typologically diverse languages using balanced training corpora, providing comprehensive analysis of both intrinsic metrics and downstream task performance to advance equitable multilingual tokenization.

Methodology

Tokenizers

Dataset Generation The dataset for tokenizer training was assembled as a typologically diverse corpus from two public sources: language-specific Wikipedia dumps[1] and the OSCAR Common Crawl-derived corpus [2]. Wikimedia XML dumps served as the primary encyclopaedic source, while OSCAR provided complementary broader coverage and domain diversity from web-crawl data for all nine target languages: Yoruba, Arabic, Mandarin Chinese, Russian, Hindi, Japanese, Swahili, Bengali, and Turkish. We targeted equal per-language character allocations for all target languages to reduce high-resource bias during tokenizer training. A deliberately balanced sampling strategy was employed, as prior work shows that tokenizer training data composition and per language representation materially affect downstream performance [Zhang et al. 2022]. To investigate the impact of training corpus size on vocabulary learning and downstream tasks, we created three balanced datasets containing 100 million, 200 million, and 400 million characters.

Data Preprocessing A uniform normalization procedure was applied across all corpora to ensure consistency between languages and sources. The preprocessing pipeline consisted of four sequential steps. First, text repair was performed using the `ftfy` library's `fix_text()` function to correct smart quotes, inconsistent punctuation, and other common encoding anomalies. Second, Unicode normalization was applied via the Python `unicodedata` module, converting all characters to NFKC (Normalization Form Compatibility

[1] https://dumps.wikimedia.org/

[2] https://oscar-project.org/

Composition) form to achieve canonical equivalence, compatibility decomposition, and recomposition. Third, non-printable Unicode control characters (e.g., \x00--\x1F, excluding standard whitespace) were removed to eliminate formatting artifacts. Finally, whitespace normalization was conducted by collapsing consecutive spaces, tabs, and newline characters into single spaces and trimming leading or trailing whitespace using regular expressions. Only non-empty, normalized lines were retained, resulting in a clean and consistent dataset for tokenizer training.

Tokenizer training We set up the processing pipelines using Hugging Face tokenizers and transferred the preprocessed data. Three tokenization algorithms were employed:

- Byte Pair Encoding (BPE)
- WordPiece
- Unigram

A total of nine tokenizer models were trained on the normalized, balanced corpus for nine languages, with three vocabulary sizes: 15k, 30k, and 50k. For BPE and WordPiece, a whitespace-based pre-tokenization strategy was applied, consistent with their standard implementations, to preserve word boundaries. In contrast, the Unigram model operated directly on raw text by design, ensuring full character coverage and better handling of scripts without reliable whitespace delimiters. The detailed configurations for each tokenizer are shown in Table 1 and Table 2, providing a concise overview of the implementation, vocabulary sizes, and preprocessing strategies used.

This design allows for a direct comparison of the three algorithms under different vocabulary sizes, ensuring that downstream evaluation results can be attributed to model differences rather than data or preprocessing inconsistencies.

Evaluation metrics To access the quality of the trained tokenizer beyond vocabulary coverage, we evaluated them using 2 metrics: Normalized Sequence Length (NSL) and Subword Fertiltiy, to compare the tokenization behaviour across languages and vocabulary sizes.

Normalized Sequence Length Normalized Sequence Length (NSL) measures the average number of tokens per character in a sequence:

$$\text{NSL} = \frac{\text{Number of tokens}}{\text{Number of characters}} \tag{1}$$

This metric shows the granularity of segmentation, higher NSL values suggest more fine-grained tokenization.

Subword Fertility Subword Fertility measures the average number of subword tokens produced per whitespace-delimited word:

$$\text{Subword Fertility} = \frac{\text{Number of tokens}}{\text{Number of words}} \tag{2}$$

A higher subword fertility value indicates that words are split into more subword units, which can be beneficial for handling rare words but may lead to longer sequences.

For evaluation, we curated language balanced dataset from publicly available TATOEBA corpus [3] for Yoruba and Bengali, and the TED2020 corpus [4] for Arabic, Mandarin Chinese, Russian, Hindi, Japanese, Swahili, and Turkish. For each language, we randomly sampled 50 sentences from the respective sources. For every sentence, we computed NSL and Subword Fertlity, then averaged these scores per tokenizer. The results are given in Table 3.

