Parity-Aware Byte-Pair Encoding: Improving Cross-lingual Fairness in Tokenization

Negar Foroutan[1]* **Clara Meister**[1]* **Debjit Paul**[1]
Joel Niklaus[2] **Sina Ahmadi**[3] **Antoine Bosselut**[1]† **Rico Sennrich**[3]†

[1]EPFL [2]Niklaus.ai [3]University of Zurich

Abstract

Tokenization is the first—and often least scrutinized—step of most NLP pipelines. Standard algorithms for learning tokenizers rely on frequency-based objectives, which favor languages dominant in the training data and consequently leave lower-resource languages with tokenizations that are disproportionately longer, morphologically implausible, or even riddled with <UNK> placeholders. This phenomenon ultimately amplifies computational and financial inequalities between users from different language backgrounds. To remedy this, we introduce Parity-aware Byte Pair Encoding (BPE), a variant of the widely-used BPE algorithm. At every merge step, Parity-aware BPE applies a fair-max rule that maximizes the compression gain of the currently worst-compressed language, trading a small amount of global compression for cross-lingual parity. We find empirically that Parity-aware BPE reduces tokenization inequality—operationalized by the Gini coefficient of per-language token costs—by up to 89% relative to Classical BPE. This comes with negligible impact on global compression rate and no evidence of systematic degradation in downstream LM performance.[1]

1 Introduction

At a time of rapid innovation and constant change in natural language processing (NLP), tokenization remains a foundational and comparatively stable component of NLP pipelines. Tokenization is the process of transforming raw sequences of bytes[2] into sequences of byte-spans, *i.e.*, subwords; it enables computational efficiency and provides essential inductive biases by defining meaningful textual units. This design choice can have a major impact on various aspects of model performance (Bostrom and Durrett, 2020; Ali et al., 2024a; Goldman et al., 2024).

The predominant tokenization algorithms—such as Byte Pair Encoding (BPE; Sennrich et al., 2016) and UnigramLM (Kudo, 2018)—construct the vocabulary by maximizing frequency-based objectives computed over an entire training corpus. In multilingual corpora, this global criterion inevitably favors the languages with the greatest representation. Under vocabulary size constraints, subwords that primarily benefit high-resource languages are preferentially included, often at the expense of those needed for lower-resource languages. This bias has both qualitative and economic consequences. Models trained on fragmented or semantically incoherent tokenizations lose valuable inductive biases and tend to perform worse. Simultaneously, texts in lower-resource languages—often tokenized into more tokens—incur higher computational costs from LLM-based services charging based on token count. This "tokenization tax" disproportionately burdens users of underrepresented languages and exacerbates existing inequalities.

In an effort to mitigate these inequities, we introduce **Parity-aware BPE**. The classic BPE algorithm constructs its vocabulary by iteratively selecting the subword pair with the highest corpus-level frequency; it adds the concatenated pair to the vocabulary and replaces all instances of the pair with the new symbol.[3] Parity-aware BPE is a simple variant of this algorithm, retaining the iterative framework but redefining the merge selection rule: at each step, it computes co-occurrence statistics separately for each language and then uses statistics from the language with the current worst compression rate for selecting the

*Equal contribution, †Equal supervision.

[1]Code for reproducing experiments is available here; Parity-aware BPE is also available in HuggingFace.

[2]Early work considered characters the "base unit" of strings, but raw bytes have become popular because their fixed 256-symbol vocabulary can encode any character from any encoding, eliminating out-of-vocabulary issues.

[3]This process acts as a form of data compression, replacing frequent subword sequences with shorter representations.

Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 7514–7538

next merge. In other words, instead of greedily maximizing a global objective, Parity-aware BPE performs a "fair-max" update that progressively equalizes string compression rates across languages. Notably, this modification affects only the vocabulary learning phase; the inference procedure remains the same as in Classical BPE.

Empirically, Parity-aware BPE substantially improves cross-lingual tokenization fairness. In our setup that mimics standard LM training settings, we observe an 89% reduction in the Gini tokenizer inequality coefficient compared to Classical BPE. Across 12 multilingual benchmarks, models trained with a Parity-aware BPE tokenizer match downstream performance compared to those trained with a Classical BPE tokenizer. In summary, Parity-aware BPE reduces tokenizer disparities between languages while showing no systematic degradation in downstream performance relative to Classical BPE.

2 Text Tokenization

Text can be decomposed at many granularities: graphemes, Unicode code points, or multi-character tokens. At the lowest level of abstraction, however, the digital representation of a string is simply a sequence of **bytes**—the foundation on which all other units are constructed. Let $b \in \mathcal{B} = \{0, \ldots, 255\}$ denote an individual byte. A finite **byte-string** is defined as $\mathbf{b} \in \mathcal{B}^*$, where $\mathbf{b} = b_1 b_2 \cdots b_{|\mathbf{b}|}$. While we treat bytes as the base unit throughout this paper, the following definitions remain compatible with alternative base units, such as characters or graphemes.

2.1 Byte-level Tokenizers

In plain terms, **tokenization** is the process of mapping raw byte-strings to sequences of subwords. A **tokenizer** specifies the rules that perform this mapping. Tokenizers consist of two components: a **vocabulary** $\mathcal{V} \subset \mathcal{B}^+$—a finite set of non-empty byte-spans, often called *subwords*[4]—and a **tokenization function** $\tau : \mathcal{B}^* \to \mathcal{V}^*$—a mapping from byte-strings $\mathbf{b}$ to sequences of tokens $\mathbf{v} = v_1, v_2, \ldots$.

Pre-tokenization and Normalization. Many tokenization algorithms include a *pre-tokenization* (and often a normalization) step that segments or rewrites raw byte strings according to deterministic criteria. Pre-tokenization can encompass several operations, including Unicode normalization or whitespace splitting. Notably, pre-tokenization determines subword boundaries, and thus also determines the set of possible candidates for the vocabulary as well as the attainable compression rate. For example, if whitespace is used as a subword boundary, languages without explicit whitespace (*e.g.*, Chinese, Japanese) or with rich morphology may have the potential for higher compression. While often overlooked, pre-tokenization choices have a large impact on a tokenizer's learned vocabulary; we point the interested reader to (Arnett et al., 2025a). For simplicity, we assume the pre-tokenization step is baked into τ.

2.2 Text Compression

Why does mapping a raw byte-string to a sequence of larger subword tokens help an LM? While byte- or character-level sequences can be provided directly to an LM, subword tokenization remains the dominant approach for turning text into model inputs. The precise inductive biases this process imbues remain an open research question (Zouhar et al., 2023a; Schmidt et al., 2024), but one plausible explanation is the *compression* it provides: a good tokenizer tends to map each input $\mathbf{b}$ to a shorter sequence of tokens, reducing the length of the model's effective input and, potentially making learning easier. At the very least, it can significantly reduce computational overhead.

For a fixed tokenizer $(\mathcal{V}, \tau)$ we define the **compression rate** of a byte-string $\mathbf{b}$ as

$$\mathrm{CR}(\mathbf{b}; \tau) \stackrel{\text{def}}{=} \frac{|\mathbf{b}|_u}{|\tau(\mathbf{b})|} \tag{1}$$

where $|\mathbf{b}|_u$ denotes the length of $\mathbf{b}$ in terms of a given **normalization unit** u (*e.g.*, characters, words, lines, or simply bytes). In words, $\mathrm{CR}(\mathbf{b}; \tau)$ measures the factor by which our original sequence length is reduced after tokenization. A higher CR indicates stronger compression.

We are generally interested in a tokenizer's average compression, which can be estimated from a corpus $\mathcal{D}$:

$$\mathrm{CR}(\mathcal{D}; \tau) \stackrel{\text{def}}{=} \frac{\sum_{\mathbf{b} \in \mathcal{D}} |\mathbf{b}|_u}{\sum_{\mathbf{b} \in \mathcal{D}} |\tau(\mathbf{b})|} \tag{2}$$

This quantity provides an estimate of the average number raw text units u that are packed into each token that an autoregressive LM processes. Tokenizers differ in how small they can make $\mathrm{CR}(\mathcal{D}; \tau)$ while remaining lossless. Further,

[4] To guarantee representability of any byte-string, we assume $\mathcal{V}$ includes all singleton bytes: $\mathcal{B} \subseteq \mathcal{V}$.

this rate can vary across strings from different languages, which motivates our last definition: language-specific compression rate.

Let $\mathcal{L} = \{\ell^{(r)}\}_{r=1}^{R}$ be our set of languages and $\mathcal{M} = \{(\mathbf{b}^{(s)}, \ell^{(s)})\}_{s=1}^{S}$ be a labeled multilingual corpus, *i.e.*, a corpus where each byte-string $\mathbf{b}^{(s)}$ is labeled with its language $\ell^{(s)}$. For a fixed tokenizer, we define a language's compression rate as

$$\mathrm{CR}(\ell; \tau) \stackrel{\text{def}}{=} \mathrm{CR}(\mathcal{D}_\ell; \tau) \tag{3}$$

where $\mathcal{D}_\ell = \{\mathbf{b}^{(s)} : (\mathbf{b}^{(s)}, \ell^{(s)}) \in \mathcal{M},\ \ell^{(s)} = \ell\}$.

2.3 Tokenization Fairness

Many deployed systems bill users per token; additionally, inference latency typically scales with token count. We therefore define tokenizer cost for language ℓ as the expected number of tokens required to encode a piece of content in ℓ. To make "content" comparable across languages, we measure cost using a parallel corpus and measure tokens per aligned segment (line/sentence/document). Inequality in these per-language costs corresponds to unequal user cost/latency purely due to language choice. We adopt the Gini coefficient across languages (defined in §5.2.1) as our primary fairness metric and success criterion, as it captures exactly this inequality. Intuitively, this metric can be thought of as a measure of how disparate $\mathrm{CR}(\ell; \tau)$ is across $\ell \in \mathcal{L}$. More details are given in §5.2.1.

3 Byte Pair Encoding

Here we provide a detailed description of the classic version of BPE; readers already familiar with the algorithm can skip this subsection.

Byte Pair Encoding (BPE; Sennrich et al., 2016) is one popular algorithm for creating a tokenizer adapted from the byte-pair compression scheme of Gage (1994). In short, BPE tokenizes text by iteratively *merging* adjacent tokens whose token-types (*i.e.*, subwords) were observed to co-occur frequently in the training data.

The notion of a **merge** lets us formalize this procedure. A merge is defined as an ordered pair $m = (v, v')$ with $v, v' \in \mathcal{V}$. The application of a merge to a token sequence replaces every bigram token v, v' by a single token $v \circ v'$. Each replacement shortens the token sequence by exactly one token, thereby *compressing* the sequence. To tokenize a piece of text with a BPE tokenizer, we start from its representation as a byte-string, *i.e.*, a sequence of base bytes, all of which necessarily appear in our tokenizer's vocabulary. We then iteratively apply a given list of merges $\mathbf{m}$ to that sequence, in the order in which they appear in $\mathbf{m}$. We denote this process with the function $\tau_{\mathbf{m}}$, *i.e.*, the tokenization function of a BPE tokenizer parameterized by $\mathbf{m}$. The vocabulary of this tokenizer is then $\mathcal{V} = \mathcal{B} \cup \{v \circ v' : (v, v') \in \mathbf{m}\}$. Note that because the merge list is fixed in advance, the encoding is deterministic. Intuition for the merge procedure is perhaps best acquired by a small example:

Example 3.1 (Example of the iterative application of merge sequence $\mathbf{m}$ to byte sequence $\mathbf{b}$)**.**

$$\mathbf{m} = [(\mathsf{b}, \mathsf{a}), (\mathsf{ba}, \mathsf{b})]; \quad \mathbf{b} = \mathsf{babab}$$
$$\mathbf{v}_0 = \mathsf{b}, \mathsf{a}, \mathsf{b}, \mathsf{a}, \mathsf{b}$$
$$\text{Step 1: } \mathsf{b}, \mathsf{a} \to \mathsf{ba} \implies \mathbf{v}_1 = \mathsf{ba}, \mathsf{ba}, \mathsf{b}$$
$$\text{Step 2: } \mathsf{ba}, \mathsf{b} \to \mathsf{bab} \implies \mathbf{v}_2 = \mathsf{ba}, \mathsf{bab}$$

Learning $\mathbf{m}$. The BPE algorithm seeks the merge list $\mathbf{m}^*$ (subject to a size constraint K) that maximizes the compression rate of the corpus $\mathcal{D}$:

$$\mathbf{m}^* = \max_{\mathbf{m}:|\mathbf{m}|=K} \mathrm{CR}(\mathcal{D}; \tau_{\mathbf{m}}) \tag{4}$$

BPE takes a greedy approach to choosing $\mathbf{m}$, finding an approximate solution to Eq. 4 (Zouhar et al., 2023b). It starts with the singleton-byte vocabulary $\mathcal{V}_0 = \mathcal{B}$ and repeatedly greedily enlarges the vocabulary. At each of K steps, the current tokenizer $\tau_{\mathbf{m}_{<k}}$ is applied to the entire training corpus, and the algorithm counts how often every adjacent pair of tokens occurs. The subword-type pair with the highest count, which we denote as $(v^\star, v'^\star)$, is deemed the most "compressive." Its concatenation $v^\star \circ v'^\star$ is added to the vocabulary, the merge $m_k = (v^\star, v'^\star)$ is recorded, and every occurrence of the bigram $(v^\star, v'^\star)$ in the corpus is replaced by the new token so the next iteration works with updated token sequences. Repeating this process K times yields the ordered list $\mathbf{m} = [m_1, \ldots, m_K]$ and the final vocabulary $\mathcal{V}_K$. When encoding a new text, $\tau_{\mathbf{m}}$ simply applies these merges in the same order. The algorithm pseudocode is provided in Alg. 1 of the Appendix.