Downstream tasks

Parts of speech (POS) Tagging

Data collection For the POS tagging task, we assembled a diverse language balanced dataset for the same 9 languages from publicly available sources. Specifically, Arabic, Mandarin Chinese, Russian, Hindi, Japanese, and Turkish data were obtained from the Universal Dependencies treebanks, whereas Yoruba and Swahili were taken from the Masakha-POS corpus and Bengali was sourced from the NLTK Indian corpus.

Preprocessing The preprocessing phase focused on ensuring uniformity in format and quality across all nine languages. For each dataset, only sentences in which the number of tokens exactly matched the number of POS tags were retained. All corpora were then standardized into a unified JSONL structure with three fields: tokens (list of words), tags (corresponding POS labels), and lang (ISO language code). Language codes were explicitly added to each sentence to maintain traceability. Following the balancing process, the combined dataset was randomly shuffled to mitigate any potential ordering effects during training. The final multilingual POS dataset was saved in UTF-8 encoding without altering the original text content, preserving the integrity of the gold-standard annotations.

Model Training For POS tagging, we fine-tuned a BERT-based token classification model on the balanced multilingual dataset using the 9 different tokenizers of varying sizes and segmentation strategies which is trained. Each tokenizer which was trained separately are loaded into the Hugging Face AutoTokenizer to ensure consistent tokenization between training and evaluation. The POS model was implemented using BertForTokenClassification with the base architecture bert-base-cased, adapting the embedding layer to match each tokenizer's vocabulary size. Input sequences were tokenized in a word-aligned manner, with subword tokens inheriting the POS label of their originating word and non-aligned tokens masked with the -100 label to exclude them from loss computation. Training was performed with a batch size of 16, a learning rate of 5e-5, and a fixed three-epoch schedule.

NER

Data collection We constructed a balanced multilingual NER dataset covering all the nine languages. For Yoruba,

[3] https://tatoeba.org/

[4] https://opus.nlpl.eu/TED2020.php

Table 1: Tokenizer configurations: Model properties.

Tokenizer Model	Library	Algorithm Type	Vocab Sizes Used
HF-BPE	HF Tokenizers	Byte-Pair Encoding	15k, 30k, 50k
HF-WordPiece	HF Tokenizers	WordPiece	15k, 30k, 50k
SP-Unigram	SentencePiece	Unigram LM	15k, 30k, 50k

Table 2: Tokenizer configurations: Pre-tokenization and coverage.

Tokenizer Model	Pre-tokenization	Special Tokens	Character Coverage
HF-BPE	Whitespace	[PAD], [UNK], [CLS], [SEP], [MASK]	100%
HF-WordPiece	Whitespace	[PAD], [UNK], [CLS], [SEP], [MASK]	100%
SP-Unigram	None (raw text)	<pad>, <unk>	100%

we used the MasakhaneNER dataset in CoNLL format, containing manually annotated tokens and entity labels. For the other eight languages, we used the WikiANN multilingual NER corpus via the Hugging Face Datasets library.

Preprocessing The Yoruba dataset was first parsed from CoNLL format into a token-tag JSON structure, handling any malformed lines by assigning an "O" (non-entity) tag. To unify label formats across datasets, we extracted the label set from WikiANN and mapped the Yoruba entity tags to this schema, discarding any samples containing unsupported tags. All datasets were standardized to a common feature schema consisting of `tokens`, `ner_tags`, and `language` fields, with `ner_tags` stored as integer indices corresponding to the unified label list. We balanced the dataset by downsampling each language to an equal number of samples (double the smallest split size between WikiANN datasets), shuffled the combined set, and split it into 80% training and 20% test subsets.

Model Training For the Named Entity Recognition (NER) experiments, a BERT-based token classification architecture was employed, trained using all 9 tokenizer configurations. The datasets were tokenized in a manner that preserved token–tag alignment, with subword units inheriting their corresponding word level labels. To address class imbalance in the entity label distribution, class weights were computed from the training corpus and incorporated into the loss function, improving recognition of less frequent entity classes. Each configuration was finetuned under the same hyperparameter settings for a fixed number of epochs, ensuring comparability across runs. Performance was evaluated using entity-level precision, recall, F1-score, and accuracy, following the seqeval metric standard for sequence labeling.