4 Parity-aware Byte Pair Encoding

Classical BPE chooses merges that maximize a *global* frequency objective, implicitly favoring the compression of languages with a larger presence in the training corpus. Here we introduce **Parity-aware BPE**, which replaces this global objective with a *max–min* criterion: at every step, it selects the merge that most improves the worst-compressed language.

4.1 Greedy Fair-Max Objective

Our adjustment to the Classical BPE objective (Eq. 4) explicitly encodes our definition of tokenizer fairness: equality of compression rates across languages. Formally, Parity-aware BPE seeks a merge list $\mathbf{m} = [m_1, \ldots, m_K]$ that maximizes the *minimum* compression rate across languages:

$$\mathbf{m}^\star = \underset{\mathbf{m}:|\mathbf{m}|=K}{\operatorname{argmax}} \min_{\ell} \mathrm{CR}(\ell; \tau_{\mathbf{m}}) \tag{5}$$

We call this a fair-max objective; it trades a small amount of global compression for cross-lingual parity.

4.2 Algorithm

Parity-aware BPE retains the greedy iterative framework of Classical BPE but changes *which* statistics drive each merge decision. At step k, it identifies the worst-compressed language under the current tokenizer, *i.e.*, under the tokenizer defined by the merge list thus far ($\tau_{\mathbf{m}_{<k}}$):

$$\ell^\star \;=\; \arg\min_{\ell \in \mathcal{L}} \mathrm{CR}(\ell; \tau_{\mathbf{m}_{<k}}) \tag{6}$$

It then uses Classical BPE's maximum pair-count merge selection criterion, but restricted to $\mathcal{D}_{\ell^\star}$—the portion of the corpus in language $\ell^\star$. The rest of the algorithm follows the Classical BPE procedure: the chosen merge is applied to all texts (*i.e.*, across $\mathcal{D}_\ell\ \forall \ell$)[5] and the procedure is repeated for $k = 1, \ldots, K$, yielding the final merge list $\mathbf{m}$. We providepseudocode in Alg. 2 of the Appendix.

Cross-lingual Compression Rate Comparison. Parity-aware BPE relies on the comparison of $\mathrm{CR}(\ell; \tau_{\mathbf{m}})$ across different ℓ. The choice of normalization unit u has a large impact on the measured $\mathrm{CR}(\ell; \tau_{\mathbf{m}})$ and even when u is held constant across measurements for different languages, if not considered carefully, the choice can introduce bias into the comparison. As concrete examples, certain normalization units are more appropriate in some languages than in others, *e.g.*, whitespace-delimited "words" are ill-defined in many languages; although principled and universal, even normalizing by number of bytes can skew perceived compression because scripts differ greatly in average number of bytes per character (*e.g.*, ASCII vs. UTF-8 CJK). Parallel corpora provide a principled solution: computing compression rates over aligned texts (sentences, lines, or documents) normalizes by content, making cross-language comparisons more meaningful. We therefore recommend the use of a parallel corpus for computing Eq. 6. Notably, this evaluation corpus need not be the same one used for computing subword pair frequency statistics, for which a larger corpus with only language annotation is necessary. For generality, we thus differentiate between the corpora used to compute frequency statistics and in computing Eq. 6, referring to them as our training and development datasets, respectively. Alg. 2 makes this difference explicit. We present experimental results both with a separate, parallel development set and using a single (not parallel) multilingual dataset for all computations.

Complexity and Data Requirements. Relative to Classical BPE, Parity-aware BPE incurs only a $O(|\mathcal{L}|)$ overhead per-merge from recomputing the language-specific compression rates on the dev set. Parity-aware BPE retains the same asymptotic complexity as Classical BPE, requiring only some modest additional bookkeeping. The need for a multi-parallel corpus can at first seem prohibitive, but several pragmatic design choices can reduce the burden of this requirement. A small multi-parallel dataset suffices to drive the max-min decision in Eq. 6, and the training dataset need not be parallel. In addition, automatic language ID tools or script heuristics can help provide language labels when none are readily available. Also note that only the BPE learning phase differs; there is no algorithmic change to the tokenization function itself.

4.3 Algorithmic Variants

We now introduce several variants of Parity-aware BPE; these variants are designed to broaden its applicability to different developer goals and resource availability, as well as to address challenges identified in preliminary experiments.

Hybrid Parity-aware BPE. Developers may wish to have more fine-grained control of the balance between global compression and cross-lingual token count parity. We support these goals with a hybrid learning algorithm that uses the global objective of Classical BPE (Eq. 4) for the first J merges, then switches to the Parity-aware objective (Eq. 5) for another K merges. J and K can be chosen by model developers to trade off global compression and fairness according to their priorities.

[5] Crucially, this is what distinguishes our algorithm from a combination of monolingual merge lists (Petrov et al., 2023), allowing us to find more "compressive" merges.

Moving-window Balancing. There may be a point where the compression in a language no longer or barely improves, even if it is repeatedly chosen for the next merge. This could happen for several reasons: the development dataset (the dataset used to choose the language) may be too small or not match the domain or language variant of the training dataset;[6] alternatively, $|\tau(\mathbf{b})|$ may approach the length of the pre-tokenized sequence, meaning there are no more possible merges. To prevent our algorithm from being "stuck" selecting the same language exclusively, we track the W most recent languages selected in Eq. 6, and do not select a language if it occurs more than $\alpha\frac{W}{|\mathcal{L}|}$ times in this moving window.

Ratio-normalized (dev set-free) Language Selection Criterion. When multi-parallel development data is partially or completely unavailable, or when the concept of a parallel corpus is not applicable for one of the data domains (*e.g.*, for code or math data), developers can instead specify per-language compression targets explicitly. At a given merge step, the language whose current compression rate is furthest below its target is selected to determine the next merge. This compression rate can be computed directly on the training set, bypassing the need for a development set.

Formally, we replace the language selection criterion in Eq. 6 with

$$\ell^\star = \arg\min_{\ell \in \mathcal{L}} \frac{\mathrm{CR}(\ell;\, \tau_{\mathbf{m}_{<k}})}{r_\ell} \tag{7}$$

where $r_\ell > 0$ is a user-specified target value for language ℓ. These ratios r_ℓ can be interpreted as desired **relative compression rates** across languages. For instance, setting all r_ℓ equal recovers an equal-compression objective (albeit measured on non-parallel data), while setting r_ℓ proportional to the average bytes per line in a reference parallel corpus approximates the content-based normalization of the standard dev-set approach without requiring that corpus at training time. Because of the baseline differences in per-language UTF-8 encoding sizes, we recommend the latter approach. Developers can also use heuristics to approximate suitable ratios.

5 Experimental Setup

We conduct experiments to evaluate the effectiveness of Parity-aware BPE, comparing it against Classical BPE. All tokenizers are byte-level.

5.1 Tokenizer Training

Training Data. We train two sets of tokenizers: (1) using the multilingual C4 (mC4) corpus (Xue et al., 2021a; Raffel et al., 2020a)[7] (2) using the FineWeb2 dataset (Penedo et al., 2025). We focus on the results for tokenizers trained on FineWeb2; mC4 results can be found in §D.

Development Data. We use the dev portion of FLORES+ (NLLB Team et al., 2024)[8] as our multilingual development corpus, *i.e.*, for choosing the focus language at each merge step when training Parity-aware BPE tokenizers (Eq. 6). We also use this dataset to estimate per-language byte-level compression rate targets for the *ratio* system; we then compare this value to the per-language byte-level compression rate of the training corpus to choose the focus language.

Data Distribution. To investigate how the number of languages, their linguistic diversity, and the variety of writing systems influence tokenizers, we consider two language sets: one with 30 languages (*30-lang*) and another with 60 languages (*60-lang*). For each set, we construct two versions of the training data: (1) ***unbalanced*** **training:** In this setting, the tokenizer is trained on the naturally occurring mixture of languages, where high-resource languages exert greater influence on merge decisions due to their larger presence in the corpus. We determine the number of training examples per language using temperature-based sampling ($\tau = 3.3$, following prior work Raffel et al., 2020b; Conneau et al., 2020; Xue et al., 2021b). This setup reflects standard practice in multilingual LLM development, where the tokenizer training distribution mirrors the overall pretraining corpus distribution to avoid domain shift between tokenizer and model training (Dagan et al., 2024; Ali et al., 2024b). We derive the per-language sampling parameters according to per-language data proportions in the mC4 dataset and use the same values for creating both the mC4 and FineWeb2-based training datasets. (2) ***balanced*** **training:** Here, we sample the same number of documents per language. This ensures that each language contributes equally to the tokenizer's training, even without explicit interventions such as Parity-aware BPE. We present results for the *unbalanced* datasets here, as this

[6] Kreutzer et al. (2022) discuss possible quality issues such as wrong or ambiguous language codes.

[7] https://huggingface.co/datasets/allenai/c4

[8] https://huggingface.co/datasets/openlanguagedata/flores_plus

is arguably the more realistic setting, with results for the *balanced* setting shown in §D. To enable tokenizer analyses as a function of different dataset qualities, we categorize the languages in each set based on the amount of training data available and the script family, performing some of our analyses by these categories. Languages with $>$ 1M examples are considered high-resource; those with 500k $-$ 1M examples are medium-resource; and those with fewer than $<$ 500k examples are classified as low-resource. The full list of languages included in each set and the script family groupings are presented in Table 9 of the Appendix.

Hyperparameter Settings. We look at vocabulary sizes $128k$ and $256k$. For *hybrid* systems, we learn half of the merges using the global strategy, and the second half using the Parity-aware strategy (*i.e.*, $J = K = \frac{1}{2}|\mathbf{m}|$). For systems with moving-window balancing (*window*), we use a window size of 100, and $\alpha = 2$. More details and ablations over these hyperparameters can be found in §C and §D.

5.2 Evaluation

Our evaluations consist of task-independent tokenizer properties (intrinsic metrics) and downstream model performance (extrinsic metrics). Within a set of evaluations, we fix the data distribution (*balanced* or *unbalanced*) and vocabulary size ($128k$ or $256k$).

5.2.1 Intrinsic Metrics

We measure a variety of intrinsic tokenizer metrics on the devtest portion of FLORES+. These metrics encompass basic tokenization properties, information-theoretic measures, cross-linguistic fairness, and morphological alignment. Because of the parallel nature of FLORES+, we can use "lines" as our normalization unit for most metrics, which we motivate as a good default normalization unit in §4.2. For some intrinsic metrics, though, other normalization units are more appropriate, *e.g.*, words or characters may be more appropriate when morphologically motivated units can better reflect linguistic structure. We make this deviation from the default normalization unit explicit when applicable. For the sake of space, we provide brief metric descriptions here; more detailed metric definitions can be found in §B.

- **Fertility** measures the average number of tokens produced per normalization unit by a tokenizer; whitespace-delimited words are often the unit of interest (and are the units used in our computations). In this case, fertility quantifies how many tokens (on average) a word is broken up into (Rust et al., 2021).
- **Compression Rate (CR)** (as defined in §2) is a measure of the degree to which a unit of text has been shrunk after applying the given tokenizer (higher is better).
- **Vocabulary Utilization** is the fraction of the tokenizer's vocabulary that actually appears in the evaluation corpus. Low utilization for a language signals wasted capacity or, when there are large differences across languages, a vocabulary allocation that is biased towards certain languages.
- **Gini Token Inequality Coefficient** (Meister, 2025) adapts the Gini coefficient to the per-language tokenization cost distribution (*e.g.*, tokens per line (document) in a parallel corpus). Values near 0 mean equal cost across languages; values closer to 1 indicate inequality.
- **MorphScore** (Arnett et al., 2025b) measures how well token boundaries align with true morpheme boundaries, computed as morpheme-level precision/recall (and F1). High scores mean tokens respect morphological structure; low precision implies over-segmentation, while low recall may suggest under-segmentation.

For completeness, we also track Type–Token Ratio and Average Token Rank (vocabulary diversity; Limisiewicz et al., 2023) as well as Rényi entropies (distributional concentration; Zouhar et al., 2023a), evaluated using the TokEval suite (Meister, 2025).

5.2.2 Extrinsic Metrics.

For extrinsic evaluation, we train models using different tokenizers and assess their performance across a range of downstream tasks.

Model Architecture and Pretraining Data. We train decoder-only Transformer models (Vaswani et al., 2017) following the LLaMA architecture (Touvron et al., 2023) with 3B parameters. We use tokenizers trained on mC4. LMs are trained on 100B tokens from FineWeb2. To create the model training dataset, we adopt temperature sampling with $\tau = 3.3$, following recommendations from prior work (Raffel et al., 2020a; Conneau et al., 2020). Full details on model configurations and training parameters are provided in §F.

Benchmarks. We evaluate the models using perplexity on a held-out validation set from the respective pretraining datasets. In addition, we evaluate their multilingual performance using twelve widely

Tokenizer	Comp. Rate (↑)	TTR (↑)	Vocab Utilization (↑)	Fertility (↓)	Rényi (α=2.5) (↑)	Gini (↓)	MorphScore Precision (↑)	MorphScore Recall (↑)
BPE	0.0275	0.0777	67.0%	**3.990**	0.49	0.064	0.537	0.659
PA-BPE	0.0276	0.0819	70.4%	4.080	0.49	**0.007**	0.539	0.671
PA-BPE (window)	0.0278	**0.0838**	**71.6%**	4.042	0.48	0.009	0.546	**0.677**
PA-BPE (hybrid)	0.0278	0.0817	69.8%	4.056	0.49	0.015	0.541	0.667
PA-BPE (hybrid+window)	**0.0279**	0.0832	70.8%	4.029	0.48	0.019	**0.546**	0.670
PA-BPE (ratios)	0.0270	0.0737	64.9%	4.398	0.49	0.040	0.532	0.669

Table 1: Intrinsic evaluation of $128k$ tokenizers trained on the (*unbalanced*) *30-lang* dataset, evaluated on the corresponding *30-lang* subset of the FLORES+ dataset. With the exception of MorphScore—which is macro-averaged across available languages—values are global statistics across the parallel corpus.

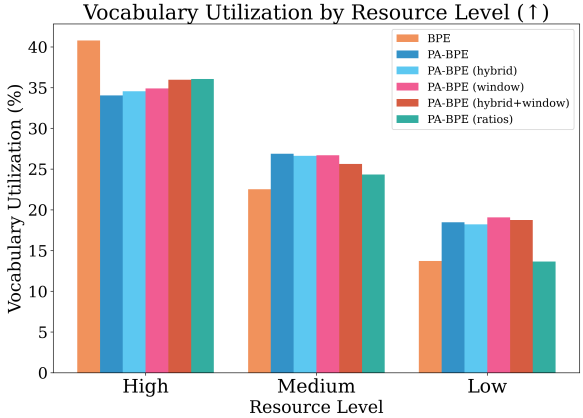

Figure 1: Vocabulary utilization for $128k$ tokenizers on the *unbalanced* *30-lang* dataset grouped by each language's resource level.

adopted benchmarks that collectively cover a broad range of tasks, including reading comprehension, commonsense reasoning, semantic similarity, and knowledge-based evaluation. These benchmarks are chosen to provide reliable and informative signals (results clearly above random chance) for small-scale models (1-3B parameters), while also supporting zero-shot evaluation without requiring supervised finetuning. The selected benchmarks covering 22 languages are Belebele, mTruthfulQA, PAWS-X, XCodah, XCSQA, XNLI, XStoryCloze, XWinogrande, MMMLU, INCLUDE, Exams, and M3Exams. Results are aggregated per language to produce a score for each model-language pair. A full list of benchmarks and aggregation procedures is provided in §G.

6 Results and Analysis

6.1 Intrinsic Evaluation

Results in Table 1 show that all variants of Parity-aware BPE substantially reduce the Gini coefficient relative to Classical BPE (from 0.064 to $\leq$ 0.019 for dev-set variants), indicating more equitable token costs across languages. Among these variants, the base Parity-aware BPE achieves the lowest Gini, *i.e.*, it is the "fairest" tokenizer. Average per-language vocabulary utilization also improves slightly for the dev-set variants compared to Classical BPE. Global compression rates and Rényi entropies remain essentially identical across all BPE variants, confirming that the fairness gain does not come at a measurable cost to overall compression efficiency. Other intrinsic metrics remain relatively stable across all Parity-aware variants; we treat these as auxiliary diagnostics, confirming that the fairness improvement does not negatively affect other tokenizer properties.

We show per-language results for compression rate in Fig. 2, which illustrates the core result for the method: Parity-aware tokenizers yield substantially more uniform compression across languages than Classical BPE, consistent with the Gini reduction reported in Table 1. Notably, as visible in both Table 1 and Fig. 2, the *ratios* variant achieves the smallest Gini reduction of any of the Parity-aware variants. This is likely attributable to domain mismatch: the per-language compression targets are derived from bytes-per-line statistics in the parallel FLORES+ dataset, which is necessarily cross-lingually domain-aligned. However, the statistics used for merge selection are computed on FineWeb2, whose domain and genre composition may differ substantially across languages. Because data from different domains can require different levels of tokenization granularity (*e.g.*, technical terminology may demand finer-grained segmentation), FLORES-derived ratios may not be ideal for achieving content-controlled parity——as operationalized by the Gini coefficient on FLORES+—on the FineWeb2 training corpus. Nevertheless, this variant remains practical alternative to improve upon Classical BPE when multi-parallel data is limited.

Fig. 1 presents vocabulary utilization grouped by resource tier, as defined in §5.1. Parity-aware tokenizers provide more consistent usage across languages, evening out the vocabulary allocation to high vs. low resource languages in comparison to Classical BPE. As with other intrinsic metrics,

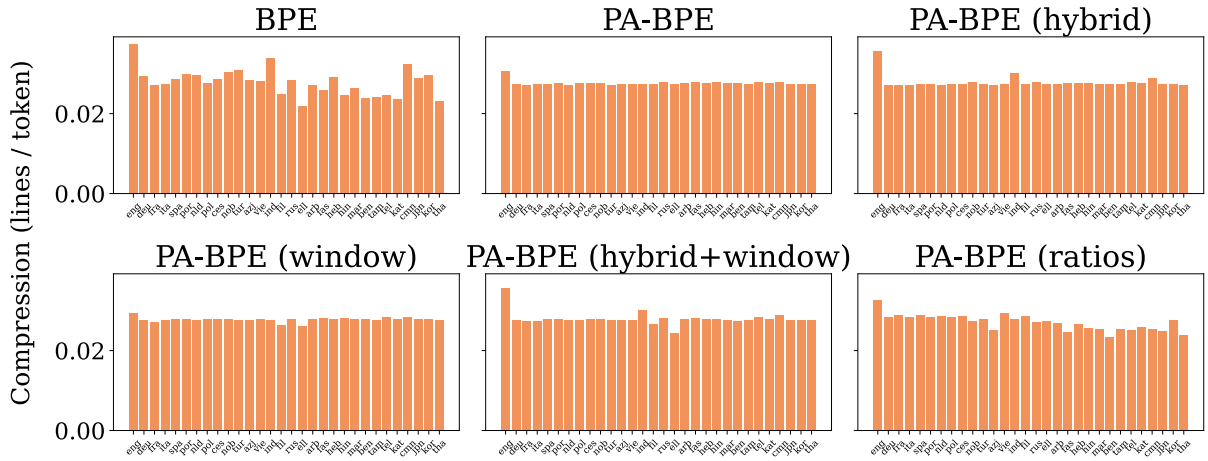

Figure 2: Per-language compression rate (↑) of FLORES+ parallel sentences when tokenized using $128k$ tokenizers trained on the (*unbalanced*) *30-lang* per language.

Language	Classical BPE	Parity-aware (hybrid)	Parity-aware (window+hybrid)	Random
Arabic	38.19 ± 2.90	**39.04** ± 2.89	38.84 ± 2.90	32.00
Bengali	24.95 ± 3.09	23.54 ± 2.98	23.91 ± 3.01	**25.00**
German	32.92 ± 3.14	34.78 ± 3.66	**36.82** ± 4.04	30.62
Greek	41.95 ± 3.18	42.55 ± 3.22	**43.16** ± 3.25	37.50
Spanish	37.53 ± 2.66	38.83 ± 2.71	**39.30** ± 2.75	32.77
Persian	**42.80** ± 5.39	39.15 ± 5.27	39.15 ± 5.27	25.00
French	**38.67** ± 3.90	36.59 ± 2.84	37.10 ± 2.82	32.00
Hindi	**33.92** ± 2.25	33.92 ± 2.24	33.86 ± 2.24	30.62
Indonesian	38.95 ± 2.62	**40.55** ± 2.66	40.46 ± 2.66	35.00
Italian	32.82 ± 2.86	**35.01** ± 3.00	34.62 ± 2.98	27.22
Japanese	37.43 ± 2.39	**37.45** ± 2.39	37.43 ± 2.39	34.00
Korean	33.00 ± 5.22	33.00 ± 5.22	**34.33** ± 5.29	25.00
Polish	29.75 ± 2.50	**31.14** ± 2.60	28.97 ± 2.49	23.75
Portuguese	**33.63** ± 2.81	33.15 ± 2.77	33.06 ± 2.77	27.50
Russian	36.36 ± 2.27	36.21 ± 2.26	**36.57** ± 2.28	32.77
Tamil	31.32 ± 2.81	**32.25** ± 2.90	32.19 ± 2.90	31.25
Telugu	32.73 ± 2.61	**33.52** ± 2.61	33.26 ± 2.61	30.00
Turkish	**39.04** ± 2.89	38.46 ± 2.83	37.89 ± 2.75	35.00
Vietnamese	33.69 ± 2.31	**33.87** ± 2.27	33.80 ± 2.30	29.50
Chinese	38.43 ± 2.11	**38.58** ± 2.11	38.32 ± 2.10	35.00
English	43.04 ± 1.84	**44.15** ± 1.85	43.74 ± 1.85	35.50
Thai	40.76 ± 1.62	40.96 ± 1.63	**41.06** ± 1.63	37.50

Table 2: Average downstream performance (accuracy %) across 12 multilingual benchmarks (tokenizers trained on the (*unbalanced*) *30-lang* dataset with $128k$ vocab size). The **Random** column shows the expected accuracy of a random classifier. Best performance per language is bolded. Benchmark details for each language are provided in §G and Table 10, respectively.

the ratios variant exhibits the least consistency in cross-resource-level vocabulary utilization among the Parity-aware variants.

Ablations. We report additional experiments in §D and §C, varying the tokenizer training corpus (mC4), vocabulary size (256k), per-language data allocation (balanced vs. unbalanced), window size for the window variant ($W = 50, 100, 150, 200$), dev set size ($N = 50, 100, 300, 1000$), and number of languages (30 vs. 60). We see consistency in results across all settings: Parity-aware BPE variants achieve substantially lower Gini coefficients than Classical BPE, while compression rates and other intrinsic metrics remain comparable.

6.2 Extrinsic Evaluation

Table 2 presents the performance of LMs trained with three different $128k$ tokenizers: Classical BPE, Hybrid Parity-aware BPE, and Hybrid Parity-aware BPE (moving-window), evaluated on a subset of the *30-lang* set (22 languages). For each language, we report mean performance, standard errors, and a random baseline to account for varying benchmark counts (Table 10). We treat downstream evaluation as a regression check to assess whether the fairness gains come at the expense of downstream performance. Results indicate that Parity-aware BPE maintains performance across languages: for models trained with the hybrid variant, we see nominal gains in 14 languages and nominal declines in 6. Individual differences are generally within standard errors, and thus, we conclude that we find no evidence that Parity-aware tokenizers would compromise downstream LM performance. Additionally, per-language perplexity in Appendix Fig. 5 show that Classical BPE produces models for which a subset of languages have notably higher perplexity. These results suggest that models trained with Parity-aware tokenizers can exhibit more uniform perplexity across languages.

7 Related Work

Multilingual Tokenization. Despite their popularity, BPE and related subword tokenization methods often underperform in multilingual settings due to limited handling of spelling variation and morphological complexity (Bostrom and Durrett, 2020). A tokenizer's tokenization parity and fertility directly impact both computational cost and model performance (Lesci et al., 2025). Prior

work has explored vocabulary allocation strategies: Zhang et al. (2022) show that larger vocabularies improve NMT robustness across scripts, while Gowda and May (2020) demonstrate that tuning BPE merges can mitigate sequence length issues. Rust et al. (2021) find that integrating specialized monolingual tokenizers into multilingual systems can boost performance; however, recent evidence suggests that optimal vocabulary size depends on the task and model (Dagan et al., 2024). For multilingual vocabulary construction, Chung et al. (2020) explore clustering-based sharing of subwords across languages, and Limisiewicz et al. (2023) propose an explicit tokenizer-merging algorithm to combine per-language vocabularies. Finally, tokenization-free models such as CANINE (Clark et al., 2022) and ByT5 (Xue et al., 2022) offer an alternative approach for improved multilingual handling.