Machine Translation

Data Collection For the Machine Translation experiments, parallel corpora were sourced from two large-scale open datasets: OPUS100 and TED2020. OPUS100 provides high-quality sentence-aligned translations across 100 languages, while TED2020 consists of transcribed and translated TED Talk segments in multiple languages. For languages with limited high-quality coverage in OPUS100, such as Swahili, TED2020 served as the primary source. All datasets were obtained from official repositories, ensuring consistent formatting and alignment between English and the respective target languages.

Preprocessing We applied a systematic preprocessing pipeline to ensure corpus integrity and comparability. Quality filtering removed sentence pairs with identical source/target text, segments under three English tokens or two target tokens, sentences exceeding 200 characters, and text dominated by numbers or punctuation. Technical strings (HTML/XML tags, encoding markers) were excluded to eliminate non-linguistic noise. To mitigate high-resource language bias, we downsampled all datasets to match the smallest corpus size post-filtering, ensuring uniform per-language representation. The dataset was randomized with a fixed seed and split 90-10 for training/testing. Records were standardized with English sentences, translations, and language codes, producing a clean, balanced multilingual dataset for rigorous cross-linguistic evaluation.

Model Training We conducted machine translation experiments using BART-large with custom tokenizers across different vocabulary sizes and algorithms. Models were trained using Seq2SeqTrainer for 5 epochs with batch size 8, gradient accumulation steps 8 (effective batch size 64), learning rate 1e-4, 10% warmup, weight decay 0.01, and FP16 precision. We trained a single multilingual model on nine languages simultaneously, with inputs truncated/padded to 256 tokens and language-specific formatting. Training stability was ensured through label smoothing (0.1) and gradient clipping (max norm 1.0). Generation used beam search (2 beams, max length 128). Models were evaluated every 500 steps using BLEU scores and exact match accuracy on the complete balanced multilingual dataset, enabling direct comparison across tokenization strategies.

Experimental Results

Intrinsic Tokenizer Evaluation

The intrinsic evaluation compares BPE, WordPiece, and SentencePiece-Unigram tokenizers for the vocabulary sizes 15k, 30k, and 50k using Normalized Sequence Length (NSL) and Subword Fertility as evaluation metrics [Rust et al. 2021, Ács, Grad-Gyenge, and Vadász 2023]. NSL

captures the relative tokenization length with respect to the original text, while Subword Fertility quantifies the average number of subword units generated per original token [Mielke et al. 2021]. Lower values for both metrics generally indicate more compact and efficient tokenization [Kudo and Richardson 2018, Salesky et al. 2020].

As shown in Table 3, increasing vocabulary size consistently reduces both metrics, yielding more compact token sequences, with the effect most pronounced in Mandarin Chinese and Japanese due to their logographic writing systems [Kudo and Richardson 2018, Conneau et al. 2020]. Among tokenizers, BPE achieves the lowest NSL and fertility across most languages, while WordPiece generally produces the highest values, particularly at smaller vocabularies, and SentencePiece Unigram performs in between [Sennrich, Haddow, and Birch 2016, Wu et al. 2016, Kudo and Richardson 2018]. Language typology strongly influences outcomes, as morphologically complex languages show larger efficiency gains with larger vocabularies, whereas morphologically simpler languages like Yoruba and Swahili remain relatively stable [Bostrom and Durrett 2020, Wang, Cho, and Gu 2020]. Overall, the results highlight that larger vocabularies provide more efficient tokenization across all algorithms, reducing sequence length and subword fragmentation, which can lower computational overhead in downstream tasks [Qiu et al. 2020].

POS Tagging Results

We further compare tokenizers on POS tagging across vocabulary sizes using accuracy and F1 metrics. From Table 4, it can be inferred that wordPiece achieves the highest overall performance, with a test accuracy of 0.7830 and weighted F1 of 0.7722 at 15k, consistently outperforming BPE and SentencePiece Unigram. This can be attributed to WordPiece's ability to preserve morphologically meaningful units, which benefits POS tagging where syntactic boundaries are crucial [Straka, Hajic, and Strakova 2016]. In contrast, BPE prioritizes frequency-based merges, often splitting or merging across morpheme boundaries, which reduces efficiency for this task despite shorter sequences [Sennrich, Haddow, and Birch 2016, Kudo 2018]. SentencePiece Unigram shows intermediate behavior, offering slightly higher macro-F1 than BPE but lacking the stability of WordPiece [Kudo and Richardson 2018]. Notably, increasing vocabulary size does not improve results and in some cases reduces accuracy, as larger vocabularies can overspecialize subword units and lose the generalization capacity needed for POS tagging [Nivre et al. 2016, Kann and Schütze 2016, Mielke et al. 2021].

NER Results

Table 5 presents the NER performance across tokenizers, vocabulary sizes, and evaluation metrics. With 15k vocabulary size, WordPiece significantly outperforms BPE and SentencePiece Unigram, achieving the highest Test F1 (0.5844) and Test Accuracy (0.7644). This suggests that WordPiece is particularly effective in low-vocabulary regimes, where its ability to balance word-level and subword-level information aids entity boundary recognition [Devlin et al. 2019, Wu et al. 2016, Li et al. 2019]. As vocabulary size increases to 30k and 50k, BPE shows competitive performance, especially in Test Accuracy (0.7032 at 50k), while SentencePiece occasionally surpasses BPE in terms of F1 score. However, WordPiece maintains overall superiority, albeit with diminishing margins, likely due to reduced fragmentation and more stable subword segmentation as vocabulary size grows [Klein et al. 2017].

Overall, these results indicate that WordPiece offers the best generalization for NER at smaller vocabulary sizes, while BPE and SentencePiece Unigram become more competitive at larger vocabularies, reflecting the trade-off between segmentation granularity and contextual representation in sequence labeling tasks [Peters et al. 2018, Akbik, Blythe, and Vollgraf 2018, Mielke et al. 2021].

Machine Translation Results

Table 6 presents the performance of different tokenization methods on multilingual BART-large across vocabulary sizes of 15k, 30k, and 50k. BPE with 15k vocabulary achieved the best BLEU score (0.1226) due to its efficient segmentation of rare words into frequent subword units, balancing vocabulary compactness with representational power [Sennrich, Haddow, and Birch 2016]. WordPiece performed poorly (BLEU = 0.0103) because its tendency to favor longer subwords led to insufficient coverage of morphologically rich words in multilingual data [Wu et al. 2016]. SentencePiece Unigram produced shorter prediction lengths, suggesting under-segmentation effects. At 30k vocabulary, WordPiece achieved BLEU of 0.1136, nearly matching BPE (0.1135). SentencePiece Unigram showed the highest Exact Match (0.086) despite lower BLEU, indicating that its probabilistic subword sampling captured more precise token boundaries, though its aggressive segmentation may reduce fluency in longer sequences [Mielke et al. 2021]. For 50k vocabulary, SentencePiece Unigram achieved the highest Exact Match (0.096) while WordPiece obtained the strongest BLEU score (0.1218). This suggests WordPiece benefits from larger vocabularies by covering more lexical items directly, improving overall fluency [Wu et al. 2016, Devlin et al. 2019], whereas Unigram maintains precision in token alignment but at the cost of sequence length imbalance [Kudo and Richardson 2018]. BPE underperformed as vocabulary grew, likely due to oversplitting that reduced sequence-to-sequence alignment efficiency. These results suggest BPE is effective for smaller vocabularies, WordPiece scales better with larger vocabularies, and SentencePiece Unigram excels in exact matching but generates shorter sequences. The performance differences stem from how each tokenizer balances subword granularity, vocabulary coverage, and sequence length, which directly impact BLEU and Exact Match metrics [Post 2018].

Discussion and Conclusion

This study systematically investigated the impact of balanced multilingual datasets on tokenizer performance across nine typologically diverse languages, revealing significant insights into how dataset composition affects both tokenization efficiency and downstream task performance. Our

Table 3: NSL and Subword Fertility across tokenizers, languages, and vocabulary sizes.