Tokenization Bias and Recent Advances. Recent research highlights biases from tokenization in LLMs. While Wan (2022) argue that character- and byte-level representations are intrinsically fair,[9] other studies (Petrov et al., 2023; Ahia et al., 2023) show that tokenization disparities across languages; even at character and byte levels; affect costs, latency, and contextual understanding. This has spurred efforts like Aya (Aryabumi et al., 2024) and methods to mitigate tokenization unfairness (Fujii et al., 2024; Abboud and Oz, 2024; Limisiewicz et al., 2024). Although newer character- and byte-level models use compression techniques such as entropy-based patching (Pagnoni et al., 2025), cross-lingual parity of these representations remains unstudied.

8 Discussion and Conclusion

Tokenizers optimized using standard algorithms can lead to disparities in users' costs and experiences depending on language choice. Parity-optimized tokenization can remedy this by explicitly balancing compression across languages, enabling fairer treatment of users of low-resource languages. Parity-aware BPE operationalizes this idea; it is designed to improve cross-lingual tokenization parity. Our experiments confirm its effectiveness: we observe up to an 89% reduction in the Gini token inequality coefficient in comparison to Classical BPE, while compression rates across all BPE variants remain very close. Crucially, this fairness gain does not come at the expense of downstream quality: across 22 languages, accuracy differences between Parity-aware and Classical BPE tokenizers fall within standard errors for nearly all languages (Table 2). In other words, we find no evidence of systematic downstream degradation from using Parity-aware BPE.

Overall, the trade-offs associated for using Parity-aware BPE are minimal. From the model-developer's perspective, Parity-aware BPE is a drop-in replacement: it requires no architectural changes and only minimal modifications to the tokenizer training pipeline. Concretely, inference remains identical to that Classical BPE. During the learning stage, the algorithm adds only an $\mathcal{O}(\mathcal{L})$ pass per merge to recompute language-level compression rates on a dev corpus, leaving the asymptotic complexity identical to Classical BPE. Moreover, we find empirically that a small, sentence-aligned development set is sufficient to drive the fair-max decision (§C.3). When resource or domain mismatches make full equality undesirable, hybrid, moving-window, and ratio-normalized variants further let practitioners trade off global compression versus strict parity; our empirical results validate that these variants perform well in practice.

Improving the equity of the tokenization step in the NLP pipeline is therefore not just desirable but feasible. With a simple modification—selecting merges that benefit the worst-compressed language—Parity-aware BPE substantially reduces the hidden "token tax" imposed on speakers of low-resource languages without sacrificing compression or task accuracy.

Future research can extend this agenda by adapting the "fair-max" objective to alternative tokenization schemes and diverse modalities, such as speech and vision. For merge-based methods like WordPiece, selection criteria can be modified to prioritize merges that provide the greatest likelihood gains for the most poorly compressed language. Similarly, for non-merge-based algorithms like UnigramLM, the parity objective can be integrated into the pruning stage to preserve tokens essential for low-resource language efficiency. Moreover, future work should develop robust benchmarks and metrics for fairness assessments that go beyond simple compression parity to ensure truly equitable processing across all linguistic and sensory inputs.

[9] They report more random performance with these representations, which we do not view as "fairer."

Limitations

Several sources of algorithmic bias can cause cross-lingual inequity in token counts to persist even when using Parity-aware BPE. First, the standard variants of Parity-aware BPE use parallel corpora to estimate per-language costs; in domains where aligned documents are unavailable or the per-languages ratios in the ratio variant can only be heuristically estimated, the algorithm's ability to identify the truly worst-compressed language at each merge step may be degraded. Second, BPE cannot compress a text beyond its pre-tokenized length. If $|\tau(\mathbf{b})|$ approaches the length of the pre-tokenized sequence for one language, any BPE variant (including Parity-aware BPE) will yield diminishing compression gains for that language. In particular, when using the moving-window balancing variant of Parity-aware BPE, the algorithm responds by relaxing the parity objective. This will also allow some cross-lingual disparities to persist.

Future work has several opportunities to address these limitations: better approaches to estimating ratios for the ratio variant, alternative variants that bypass the need for a parallel development set entirely, and alternative pretokenization strategies that ensure some languages are not disadvantaged by the pretoken splitting boundaries—for example, via integration with an approach such as SuperBPE (Liu et al., 2026).

More broadly, while we consider 60 languages and two vocabulary sizes, the interplay between tokenization parity and model scaling still needs to be explored for much larger models, larger language sets and for code or multimodal inputs. Finally, fairness in this work is defined purely in terms of token counts. While we measure other potential quantifications of fairness (*e.g.*, morphological alignment), there are still other notions that are unaccounted for. We leave optimization for these metrics during tokenizer learning to future work.

Acknowledgement

RS acknowledges support by the Swiss National Science Foundation through the MUTAMUR project (no. 213976). We gratefully acknowledge the support of the Swiss National Science Foundation (No. 215390), Innosuisse (PFFS-21-29), the European Research Council (Starting grant no. 101222478, RESPECT-LM), the AI2050 program at Schmidt Sciences (Grant #G-25-69783), Sony Group Corporation, and the Swiss National Supercomputing Center (CSCS) in the form of an infrastructure engineering and development project. We thank Tiago Pimentel for his feedback on earlier drafts of this manuscript.

References

Khadige Abboud and Gokmen Oz. 2024. Towards equitable natural language understanding systems for dialectal cohorts: Debiasing training data. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation, LREC/COLING 2024, 20-25 May, 2024, Torino, Italy*, pages 16487–16499. ELRA and ICCL.

Orevaoghene Ahia, Sachin Kumar, Hila Gonen, Jungo Kasai, David R. Mortensen, Noah A. Smith, and Yulia Tsvetkov. 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, EMNLP 2023, Singapore, December 6-10, 2023*, pages 9904–9923. Association for Computational Linguistics.

Mehdi Ali, Michael Fromm, Klaudia Thellmann, Richard Rutmann, Max Lübbering, Johannes Leveling, Katrin Klug, Jan Ebert, Niclas Doll, Jasper Buschhoff, Charvi Jain, Alexander Weber, Lena Jurkschat, Hammam Abdelwahab, Chelsea John, Pedro Ortiz Suarez, Malte Ostendorff, Samuel Weinbach, Rafet Sifa, Stefan Kesselheim, and Nicolas Flores-Herr. 2024a. Tokenizer choice for LLM training: Negligible or crucial? In *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 3907–3924, Mexico City, Mexico. Association for Computational Linguistics.

Mehdi Ali, Michael Fromm, Klaudia Thellmann, Richard Rutmann, Max Lübbering, Johannes Leveling, Katrin Klug, Jan Ebert, Niclas Doll, Jasper Buschhoff, et al. 2024b. Tokenizer choice for llm training: Negligible or crucial? In *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 3907–3924.

Alejandro Hernández-Cano Apertus Project, Alexander Hägele, Allen Hao Huang, Angelika Romanou, Antoni-Joan Solergibert, Barna Pasztor, Bettina Messmer, Dhia Garbaya, Eduard Frank Ďurech, Ido Hakimi, Juan García Giraldo, Mete Ismayilzada, Negar Foroutan, Skander Moalla, Tiancheng Chen, Vinko Sabolčec, Yixuan Xu, Michael Aerni, Badr AlKhamissi, Ines Altemir Marinas, Mohammad Hossein Amani, Matin Ansaripour, Ilia Badanin, Harold Benoit, Emanuela Boros, Nicholas Browning, Fabian Bösch, Maximilian Böther, Niklas Canova, Camille Challier, Clement Charmillot, Jonathan Coles, Jan Deriu, Arnout Devos, Lukas Drescher, Daniil Dzenhaliou, Maud Ehrmann, Dongyang Fan, Simin Fan, Silin Gao, Miguel Gila, María Grandury, Diba Hashemi, Alexander Hoyle, Jiaming Jiang, Mark

Klein, Andrei Kucharavy, Anastasiia Kucherenko, Frederike Lübeck, Roman Machacek, Theofilos Manitaras, Andreas Marfurt, Kyle Matoba, Simon Matrenok, Henrique Mendoncça, Fawzi Roberto Mohamed, Syrielle Montariol, Luca Mouchel, Sven Najem-Meyer, Jingwei Ni, Gennaro Oliva, Matteo Pagliardini, Elia Palme, Andrei Panferov, Léo Paoletti, Marco Passerini, Ivan Pavlov, Auguste Poiroux, Kaustubh Ponkshe, Nathan Ranchin, Javi Rando, Mathieu Sauser, Jakhongir Saydaliev, Muhammad Ali Sayfiddinov, Marian Schneider, Stefano Schuppli, Marco Scialanga, Andrei Semenov, Kumar Shridhar, Raghav Singhal, Anna Sotnikova, Alexander Sternfeld, Ayush Kumar Tarun, Paul Teiletche, Jannis Vamvas, Xiaozhe Yao, Hao Zhao Alexander Ilic, Ana Klimovic, Andreas Krause, Caglar Gulcehre, David Rosenthal, Elliott Ash, Florian Tramèr, Joost VandeVondele, Livio Veraldi, Martin Rajman, Thomas Schulthess, Torsten Hoefler, Antoine Bosselut, Martin Jaggi, and Imanol Schlag. 2025. Apertus: Democratizing open and compliant llms for global language environments. *arXiv e-prints*, pages arXiv–2509.

Catherine Arnett, Tyler A. Chang, Stella Biderman, and Benjamin K. Bergen. 2025a. Explaining and mitigating crosslingual tokenizer inequities. In *Advances in Neural Information Processing Systems 38 (NeurIPS 2025)*.

Catherine Arnett, Marisa Hudspeth, and Brendan O'Connor. 2025b. Evaluating Morphological Alignment of Tokenizers in 70 Languages. In *Proceedings of the ICML 2025 Tokenization Workshop (TokShop)*.

Viraat Aryabumi, John Dang, Dwarak Talupuru, Saurabh Dash, David Cairuz, Hangyu Lin, Bharat Venkitesh, Madeline Smith, Kelly Marchisio, Sebastian Ruder, et al. 2024. Aya 23: Open weight releases to further multilingual progress. *CoRR*, abs/2405.15032.

Lucas Bandarkar, Davis Liang, Benjamin Muller, Mikel Artetxe, Satya Narayan Shukla, Donald Husa, Naman Goyal, Abhinandan Krishnan, Luke Zettlemoyer, and Madian Khabsa. 2024. The Belebele benchmark: a parallel reading comprehension dataset in 122 language variants. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 749–775, Bangkok, Thailand. Association for Computational Linguistics.

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

Michael Chen, Mike D'Arcy, Alisa Liu, Jared Fernandez, and Doug Downey. 2019. CODAH: An adversarially-authored question answering dataset for common sense. In *Proceedings of the 3rd Workshop on Evaluating Vector Space Representations for NLP*, pages 63–69. Association for Computational Linguistics.

Hyung Won Chung, Dan Garrette, Kiat Chuan Tan, and Jason Riesa. 2020. Improving multilingual models with language-clustered vocabularies. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 4536–4546. Association for Computational Linguistics.

Jonathan H. Clark, Dan Garrette, Iulia Turc, and John Wieting. 2022. CANINE: Pre-training an efficient tokenization-free encoder for language representation. *Trans. Assoc. Comput. Linguistics*, 10:73–91.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451. Association for Computational Linguistics.

Alexis Conneau, Ruty Rinott, Guillaume Lample, Adina Williams, Samuel Bowman, Holger Schwenk, and Veselin Stoyanov. 2018. XNLI: Evaluating cross-lingual sentence representations. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2475–2485, Brussels, Belgium. Association for Computational Linguistics.

Gautier Dagan, Gabriel Synnaeve, and Baptiste Rozière. 2024. Getting the most out of your tokenizer for pre-training and domain adaptation. In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24. JMLR.org.

Kazuki Fujii, Taishi Nakamura, Mengsay Loem, Hiroki Iida, Masanari Ohi, Kakeru Hattori, Hirai Shota, Sakae Mizuki, Rio Yokota, and Naoaki Okazaki. 2024. Continual pre-training for cross-lingual LLM adaptation: Enhancing japanese language capabilities. In *First Conference on Language Modeling*.

Philip Gage. 1994. A new algorithm for data compression. *C Users Journal*, 12(2):23–38.