Language	Tokenizer	15k voc size		30k voc size		50k voc size	
		NSL	Subword Fertility	NSL	Subword Fertility	NSL	Subword Fertility
Yoruba	BPE	0.4584	1.9911	0.3993	1.7278	0.3668	1.5783
	WordPiece	0.8137	3.5557	0.4584	1.9867	0.3827	1.6498
	SentencePiece Unigram	0.5471	2.3723	0.4178	1.8040	0.3963	1.7086
Arabic	BPE	0.4908	2.8232	0.3726	2.1427	0.3316	1.9056
	WordPiece	0.8353	4.8084	0.4485	2.5773	0.3567	2.0482
	SentencePiece Unigram	0.4859	2.7878	0.3879	2.2264	0.3427	1.9706
Mandarin Chinese	BPE	0.8479	8.7353	0.7438	7.6769	0.6985	7.2173
	WordPiece	0.8391	8.4954	0.7628	7.7758	0.6786	6.9250
	SentencePiece Unigram	0.9393	9.6524	0.8556	8.7974	0.8166	8.4003
Russian	BPE	0.4621	3.3589	0.3367	2.4322	0.2990	2.1634
	WordPiece	0.8701	6.3412	0.4406	3.1862	0.3276	2.3668
	SentencePiece Unigram	0.4879	3.5411	0.3414	2.4549	0.3091	2.2264
Hindi	BPE	0.4727	2.5893	0.3566	1.9619	0.3258	1.8028
	WordPiece	0.7793	4.1668	0.4210	2.2823	0.3377	1.8444
	SentencePiece Unigram	0.5174	2.8608	0.3819	2.1243	0.3497	1.9463
Japanese	BPE	0.7739	11.0098	0.5978	8.4776	0.5290	7.4856
	WordPiece	0.9168	13.1019	0.7082	10.0753	0.5775	8.1898
	SentencePiece Unigram	0.8368	11.8092	0.6681	9.3931	0.5973	8.3685
Swahili	BPE	0.4008	2.6293	0.2953	1.9350	0.2620	1.7163
	WordPiece	0.8573	5.6392	0.3759	2.4664	0.2910	1.9076
	SentencePiece Unigram	0.4068	2.6739	0.3031	1.9849	0.2632	1.7273
Bengali	BPE	0.4270	2.9301	0.2897	1.9807	0.2479	1.6934
	WordPiece	0.8569	5.8917	0.3897	2.6693	0.2759	1.8865
	SentencePiece Unigram	0.4621	3.1692	0.2958	2.0243	0.2512	1.7167
Turkish	BPE	0.4295	3.3339	0.3166	2.4487	0.2807	2.1667
	WordPiece	0.8805	6.8531	0.3937	3.0524	0.3050	2.3543
	SentencePiece Unigram	0.4482	3.4741	0.3262	2.5227	0.2826	2.1694

Table 4: POS Performance comparison across tokenizers, vocabulary sizes, and metrics. Best values per block are highlighted in bold.

Tokenizer	Voc size	Epoch	Train Acc	Train F1 Macro	Train F1 Weighted	Test Acc	Test F1 Macro	Test F1 Weighted
BPE		3	0.7847	0.2957	0.7722	0.7207	0.2973	0.7057
WordPiece	15k	3	**0.8413**	**0.3889**	**0.8325**	**0.7830**	**0.3952**	**0.7722**
SentencePiece Unigram		3	0.8067	0.3209	0.7947	0.7484	0.3299	0.7340
BPE		3	0.7526	0.2686	0.7366	0.6932	0.2724	0.6735
WordPiece	30k	3	**0.7964**	0.2877	**0.7839**	**0.7325**	0.2915	**0.7174**
SentencePiece Unigram		3	0.7752	**0.3141**	0.7625	0.7211	**0.3218**	0.7061
BPE		3	0.7643	0.2719	0.7484	0.7015	0.2775	0.6812
WordPiece	50k	3	**0.8001**	0.2848	**0.7880**	**0.7335**	0.2883	**0.7187**
SentencePiece Unigram		3	0.7662	**0.3085**	0.7528	0.7107	**0.3141**	0.6944

Table 5: NER Performance comparison across tokenizers, vocabulary sizes, and metrics

Tokenizer	Voc size	Epoch	Train Precision	Train Recall	Train F1	Train Accuracy	Test Precision	Test Recall	Test F1	Test Accuracy
BPE	15k	3	0.3113	0.6453	0.4200	0.6878	0.2452	0.5088	0.3309	0.6324
WordPiece		3	**0.5990**	**0.8601**	**0.7062**	**0.8374**	**0.4937**	**0.7157**	**0.5844**	**0.7644**
SentencePiece Unigram		3	0.4949	0.8398	0.6228	0.7856	0.3769	0.6441	0.4756	0.7102
BPE	30k	3	0.3730	0.7452	0.4971	0.7604	0.2640	0.5393	0.3545	0.6828
WordPiece		3	**0.4432**	**0.8106**	**0.5731**	**0.7898**	**0.3251**	**0.6063**	**0.4233**	**0.7014**
SentencePiece Unigram		3	0.4120	0.7810	0.5394	0.7449	0.2920	0.5690	0.3860	0.6532
BPE	50k	3	0.3951	0.7533	0.5183	0.7902	0.2650	0.5290	0.3531	0.7032
WordPiece		3	**0.4266**	0.7842	0.5525	**0.7927**	**0.3034**	**0.5767**	**0.3976**	**0.7153**
SentencePiece Unigram		3	0.4258	**0.7885**	**0.5530**	0.7599	0.2914	0.5515	0.3813	0.6694

Table 6: Performance comparison of different tokenization methods on multilingual BART-large. Best scores per vocabulary size are in **bold**.

Voc Size	Tokenizer Type	BLEU	Exact Match	Avg. Pred Len	Avg. Label Len
15k	BPE	**0.1226**	**0.069**	27.41	23.05
	WordPiece	0.0103	0	20	35.11
	SentencePiece Unigram	0.0526	0.031	9.14	7.37
30k	BPE	0.1135	0.006	17.99	18.55
	WordPiece	**0.1136**	0.067	23.30	21.65
	SentencePiece Unigram	0.0966	**0.086**	7.58	7.37
50k	BPE	0.1039	0.055	22.35	16.97
	WordPiece	**0.1218**	0.078	19.94	17.96
	SentencePiece Unigram	0.1039	**0.096**	7.13	7.37

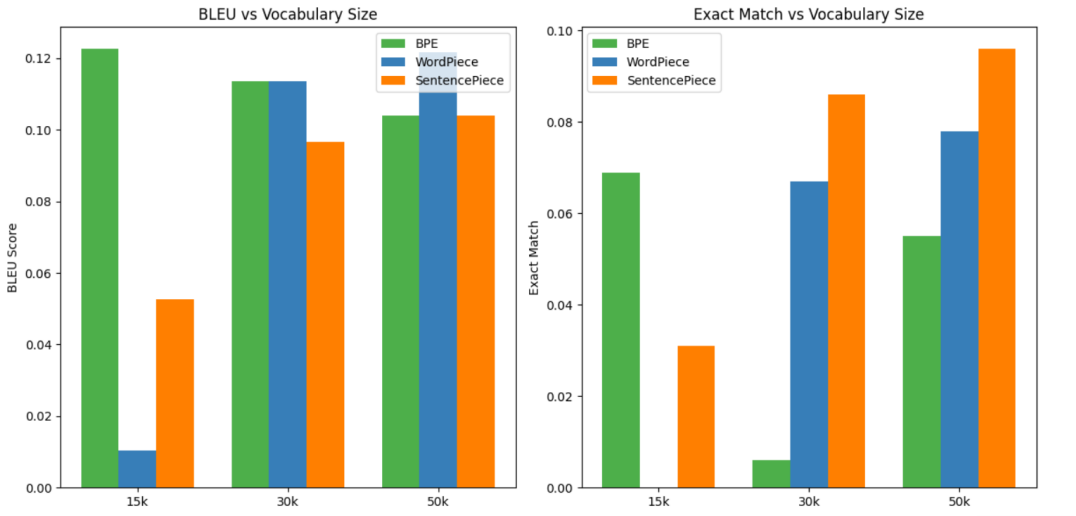

Figure 1: BLEU and Exact Match performance of tokenization methods across vocabulary sizes on multilingual machine translation, showing vocabulary-dependent performance patterns for each algorithm.

findings demonstrate that balanced training data substantially improves tokenization fairness, with BPE consistently achieving the lowest Normalized Sequence Length and Subword Fertility values across most languages, indicating more compact tokenization through its frequency-based merging strategy. Notably, logographic languages like Mandarin Chinese and Japanese showed dramatic improvements with larger vocabularies, with NSL values decreasing from 0.85 to 0.70 for Chinese when moving from 15k to 50k vocabulary sizes. In downstream tasks, tokenizer choice proved highly task-dependent: WordPiece excelled in POS tagging (accuracy: 0.7830) and NER (F1: 0.5844) due to its ability to preserve morphologically meaningful boundaries, while BPE performed best in machine translation at smaller vocabularies (BLEU: 0.1226 at 15k). These results provide empirical evidence that balanced datasets enable tokenizers to learn more representative subword units across diverse writing systems, reducing the over-segmentation typically observed in low-resource languages and addressing computational inequities where underrepresented languages can incur up to 68% additional processing costs.