Omer Goldman, Avi Caciularu, Matan Eyal, Kris Cao, Idan Szpektor, and Reut Tsarfaty. 2024. Unpacking tokenization: Evaluating text compression and its correlation with model performance. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 2274–2286, Bangkok, Thailand. Association for Computational Linguistics.

Thamme Gowda and Jonathan May. 2020. Finding the optimal vocabulary size for neural machine translation. In *Findings of the Association for Computational Linguistics: EMNLP*, volume EMNLP 2020 of *Findings of ACL*, pages 3955–3964. Association for Computational Linguistics.

Nathan Habib, Clémentine Fourrier, Hynek Kydlíček, Thomas Wolf, and Lewis Tunstall. 2023. Lighteval: A lightweight framework for llm evaluation.

Alexander Hägele, Elie Bakouch, Atli Kosson, Loubna Ben allal, Leandro Von Werra, and Martin Jaggi. 2024. Scaling laws and compute-optimal training beyond fixed training durations. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Momchil Hardalov, Todor Mihaylov, Dimitrina Zlatkova, Yoan Dinkov, Ivan Koychev, and Preslav Nakov. 2020. EXAMS: A multi-subject high school examinations dataset for cross-lingual and multilingual question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5427–5444, Online. Association for Computational Linguistics.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021. Measuring massive multitask language understanding. *Proceedings of the International Conference on Learning Representations (ICLR)*.

Julia Kreutzer, Isaac Caswell, Lisa Wang, Ahsan Wahab, Daan van Esch, Nasanbayar Ulzii-Orshikh, Allahsera Tapo, Nishant Subramani, Artem Sokolov, Claytone Sikasote, Monang Setyawan, Supheakmungkol Sarin, Sokhar Samb, Benoît Sagot, Clara Rivera, Annette Rios, Isabel Papadimitriou, Salomey Osei, Pedro Ortiz Suarez, Iroro Orife, Kelechi Ogueji, Andre Niyongabo Rubungo, Toan Q. Nguyen, Mathias Müller, André Müller, Shamsuddeen Hassan Muhammad, Nanda Muhammad, Ayanda Mnyakeni, Jamshidbek Mirzakhalov, Tapiwanashe Matangira, Colin Leong, Nze Lawson, Sneha Kudugunta, Yacine Jernite, Mathias Jenny, Orhan Firat, Bonaventure F. P. Dossou, Sakhile Dlamini, Nisansa de Silva, Sakine Çabuk Ballı, Stella Biderman, Alessia Battisti, Ahmed Baruwa, Ankur Bapna, Pallavi Baljekar, Israel Abebe Azime, Ayodele Awokoya, Duygu Ataman, Orevaoghene Ahia, Oghenefego Ahia, Sweta Agrawal, and Mofetoluwa Adeyemi. 2022. Quality at a glance: An audit of web-crawled multilingual datasets. *Transactions of the Association for Computational Linguistics*, 10:50–72.

Taku Kudo. 2018. Subword regularization: Improving neural network translation models with multiple subword candidates. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL*, pages 66–75. Association for Computational Linguistics.

Viet Lai, Chien Nguyen, Nghia Ngo, Thuat Nguyen, Franck Dernoncourt, Ryan Rossi, and Thien Nguyen. 2023. Okapi: Instruction-tuned large language models in multiple languages with reinforcement learning from human feedback. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 318–327. Association for Computational Linguistics.

Pietro Lesci, Clara Meister, Thomas Hofmann, Andreas Vlachos, and Tiago Pimentel. 2025. Causal estimation of tokenisation bias. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 28325–28340, Vienna, Austria. Association for Computational Linguistics.

Tomasz Limisiewicz, Jiří Balhar, and David Mareček. 2023. Tokenization impacts multilingual language modeling: Assessing vocabulary allocation and overlap across languages. In *Findings of the Association for Computational Linguistics*, pages 5661–5681. Association for Computational Linguistics.

Tomasz Limisiewicz, Terra Blevins, Hila Gonen, Orevaoghene Ahia, and Luke Zettlemoyer. 2024. MYTE: Morphology-driven byte encoding for better and fairer multilingual language modeling. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics*, pages 15059–15076. Association for Computational Linguistics.

Bill Yuchen Lin, Seyeon Lee, Xiaoyang Qiao, and Xiang Ren. 2021. Common sense beyond English: Evaluating and improving multilingual language models for commonsense reasoning. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1274–1287, Online. Association for Computational Linguistics.

Stephanie Lin, Jacob Hilton, and Owain Evans. 2022a. TruthfulQA: Measuring how models mimic human falsehoods. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3214–3252, Dublin, Ireland. Association for Computational Linguistics.

Xi Victoria Lin, Todor Mihaylov, Mikel Artetxe, Tianlu Wang, Shuohui Chen, Daniel Simig, Myle Ott, Naman Goyal, Shruti Bhosale, Jingfei Du, Ramakanth Pasunuru, Sam Shleifer, Punit Singh Koura, Vishrav Chaudhary, Brian O'Horo, Jeff Wang, Luke Zettlemoyer, Zornitsa Kozareva, Mona T. Diab, Veselin Stoyanov, and Xian Li. 2022b. Few-shot learning with multilingual generative language models. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 9019–9052. Association for Computational Linguistics.

Alisa Liu, Jonathan Hayase, Valentin Hofmann, Sewoong Oh, Noah A. Smith, and Yejin Choi. 2026. SuperBPE: Space travel for language models. In *Tokenization Workshop*.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *International Conference on Learning Representations*.

Pedro Henrique Martins, João Alves, Patrick Fernandes, Nuno M Guerreiro, Ricardo Rei, Amin Farajian, Mateusz Klimaszewski, Duarte M Alves, José Pombal, Nicolas Boizard, et al. 2025. Eurollm-9b: Technical report. *arXiv preprint arXiv:2506.04079*.

Clara Meister. 2025. TokEval: A tokenizer analysis suite.

Nasrin Mostafazadeh, Michael Roth, Annie Louis, Nathanael Chambers, and James Allen. 2017. LSDSem 2017 shared task: The story cloze test. In *Proceedings of the 2nd Workshop on Linking Models of Lexical, Sentential and Discourse-level Semantics*, pages 46–51.

Niklas Muennighoff, Thomas Wang, Lintang Sutawika, Adam Roberts, Stella Biderman, Teven Le Scao, M Saiful Bari, Sheng Shen, Zheng Xin Yong, Hailey Schoelkopf, Xiangru Tang, Dragomir Radev, Alham Fikri Aji, Khalid Almubarak, Samuel Albanie, Zaid Alyafeai, Albert Webson, Edward Raff, and Colin Raffel. 2023. Crosslingual generalization through multitask finetuning. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15991–16111, Toronto, Canada. Association for Computational Linguistics.

NLLB Team, Marta R. Costa-jussà, James Cross, Onur Çelebi, Maha Elbayad, Kenneth Heafield, Kevin Heffernan, Elahe Kalbassi, Janice Lam, Daniel Licht, Jean Maillard, Anna Sun, Skyler Wang, Guillaume Wenzek, Al Youngblood, Bapi Akula, Loic Barrault, Gabriel Mejia Gonzalez, Prangthip Hansanti, John Hoffman, Semarley Jarrett, Kaushik Ram Sadagopan, Dirk Rowe, Shannon Spruit, Chau Tran, Pierre Andrews, Necip Fazil Ayan, Shruti Bhosale, Sergey Edunov, Angela Fan, Cynthia Gao, Vedanuj Goswami, Francisco Guzmán, Philipp Koehn, Alexandre Mourachko, Christophe Ropers, Safiyyah Saleem, Holger Schwenk, and Jeff Wang. 2024. Scaling neural machine translation to 200 languages. *Nature*, 630(8018):841–846.

Artidoro Pagnoni, Ramakanth Pasunuru, Pedro Rodríguez, John Nguyen, Benjamin Muller, Margaret Li, Chunting Zhou, Lili Yu, Jason E. Weston, Luke Zettlemoyer, Gargi Ghosh, Mike Lewis, Ari Holtzman, and Srini Iyer. 2025. Byte latent transformer: Patches scale better than tokens. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 9238–9258. ACL.

Guilherme Penedo, Hynek Kydlíček, Vinko Sabolčec, Bettina Messmer, Negar Foroutan, Amir Hossein Kargaran, Colin Raffel, Martin Jaggi, Leandro Von Werra, and Thomas Wolf. 2025. Fineweb2: One pipeline to scale them all — adapting pre-training data processing to every language. In *Second Conference on Language Modeling*.

Aleksandar Petrov, Emanuele La Malfa, Philip H. S. Torr, and Adel Bibi. 2023. Language model tokenizers introduce unfairness between languages. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020a. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21(1).

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020b. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21(140):1–67.

Angelika Romanou, Negar Foroutan, Anna Sotnikova, Sree Harsha Nelaturu, Shivalika Singh, Rishabh Maheshwary, Micol Altomare, Zeming Chen, Mohamed A. Haggag, Snegha A, Alfonso Amayuelas, Azril Hafizi Amirudin, Danylo Boiko, Michael Chang, Jenny Chim, Gal Cohen, Aditya Kumar Dalmia, Abraham Diress, Sharad Duwal, Daniil Dzenhaliou, Daniel Fernando Erazo Florez, Fabian Farestam, Joseph Marvin Imperial, Shayekh Bin Islam, Perttu Isotalo, Maral Jabbarishiviari, Börje F. Karlsson, Eldar Khalilov, Christopher Klamm, Fajri Koto, Dominik Krzemiński, Gabriel Adriano de Melo, Syrielle Montariol, Yiyang Nan, Joel Niklaus, Jekaterina Novikova, Johan Samir Obando Ceron, Debjit Paul, Esther Ploeger, Jebish Purbey, Swati Rajwal, Selvan Sunitha Ravi, Sara Rydell, Roshan Santhosh, Drishti Sharma, Marjana Prifti Skenduli, Arshia Soltani Moakhar, Bardia Soltani Moakhar, Ayush Kumar Tarun, Azmine Toushik Wasi, Thenuka Ovin Weerasinghe, Serhan Yilmaz, Mike Zhang, Imanol Schlag, Marzieh Fadaee, Sara Hooker, and Antoine Bosselut. 2025. INCLUDE: Evaluating multilingual language understanding with regional knowledge. In *The Thirteenth International Conference on Learning Representations*.

Phillip Rust, Jonas Pfeiffer, Ivan Vulic, Sebastian Ruder, and Iryna Gurevych. 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, ACL/IJCNLP 2021*, pages 3118–3135. ACL.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2021. WinoGrande: an adversarial winograd schema challenge at scale. *Commun. ACM*, 64(9):99–106.

Craig W Schmidt, Varshini Reddy, Haoran Zhang, Alec Alameddine, Omri Uzan, Yuval Pinter, and Chris Tanner. 2024. Tokenization is more than compression. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 678–702. Association for Computational Linguistics.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. 2019. CommonsenseQA: A question answering challenge targeting commonsense knowledge. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1*, pages 4149–4158. Association for Computational Linguistics.

Alexey Tikhonov and Max Ryabinin. 2021. It's all in the heads: Using attention heads as a baseline for cross-lingual transfer in commonsense reasoning. In *Findings of the Association for Computational Linguistics: ACL/IJCNLP 2021, Online Event, August 1-6, 2021*, volume ACL/IJCNLP 2021 of *Findings of ACL*, pages 3534–3546. Association for Computational Linguistics.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurélien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023. LLaMA: Open and efficient foundation language models. *CoRR*, abs/2302.13971.

Ahmet Üstün, Viraat Aryabumi, Zheng Yong, Wei-Yin Ko, Daniel D'souza, Gbemileke Onilude, Neel Bhandari, Shivalika Singh, Hui-Lee Ooi, Amr Kayid, et al. 2024. Aya model: An instruction finetuned open-access multilingual language model. In *Proceedings of the 62nd annual meeting of the Association for Computational Linguistics (volume 1: Long papers)*, pages 15894–15939.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems*, pages 5998–6008.

Ada Wan. 2022. Fairness in representation for multilingual NLP: Insights from controlled experiments on conditional language modeling. In *International Conference on Learning Representations*.

Linting Xue, Aditya Barua, Noah Constant, Rami Al-Rfou, Sharan Narang, Mihir Kale, Adam Roberts, and Colin Raffel. 2022. ByT5: Towards a token-free future with pre-trained byte-to-byte models. *Trans. Assoc. Comput. Linguistics*, 10:291–306.

Linting Xue, Noah Constant, Adam Roberts, Mihir Kale, Rami Al-Rfou, Aditya Siddhant, Aditya Barua, and Colin Raffel. 2021a. mT5: A massively multilingual pre-trained text-to-text transformer. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 483–498, Online. Association for Computational Linguistics.

Linting Xue, Noah Constant, Adam Roberts, Mihir Kale, Rami Al-Rfou, Aditya Siddhant, Aditya Barua, and Colin Raffel. 2021b. mt5: A massively multilingual pre-trained text-to-text transformer. In *Proceedings of the 2021 conference of the North American chapter of the association for computational linguistics: Human language technologies*, pages 483–498.

Yinfei Yang, Yuan Zhang, Chris Tar, and Jason Baldridge. 2019. PAWS-X: A cross-lingual adversarial dataset for paraphrase identification. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3687–3692, Hong Kong, China. Association for Computational Linguistics.

Shiyue Zhang, Vishrav Chaudhary, Naman Goyal, James Cross, Guillaume Wenzek, Mohit Bansal, and Francisco Guzmán. 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 (Volume 1: Research Track), AMTA 2022, Orlando, USA, September 12-16, 2022*, pages 97–116. Association for Machine Translation in the Americas.

Wenxuan Zhang, Mahani Aljunied, Chang Gao, Yew Ken Chia, and Lidong Bing. 2023. M3exam: A multilingual, multimodal, multilevel benchmark for examining large language models. *Advances in Neural Information Processing Systems*, 36:5484–5505.

Vilém Zouhar, Clara Meister, Juan Gastaldi, Li Du, Mrinmaya Sachan, and Ryan Cotterell. 2023a. Tokenization and the noiseless channel. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5184–5207, Toronto, Canada. Association for Computational Linguistics.

Vilém Zouhar, Clara Meister, Juan Gastaldi, Li Du, Tim Vieira, Mrinmaya Sachan, and Ryan Cotterell. 2023b. A formal perspective on byte-pair encoding. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 598–614, Toronto, Canada. Association for Computational Linguistics.

A Pseudocode

Algorithm 1: Algorithm for learning $\mathbf{m}$ using Classical BPE.

Input: Corpus $\mathcal{D}$; number of merges K
Output: Vocabulary $\mathcal{V}_K$; merge sequence $\mathbf{m}_K$

$\mathcal{V}_0 \leftarrow \mathcal{B}$
$\mathbf{m}_0 \leftarrow \langle\,\rangle$
for $k \leftarrow 1$ **to** K **do**
  // Count all adjacent token pairs
  Pairs $\leftarrow \{\}$
  foreach *occurrence of consecutive tokens* $v\,v'$ *in* $\mathcal{D}$ *where* $v, v' \in \mathcal{V}_{k-1}$ **do**
    Pairs$[(v, v')] \leftarrow$ *Pairs*$[(v, v')] + 1$
  $(v^\star, v'^\star) \leftarrow \arg\max_{(v,v')}$ *Pairs*$[(v, v')]$
  $w^\star \leftarrow v^\star \circ v'^\star$
  // Update vocabulary and merge sequence
  $\mathcal{V}_k \leftarrow \mathcal{V}_{k-1} \cup \{w^\star\}$
  $\mathbf{m}_k \leftarrow \mathbf{m}_{k-1} + \langle (v^\star, v'^\star) \rangle$
  // Replace all occurrences in corpus
  foreach *occurrence of* $v^\star\,v'^\star$ *in* $\mathcal{D}$ **do**
    Replace $v^\star\,v'^\star$ with $w^\star$
return $\mathcal{V}_K$, $\mathbf{m}_K$

Algorithm 2: Algorithm for learning $\mathbf{m}$ using Parity-aware Byte Pair Encoding with separate training and development sets.

Input: $\{\mathcal{D}_\ell\}_{\ell\in\mathcal{L}}$ (multilingual training corpus); $\{\mathcal{D}_\ell^{\text{dev}}\}_{\ell\in\mathcal{L}}$ (multilingual development corpus); K (number of merges)
Output: $\mathcal{V}_K$ (vocabulary); $\mathbf{m}_K$ (merge list)

$\mathcal{V}_0 \leftarrow \mathcal{B}$; $\mathbf{m}_0 \leftarrow \langle\,\rangle$
for $k \leftarrow 1$ **to** K **do**
  // Calculate compression rate for each language
  foreach *language* $\ell \in \mathcal{L}$ **do**
    $\mathrm{CR}(\mathcal{D}_\ell^{\text{dev}}, \tau_{\mathbf{m}_{<k}}) \leftarrow \frac{\sum_{\mathbf{b}\in\mathcal{D}_\ell^{\text{dev}}} |\mathbf{b}|_u}{\sum_{\mathbf{b}\in\mathcal{D}_\ell^{\text{dev}}} |\tau_{\mathbf{m}_{<k}}(\mathbf{b})|}$
  $\ell^\star \leftarrow \arg\min_{\ell\in\mathcal{L}}\ \mathrm{CR}(\mathcal{D}_\ell^{\text{dev}}, \tau_{\mathbf{m}_{<k}})$
  // Consider token pairs only in $\mathcal{D}_{\ell^\star}$
  Pairs $\leftarrow \{\}$
  foreach *occurrence of consecutive tokens* $v\,v'$ *in* $\mathcal{D}_{\ell^\star}$ *where* $v, v' \in \mathcal{V}_{k-1}$ **do**
    Pairs$[(v, v')] \leftarrow$ *Pairs*$[(v, v')] + 1$
  $(v^\star, v'^\star) \leftarrow \arg\max_{(v,v')}$ *Pairs*$[(v, v')]$
  $w^\star \leftarrow v^\star \circ v'^\star$
  // Update vocabulary and merge list
  $\mathcal{V}_k \leftarrow \mathcal{V}_{k-1} \cup \{w^\star\}$
  $\mathbf{m}_k \leftarrow \mathbf{m}_{<k} + \langle (v^\star, v'^\star) \rangle$
  // Apply merge across all languages
  foreach *language* $\ell \in \mathcal{L}$ **do**
    foreach *occurrence of* $v^\star\,v'^\star$ *in* $\mathcal{D}_\ell$ *and* $\mathcal{D}_\ell^{\text{dev}}$ **do**
      Replace $v^\star\,v'^\star$ with $w^\star$
return $\mathcal{V}_K$, $\mathbf{m}_K$

B Intrinsic Tokenizer Evaluation Metrics

We provide detailed descriptions of the intrinsic tokenizer metrics used in §6, grouped by the general tokenizer characteristic the metric aims to assess. Metric formulae are defined in terms of our definition of a tokenizer $T = (\mathcal{V}, \tau, \lrcorner)$ given in §2. For this tokenizer, we denote the empirical unigram frequency distribution of tokens $v \in \mathcal{V}$ as X_T, which is computed on our evaluation corpus.

B.1 Vocabulary Usage

Vocabulary Utilization and Type-Token Ratio. Vocabulary utilization measures the proportion of a tokenizer's full vocabulary that is actively used when processing a given corpus. For tokenizer T on corpus $\mathcal{D}$, we compute it as:

$$\text{VocabUtil}(T) = \frac{|\{v : v \in \tau(\mathbf{b}), \mathbf{b} \in \mathcal{D}\}|}{|\mathcal{V}|} \quad (8)$$

Here, the numerator counts the number of distinct tokens observed across the tokenization of all strings in the corpus. The type-token ratio quantifies lexical diversity by measuring the proportion of unique tokens (types) relative to the total number of tokens produced by a tokenizer:

$$\text{TTR}(T) = \frac{|\{v : v \in \tau(\mathbf{b}), \mathbf{b} \in \mathcal{D}\}|}{\sum_{\mathbf{b}\in\mathcal{D}} |\tau(\mathbf{b})|} \quad (9)$$

where $|\tau(\mathbf{b})|$ is the number of tokens produced by tokenizer T for input $\mathbf{b}$. In words, the numerator counts distinct token types and the denominator counts total tokens across the corpus.

High vocabulary utilization and type-token ratio indicate efficient use of the learned vocabulary; low

values of these metrics for a particular language may suggest tokenizer bias, as only a small portion of the tokenizer's vocabulary is used/applicable for that language.

Average Token Rank. Average token rank (Limisiewicz et al., 2023) measures the typical position of tokens in a tokenized text within the frequency-ordered vocabulary. In more detail, we compute the rank of each token (denoted as $\text{rank}(v)$) in our unigram frequency distribution X_T; rank 1 corresponds to the most frequent token. We compute average token rank across tokens in the evaluation corpus as:

$$\text{AvgRank}(T) = \frac{\sum_{\mathbf{b}\in\mathcal{D}} \sum_{v\in\tau(\mathbf{b})} \text{rank}(v)}{\sum_{\mathbf{b}\in\mathcal{D}} |\tau(\mathbf{b})|} \quad (10)$$

This metric can be seen as another measure of the proportion of the vocabulary used by a tokenizer. Lower average ranks indicate that the tokenizer predominantly uses a small set of tokens, while higher averages suggest more diverse token usage, including rare vocabulary items. When computed per language (*i.e.*, when ranks are computed using the language's respective frequency distribution), systematic differences in average token rank across languages reveal vocabulary allocation bias.

B.2 Information-theoretic Metrics

Compression Rate. We evaluate compression rate—as defined in eq. 1—across a parallel corpus. As discussed in §4.2, this enables us to use lines (documents) as our normalization unit. Recall that higher compression rates are generally desirable for computational efficiency in downstream tasks. In multilingual corpora, compression ratio disparities across languages indicate systematic tokenizer bias, where certain languages achieve better compression efficiency than others, potentially leading to unequal computational costs.

Rényi Entropy. We compute Rényi entropy of order α over the empirical unigram frequency distribution X_T for a given tokenizer T to capture different aspects of token distribution:

$$H_\alpha(X_T) = \frac{1}{1-\alpha} \log_2 \left(\sum_{v\in\mathcal{V}} p(v)^\alpha \right) \quad (11)$$

for $0\alpha < \infty$; at $\alpha = 1$, the definition of Shannon entropy is typically adopted. Rényi entropy provides a parametric family of measures that emphasize different aspects of the distribution: H_1 (Shannon entropy), H_2 (collision entropy), and H_∞ (min-entropy). Rényi efficiency is Rényi entropy normalized by the size of the support, which is helpful for comparing tokenizers with different vocabulary sizes (Zouhar et al., 2023a). As all of our comparisons are between tokenizers of the same vocabulary size, we omit this normalization step and compare entropies directly.

B.3 Morphological and Multilingual Fairness Metrics

Fertility. Fertility measures the average number of tokens produced per unit (word, character, or byte) by a tokenizer; the unit of interest for fertility is often the *word*, in which case, fertility quantifies how many tokens (on average) a word is broken up into. We use words as our normalization unit in our computations, as determined by the HuggingFace Whitespace Pretokenizer. We formally define tokenizer fertility for a given corpus $\mathcal{D}$ as:

$$\text{Fertility}(T) = \frac{\sum_{\mathbf{b}\in\mathcal{D}} |\tau(\mathbf{b})|}{\sum_{\mathbf{b}\in\mathcal{D}} |\mathbf{b}|_u} \quad (12)$$

This metric can give a sense for the computational efficiency imbued by a tokenizer, as well as for sequence length estimates for downstream modeling tasks.

MorphScore. MorphScore (Arnett et al., 2025b) evaluates tokenizer quality through morpheme-level precision and recall, measuring how well tokenizers preserve morphological information during segmentation. We point the reader to the original work. Differences in cross-language MorphScore reveal how consistently a tokenizer's sub-token boundaries align with true morpheme boundaries. A higher score in one language than another indicates that the tokenizer preserves that language's morphological structure more faithfully. MorphScore provides a notion of both precision and recall (we point the reader to the original work for the exact description of the computation). Low precision indicates tokenizer oversegmentation; low recall is suggestive of *under* segmentation.

Gini Token Inequality Coefficient. We use an adaptation of the Gini coefficient—often used as a measure of economic inequality—to encapsulate tokenizer fairness across languages (Meister, 2025). Formally, let $c_1 \leq c_2 \leq \ldots \leq c_n$ be the "costs" under a given tokenizer T for languages

$\mathcal{L} = \{l_1, l_2, \ldots, l_n\}$. Here, we quantify cost as the average number of tokens it takes to encode the unit of interest (*e.g.*, a byte, word or line);[10] when using a parallel corpus, this can be cost per line (document), which controls for discrepancies between average character byte lengths across different scripts. The Gini coefficient for tokenizer T is then:

$$\text{Gini}(T) = \frac{1}{n}\left(n + 1 - 2\frac{\sum_{i=1}^{n}(n+1-i)c_i}{\sum_{i=1}^{n} c_i}\right) \tag{13}$$

Values range from 0 (completely equal costs across languages) to 1 (maximum inequality). This metric condenses multilingual tokenizer fairness into a single number by measuring the degree of inequality in computational costs across languages; lower Gini coefficients indicate more equitable tokenizer compression across languages, while higher values suggest systematic bias toward certain languages.