While our results are promising, several limitations must be acknowledged. Future research should address these limitations by expanding evaluation to character-level and neural tokenization methods and exploring intermediate balancing strategies tailored to specific language families. We plan to extend our framework to generative tasks where tokenizer choice may have different implications for output quality and fairness, and develop dynamic balancing strategies that adapt to evolving multilingual corpora. Multi-tokenizer approaches within single models represent a promising avenue for leveraging different tokenization strategies simultaneously across languages or tasks.

This study establishes the critical importance of balanced training data in achieving fair and efficient multilingual tokenization. While optimal tokenizer choice remains task-dependent, balanced datasets consistently improve performance across all evaluated conditions. By bridging tokenizer efficiency and linguistic fairness, this work contributes to inclusive language technologies that serve diverse global communities with computational equity, moving beyond one-size-fits-all paradigms toward more equitable multilingual NLP systems.

References

Ahia, O.; Kreutzer, J.; and Hooker, S. 2023. The low-resource double bind: An empirical study of pruning for low-resource machine translation. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, 1–15.

Akbik, A.; Blythe, D.; and Vollgraf, R. 2018. Contextual string embeddings for sequence labeling. In *Proceedings of the 27th International Conference on Computational Linguistics*, 1638–1649.

Ali, M.; Fromm, M.; Thellmann, K.; Rutmann, R.; Lübbering, M.; Leveling, J.; Klug, K.; Ebert, J.; Doll, N.; Buschhoff, J. S.; et al. 2023. Tokenizer choice for LLM training: negligible or crucial? arXiv:2310.08754.

Blasi, D.; Anastasopoulos, A.; and Neubig, G. 2024. MAGNET: Improving the multilingual fairness of language models with adaptive gradient-based tokenization. In *Advances in Neural Information Processing Systems*, volume 37.

Blodgett, S. L.; Barocas, S.; Daumé III, H.; and Wallach, H. 2020. Language (technology) is power: A critical survey of "bias" in NLP. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 5454–5476.

Bostrom, K.; and Durrett, G. 2020. Byte pair encoding is suboptimal for language model pretraining. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, 4617–4624.

Clark, J. H.; Garrette, D.; Turc, I.; and Wieting, J. 2022. Canine: Pre-training an efficient Tokenization-Free encoder for language representation. *Transactions of the Association for Computational Linguistics*, 10: 73–91.

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

Devlin, J.; Chang, M.-W.; Lee, K.; and Toutanova, K. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*.

Kann, K.; and Schütze, H. 2016. Single-model encoder-decoder with explicit morphological representation for reinflection. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, 555–560.

Klein, G.; Kim, Y.; Deng, Y.; Senellart, J.; and Rush, A. M. 2017. OpenNMT: Open-source toolkit for neural machine translation. In *Proceedings of ACL 2017, System Demonstrations*, 67–72.

Kudo, T. 2018. Subword Regularization: Improving Neural Network Translation Models with Multiple Subword Candidates. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*.

Kudo, T.; and Richardson, J. 2018. SentencePiece: a simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 66–71.

Li, X.; Feng, J.; Meng, Y.; Han, Q.; Wu, F.; and Li, J. 2019. A unified MRC framework for named entity recognition. *arXiv preprint arXiv:1910.11476*.

Lindsey, L. M.; Pershing, N. L.; Habib, A.; Stephens, W. Z.; Blaschke, A. J.; Jiang, X.; Sundar, H.; and Dufault-Thompson, K. 2024. A comparison of tokenization impact in attention based and state space genomic language models. *bioRxiv*.

Mielke, S. J.; Alyafeai, Z.; Salesky, E.; Raffel, C.; Dey, M.; Gallé, M.; Raja, A.; Si, C.; Lee, W. Y.; Sagot, B.; et al.