C Hyperparameter Selection

C.1 Hybrid Parity-aware BPE

For the hybrid Parity-aware tokenizer, we set the number of Parity-aware merges K equal to the number of classical BPE merges J (*i.e.*, $K = J$, with each set to either $64k$ or $128k$). This choice represents a midpoint between classical BPE ($J > 0, K = 0$) and fully Parity-aware BPE ($J = 0, K > 0$). Alternative settings allow practitioners to trade off efficiency on high-resource languages (*e.g.*, English) against cross-lingual fairness. Because no single operating point is objectively optimal across use cases, we do not claim a universally optimal choice for this trade-off.

C.2 Moving-Window Balancing

The moving-window balancing mechanism is designed to prevent degenerate cases in which a single language repeatedly consumes merge operations despite diminishing compression gains. The window size parameter ranges from 1 (enforcing uniform language selection across merges) to the number of languages N (equivalent to no constraint). In a 30-language setting, this implies that even if compression improvements for a given language stagnate, it can consume at most approximately $1/15$ of merge operations. Increasing the window size causes the behavior to increasingly resemble the unbalanced setting.

[10]This is equivalent to fertility, or the inverse of the compression rate.

Because this mechanism primarily guards against pathological merge allocation rather than directly optimizing compression, we do not expect the window size to be a sensitive hyperparameter.

To validate this intuition, we conduct ablation experiments varying the window size in the moving-window balancing variant. Table 3 reports intrinsic tokenizer metrics for window sizes ranging from 50 to 200. Average compression rate, morphological plausibility, and other intrinsic metrics remain largely stable across window sizes. Empirically, the windowing mechanism prevents repeated selection of the same language caused by development–training set mismatch, without introducing sensitivity to the specific window size. These results suggest that the moving-window approach can be applied robustly without careful tuning.

C.3 Parallel Development Set Size

We further study the effect of the size of the parallel development set used in cross-lingual compression rate comparisons for language selection (Eq. 7). Table 4 reports results for development set sizes of 100, 300, and 1000 examples.

The results indicate that relatively small development sets suffice to capture most of the fairness improvements. A parallel development set of 100 examples achieves a 67% reduction in the Gini Inequality Coefficient relative to the classical baseline, compared to a 72% reduction when using the full 1000-example development set. Global compression rates remain essentially unchanged across settings, suggesting that incorporating a parallel development set is feasible without imposing substantial data or computational burdens.

D Additional Results and Ablation Studies

The following ablation studies are all with tokenizers trained on the mC4 dataset. We ablate tokenizer vocabulary size, number of languages in the training dataset and per-language dataset balance (*i.e.*, whether dataset composition is *balanced* across languages or *unbalanced*). We also include a comparison against the UnigramLM tokenization algorithm (Kudo, 2018); we use the HuggingFace implementation of the tokenizer.

In this section, we present the results of our ablation studies. Table 6 reports the intrinsic evaluation of tokenizers with a $128k$ vocabulary size on the (*unbalanced*) *60-lang* dataset. Table 7 shows

Tokenizer	Comp. Rate (↑)	TTR (↑)	Vocab Utilization (↑)	Fertility (↓)	Rényi (α=2.5) (↑)	Gini (↓)	MorphScore Precision (↑)	MorphScore Recall (↑)
PA-BPE (W=50)	0.0277	0.0832	71.3%	4.056	**0.48**	**0.008**	0.542	0.673
PA-BPE (W=100)	**0.0278**	**0.0838**	**71.6%**	**4.042**	0.48	0.009	**0.546**	**0.677**
PA-BPE (W=150)	0.0277	0.0835	71.5%	4.049	0.48	0.009	0.543	0.674
PA-BPE (W=200)	0.0278	0.0837	71.6%	4.043	0.48	0.009	0.546	0.677

Table 3: Intrinsic tokenizer evaluation metrics for ablations over the moving-window size. Results indicate minimal sensitivity to the window size parameter.

Tokenizer	Comp. Rate (↑)	TTR (↑)	Vocab Utilization (↑)	Fertility (↓)	Rényi (α=2.5) (↑)	Gini (↓)	MorphScore Precision (↑)	MorphScore Recall (↑)
PA-BPE (N=100)	0.0277	**0.0825**	**70.8%**	3.973	0.49	0.014	0.538	0.669
PA-BPE (N=300)	0.0277	0.0823	70.7%	4.019	0.49	0.012	0.539	**0.672**
PA-BPE (N=997)	0.0276	0.0819	70.4%	4.080	**0.49**	**0.007**	0.539	0.671
PA-BPE hybrid (N=100)	**0.0278**	0.0821	70.1%	**3.967**	0.48	0.022	0.540	0.666
PA-BPE hybrid (N=300)	0.0278	0.0819	70.0%	4.015	0.49	0.019	0.540	0.667
PA-BPE hybrid (N=997)	0.0278	0.0817	69.8%	4.056	0.49	0.015	**0.541**	0.667

Table 4: Intrinsic tokenizer evaluation metrics for ablations over the size of the parallel development set. Smaller development sets retain most fairness gains. Other intrinsic metrics are very stable across different choices.

the corresponding results for the (*balanced*) *30-lang* dataset, also with a $128k$ vocabulary size. Finally, Table 8 presents the intrinsic evaluation of tokenizers with a $256k$ vocabulary size on the (*unbalanced*) *30-lang* dataset. Together, these results demonstrate the effectiveness of Parity-aware BPE across different language settings, vocabulary sizes, and data distributions.

Training Data Distribution. To assess the sensitivity of the Parity-aware algorithm to training data distribution, we also analyze results for $128k$ tokenizers trained on the *balanced* version of the dataset. The results in Table 7 indicate that Parity-aware BPE tokenizers perform similarly to Classical BPE across most metrics. However, in terms of fertility, Classical BPE outperforms the Parity-aware variants. This suggests that Parity-aware tokenizers are particularly beneficial in unbalanced settings, where low-resource languages are more disadvantaged, whereas in balanced scenarios their advantage diminishes. We also interestingly see in Fig. 4—again for the (*balanced*) *30-lang* setting—that Parity-aware tokenizers yield the largest absolute increases in vocabulary utilization for high-resource languages. Low- and medium-resource languages also improve, though to a smaller extent. One logical conclusion from this result is that the effect of Parity-aware tokenizer is better described as balancing utilization across languages rather than directly compensating for data scarcity.

Language Model Perplexities. We report language model perplexities on the FineWeb2 validation set in Fig. 5. Results are shown per language. We see a noticeably larger cross-lingual spread in perplexity for language models trained using the Classical BPE tokenizer than for those trained using Parity-aware variants. The Parity-aware tokenizers seem to eliminate the long tail present under Classical BPE while maintaining comparable mean perplexity across languages. Note that we normalize by number of bytes in the text rather than by number of tokens to account for differences in tokenization lengths.

E Balancing the tokenizer training set

Prior work has addressed representational imbalance in subword tokenization primarily through corpus-level reweighting or balancing strategies, such as equalizing the number of documents or bytes per language prior to training a BPE or unigram tokenizer. While such approaches ensure that low-resource languages contribute to the learned vocabulary, they optimize the tokenizer with respect to an artificial distribution that differs from the natural data distribution encountered during downstream training and inference. This mismatch can result in inefficient allocation of vocabulary capacity, over-representation of rare patterns, and increased tokenization length for high-frequency languages. Moreover, corpus balancing constitutes a coarse-grained intervention: the choice of balanc-

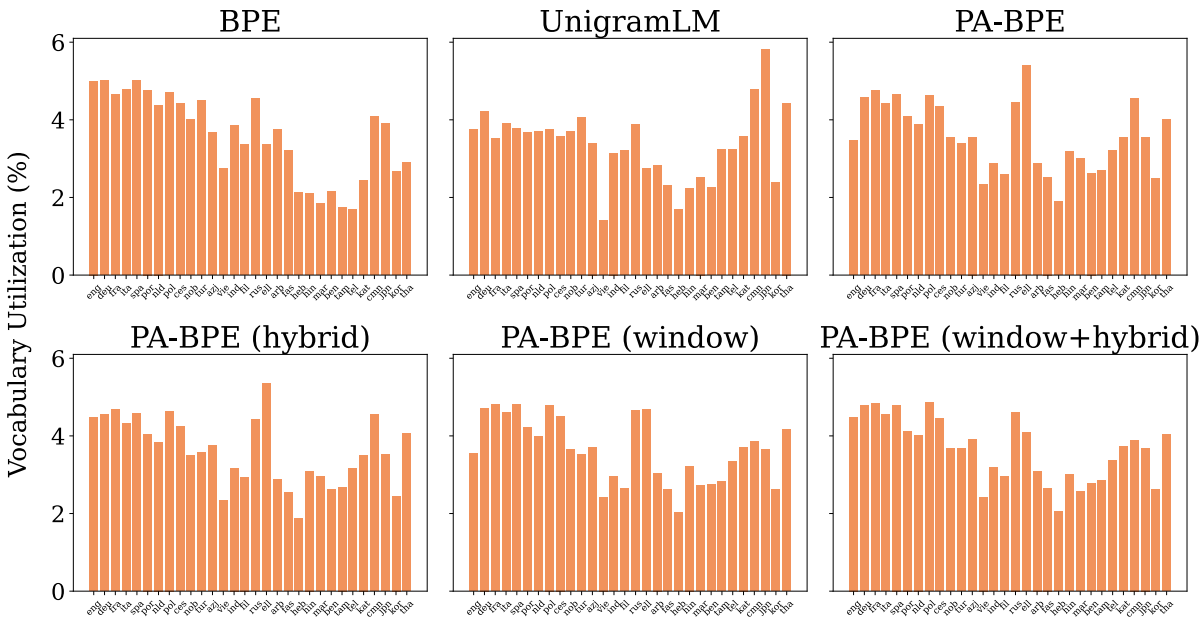

Figure 3: Vocabulary utilization of $128k$ tokenizers on the (*unbalanced*) *30-lang* per language. Dashed lines indicate the global average.

Tokenizer	Comp. Rate (↑)	TTR (↑)	Vocab Utilization (↑)	Fertility (↓)	Rényi (α=2.5) (↑)	Gini (↓)	MorphScore Precision (↑)	MorphScore Recall (↑)
BPE	0.0262	0.0718	65.1%	4.178 ± 0.048	**0.50**	0.067	**0.434 ± 0.045**	**0.527 ± 0.042**
UnigramLM	0.0193	0.0457	56.3%	4.536 ± 0.041	0.29	0.095	0.147 ± 0.035	0.262 ± 0.049
PA-BPE	0.0260	0.0727	66.4%	4.138 ± 0.047	0.50	**0.027**	0.418 ± 0.048	0.508 ± 0.044
PA-BPE (hybrid)	0.0264	0.0737	66.3%	**4.116 ± 0.048**	0.50	0.027	0.426 ± 0.047	0.515 ± 0.044
PA-BPE (window)	0.0262	0.0749	67.9%	4.152 ± 0.048	0.50	0.029	0.417 ± 0.046	0.507 ± 0.042
PA-BPE (window+hybrid)	**0.0266**	**0.0761**	**67.9%**	4.127 ± 0.048	0.49	0.031	0.430 ± 0.045	0.517 ± 0.043

Table 5: Intrinsic evaluation of $128k$ tokenizers on the (*unbalanced*) ***30-lang*** mC4 dataset. Values are global statistics, except for MorphScore, which is macro-averaged across available languages.

Tokenizer	Comp. Rate (↑)	TTR (↑)	Vocab Utilization (↑)	Fertility (↓)	Rényi (α=2.5) (↑)	Gini (↓)	MorphScore Precision (↑)	MorphScore Recall (↑)
BPE	0.0222	**0.0341**	**72.8%**	**3.564 ± 0.026**	0.51	0.144	**0.335 ± 0.029**	**0.481 ± 0.027**
PA-BPE	0.0212	0.0264	58.9%	3.707 ± 0.027	**0.52**	**0.083**	0.288 ± 0.027	0.466 ± 0.026
PA-BPE (hybrid)	0.0219	0.0279	60.4%	3.624 ± 0.027	0.52	0.100	0.293 ± 0.027	0.450 ± 0.027
PA-BPE (window)	0.0218	0.0321	70.1%	3.647 ± 0.027	0.51	0.101	0.314 ± 0.027	0.469 ± 0.026
PA-BPE (window+hybrid)	**0.0222**	0.0338	72.1%	3.576 ± 0.027	0.50	0.116	0.323 ± 0.027	0.466 ± 0.027

Table 6: Intrinsic evaluation of $128k$ tokenizers on the (*unbalanced*) ***60-lang*** mC4 dataset. Values are global statistics, except for MorphScore, which is macro-averaged across available languages.