2021. Between words and characters: A brief history of open-vocabulary modeling and tokenization in NLP. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics*, 5647–5664.

Nivre, J.; de Marneffe, M.-C.; Ginter, F.; Goldberg, Y.; Hajic, J.; Manning, C. D.; McDonald, R.; Petrov, S.; Pyysalo, S.; Silveira, N.; et al. 2016. Universal Dependencies v1: A multilingual treebank collection. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation*, 1659–1666.

Orevaoghene Ahia, H. G. J. K. D. R. M. N. A. S. Y. T., Sachin Kumar. 2023. Do all languages cost the same? Tokenization in the era of commercial language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, 9904–9923.

Peters, M. E.; Neumann, M.; Iyyer, M.; Gardner, M.; Clark, C.; Lee, K.; and Zettlemoyer, L. 2018. Deep contextualized word representations. In *Proceedings of NAACL-HLT*, 2227–2237.

Petrov, A.; Emanuele, L. M.; Torr, P. H. S.; and Bibi, A. 2023. Language model tokenizers introduce unfairness between languages. arXiv:2306.15141.

Post, M. 2018. A call for clarity in reporting BLEU scores. In *Proceedings of the Third Conference on Machine Translation*, 186–191.

Qiu, X.; Sun, T.; Xu, Y.; Shao, Y.; Dai, N.; and Huang, X. 2020. Pre-trained models for natural language processing: A survey. *Science China Technological Sciences*, 63(10): 1872–1897.

Rust, P.; Pfeiffer, J.; Vulić, I.; Ruder, S.; and Gurevych, I. 2021. How good is your tokenizer? on the monolingual performance of multilingual language models. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing*, 3118–3135.

Salesky, E.; Vania, C.; Mielke, S. J.; Koo, T.; and Neubig, G. 2020. A linguistically motivated approach to multilingual tokenizer learning. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, 3463–3478.

Sennrich, R.; Haddow, B.; and Birch, A. 2016. Neural Machine Translation of Rare Words with Subword Units. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*.

Straka, M.; Hajic, J.; and Strakova, J. 2016. UDPipe: trainable pipeline for processing CoNLL-U files performing tokenization, morphological analysis, POS tagging and parsing. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation*, 4290–4297.

Wang, C.; Cho, K.; and Gu, J. 2020. Neural machine translation with byte-level subwords. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, 9154–9160.

Wu, Y.; Schuster, M.; Chen, Z.; Le, Q. V.; Norouzi, M.; Macherey, W.; Krikun, M.; Cao, Y.; Gao, Q.; Macherey, K.; et al. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*.

Xiang Zhang, C. Y., Juntai Cao. 2024. Counting ability of large language models and impact of tokenization. ArXiv:2410.19730v2.

Xue, L.; Barua, A.; Constant, N.; Al-Rfou, R.; Narang, S.; Kale, M.; Roberts, A.; and Raffel, C. 2021. ByT5: Towards a token-free future with pre-trained byte-to-byte models. arXiv:2105.13626.

Zhang, S.; Chaudhary, V.; Goyal, N.; Cross, J.; Wenzek, G.; Bansal, M.; and Guzmán, F. 2022. How Robust is Neural Machine Translation to Language Imbalance in Multilingual Tokenizer Training? In *Proceedings of the 15th biennial conference of the Association for Machine Translation in the Americas*.

Ács, J.; Grad-Gyenge, L.; and Vadász, P. 2023. Subword pooling makes a difference. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, 2284–2295.

Computational Resources

All experiments in this study were conducted using two distinct computational environments. Tokenizer training for all nine languages across three vocabulary sizes (15k, 30k, 50k) and three algorithms (BPE, WordPiece, Unigram), along with Part-of-Speech tagging and Named Entity Recognition model training, were performed using Google Colab with Tesla T4 GPU (16GB VRAM) and Python 3.10 with CUDA 11.8. Machine Translation experiments using multilingual BART-large required enhanced computational capacity and were conducted on a local workstation equipped with NVIDIA GeForce RTX 4090 (24GB VRAM) and CUDA 11.8. The choice of computational environments was determined by the memory requirements of each task, with the larger BART-large models for machine translation necessitating the higher VRAM capacity of the RTX 4090 GPU.