Tokenizer	Comp. Rate (↑)	TTR (↑)	Vocab Utilization (↑)	Fertility (↓)	Rényi (α=2.5) (↑)	Gini (↓)	MorphScore Precision (↑)	MorphScore Recall (↑)
BPE	**0.0267**	0.0758	67.4%	**4.092**	0.49	0.051	**0.429**	0.513
PA-BPE	0.0259	0.0726	66.4%	4.141	**0.50**	0.027	0.416	0.508
PA-BPE (hybrid)	0.0264	0.0738	66.3%	4.113	0.50	**0.023**	0.420	0.508
PA-BPE (window)	0.0261	0.0748	67.9%	4.155	0.50	0.029	0.418	0.507
PA-BPE (window+hybrid)	0.0266	**0.0764**	**68.1%**	4.098	0.49	0.027	0.429	**0.513**

Table 7: Intrinsic evaluation of $128k$ tokenizers on the (*balanced*) ***30-lang*** mC4 dataset. Values are global statistics, except for MorphScore, which is macro-averaged across available languages.

Tokenizer	Comp. Rate (↑)	TTR (↑)	Vocab Utilization (↑)	Fertility (↓)	Rényi (α=2.5) (↑)	Gini (↓)	MorphScore Precision (↑)	MorphScore Recall (↑)
BPE	0.0294	0.1200	48.3%	**3.696** ± **0.042**	0.46	0.056	0.529 ± 0.047	**0.598** ± **0.044**
PA-BPE	0.0288	0.1143	47.0%	3.752 ± 0.043	**0.47**	0.028	0.501 ± 0.050	0.570 ± 0.046
PA-BPE (hybrid)	0.0291	0.1157	47.1%	3.739 ± 0.043	0.47	**0.026**	0.513 ± 0.049	0.582 ± 0.046
PA-BPE (window)	0.0292	0.1203	48.9%	3.725 ± 0.043	0.47	0.032	0.514 ± 0.049	0.580 ± 0.045
PA-BPE (window+hybrid)	0.0294	0.1217	49.0%	3.709 ± 0.043	0.46	0.030	0.524 ± 0.048	0.590 ± 0.045
PA-BPE (hybrid-127k)	0.0294	0.1188	47.9%	3.722 ± 0.043	0.46	0.027	0.521 ± 0.049	0.590 ± 0.045
PA-BPE (window+hybrid-127k)	**0.0296**	**0.1231**	**49.3%**	3.698 ± 0.043	0.46	0.032	**0.530** ± **0.048**	0.596 ± 0.045

Table 8: Intrinsic evaluation of $256k$ tokenizers on the (*unbalanced*) ***30-lang*** mC4 dataset. Values are global statistics, except for MorphScore, which is macro-averaged across available languages.

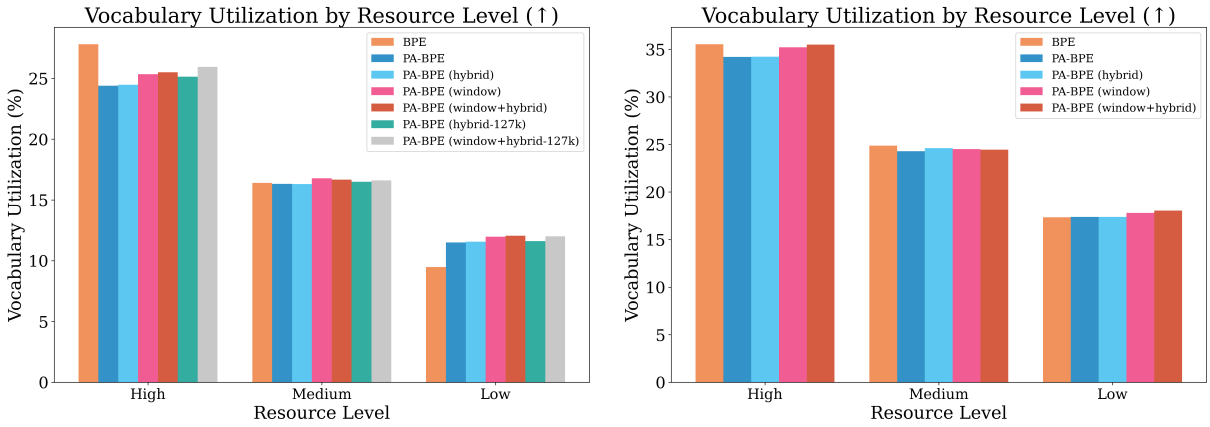

Figure 4: Vocabulary utilization grouped by language resource levels for the $256k$ tokenizer trained on the (*unbalanced*) *30-lang* mC4 dataset (left) and $128k$ tokenizer trained on the (*balanced*) mC4 *30-lang* dataset (right).

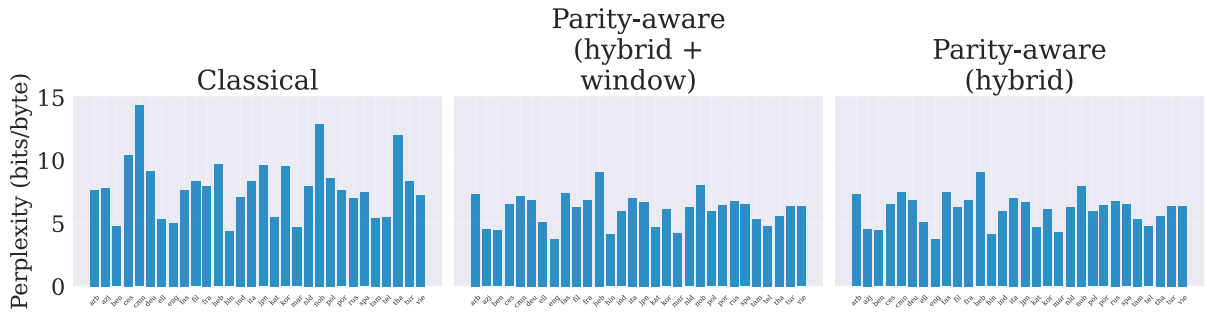

Figure 5: Per-language perplexities (↓) normalized by byte of language models trained using the specified tokenizer trained on the (*unbalanced*) $128k$ *30-lang* dataset. Results are computed on the language model validation set at the final checkpoint (see §F for language model details).

ing criterion substantially affects outcomes, and the approach does not account for heterogeneity within groups or for the local utility of individual merge operations.

Parity-aware BPE instead operates on the unaltered corpus distribution and introduces language-level constraints directly into the merge selection process. By augmenting merge scores with Parity-aware penalties or budgets, this approach enables fine-grained control over how subword units are allocated across languages, while preserving the efficiency benefits of frequency-driven merging. In contrast to the balancing approach, Parity-aware merges maintain alignment with the downstream data distribution and typically yield lower average tokenization costs at inference, while mitigating systematic over-fragmentation of underrepresented languages. Overall, Parity-aware BPE offers a principled mechanism for addressing tokenizer bias at the level of individual merge decisions, complementing existing corpus-level mitigation strategies.

F Language Model Training

Here we provide details about the language models used for evaluating extrinsic tokenizer metrics.

F.1 Model Architecture

Our experiments focus on models with 3 billion parameters (3B), following the LLaMA architecture (Touvron et al., 2023). Model size is determined by adjusting the number of layers, hidden dimensions, and attention heads.

F.2 Training Hyperparameters

We train our models using SwissAI's fork of Hugging Face's Nanotron repository.[11] The key training hyperparameters are as follows:

- **Learning Rate.** We use a learning rate of 8×10^{-4} with linear warmup over the first 4% of training. A "1-sqrt" decay schedule (Hägele et al., 2024) is applied during the final 20%.
- **Optimizer.** All experiments use AdamW with $\beta = (0.9, 0.95)$ (Loshchilov and Hutter, 2019).
- **Weight Decay.** We set the weight decay parameter to $\lambda = 0.1$ for regularization.
- **Batch Size.** The micro-batch size is fixed at 5 across all runs.

[11]The codebase: https://github.com/swiss-ai/nanotron-multilingual

F.3 Hardware Setup

Training is performed on a large-scale cluster. Each node is equipped with 4 NVIDIA Grace-Hopper H100 GPUs (96 GB memory each). The 3B models are trained on 64 nodes (256 GPUs in total), requiring approximately 18 hours per 100B tokens. This corresponds to a global batch size of 640 examples.

F.4 Sampling Methods

Let $\mathcal{L}$ be the set of languages in the dataset, and let $\pi^{\text{natural}} \in \Delta_{|\mathcal{L}|}$ represent the natural distribution of these languages, defined as:

$$\pi_l^{\text{natural}} = \frac{\omega_l}{\sum_{l' \in \mathcal{L}} \omega_{l'}}$$

where ω_l denotes the number of words (or tokens) for language l in the dataset. In this work, we use the number of words as a proxy for language frequency, a common practice when presenting statistics for highly multilingual datasets (Penedo et al., 2025). In order to create our LM training dataset, we use temperature sampling. This method adjusts the natural distribution using a temperature parameter τ to create a less skewed distribution:

$$\pi_l^{\text{temp},\tau} = \frac{\omega_l^{1/\tau}}{\sum_{l' \in \mathcal{L}} \omega_{l'}^{1/\tau}}$$

By tuning τ, the distribution can be shifted towards uniformity, thereby reducing imbalance among languages.

G Downstream Benchmark Evaluation

We evaluate our models using HuggingFace's Lighteval codebase (Habib et al., 2023).

G.1 Benchmarks

To assess multilingual performance, we select twelve widely used benchmarks that span diverse downstream tasks, including reading comprehension, commonsense reasoning, semantic similarity, and knowledge-based evaluation (Üstün et al., 2024; Apertus Project et al., 2025; Martins et al., 2025). At the same time, we ensured that the chosen benchmarks provide meaningful performance signals (results clearly above random chance) for small-scale models (1-3B parameters). Additionally, all selected benchmarks can be evaluated in a zero-shot setting and do not require supervised finetuning. To ensure consistency with prior work, we followed established evaluation practices.

- **Belebele**:[12] A multilingual reading comprehension dataset containing passages and corresponding questions in many languages. It evaluates models' ability to understand text and answer related questions (Bandarkar et al., 2024).

- **mTruthfulQA**:[13] A multilingual extension of TruthfulQA, designed to measure whether models generate accurate and non-misleading answers across a broad range of questions (Lin et al., 2022a; Lai et al., 2023).

- **PAWS-X**:[14] A multilingual paraphrase identification benchmark that extends the original PAWS dataset, providing sentence pairs annotated for semantic equivalence (Yang et al., 2019).

- **XCodah**:[15] A multilingual adaptation of CODAH for adversarially-authored commonsense reasoning tasks, testing robustness in natural language understanding (Lin et al., 2021; Chen et al., 2019).

- **XCSQA**:[16] A multilingual version of CommonsenseQA, consisting of multiple-choice questions that require reasoning about everyday concepts and their relations (Lin et al., 2021; Talmor et al., 2019).

- **XNLI**:[17] A cross-lingual natural language inference benchmark, evaluating whether models can perform entailment, contradiction, and neutrality classification across multiple languages (Conneau et al., 2018).

- **XStoryCloze**:[18] A multilingual extension of the StoryCloze Test, where models must choose the most coherent ending to short narratives, testing story comprehension and commonsense reasoning (Mostafazadeh et al., 2017; Lin et al., 2022b).

- **XWinogrande**:[19] A multilingual version of WinoGrande, containing sentences with ambiguous pronouns. It measures models' ability to resolve coreference using contextual and commonsense cues (Sakaguchi et al., 2021; Muennighoff et al., 2023; Tikhonov and Ryabinin, 2021).

- **MMMLU**:[20] A multilingual adaptation of MMLU, evaluating model performance across a wide spectrum of tasks and domains (Hendrycks et al., 2021; Lai et al., 2023).

- **INCLUDE**:[21] A large-scale benchmark covering 44 languages, designed to evaluate multilingual LLMs in realistic language environments with a focus on knowledge and reasoning (Romanou et al., 2025).

- **Exams**:[22] A benchmark of standardized test questions across subjects and educational levels, used to assess reasoning and problem-solving abilities in exam-like conditions (Hardalov et al., 2020).

- **M3Exams**:[23] A multilingual exam-style benchmark that extends Exams across different languages, subjects, and difficulty levels (Zhang et al., 2023).

G.2 Score Aggregations

We aggregate benchmark results to compute a language-specific score for each model. Let $\mathcal{T}_l$ be the set of benchmarks (or tasks) containing a split for language l. The aggregated score for a model m per language l is defined as:

$$s_l^m = \frac{1}{|\mathcal{T}_l|} \sum_{t \in \mathcal{T}_l} s_{t,l}^m$$

where s_l^m is the score of a model m on the split l of a task t To mitigate biases arising from varying numbers of benchmarks per language, we compute a language-specific random baseline ζ_l. This baseline helps assess whether a given aggregated

[12] https://huggingface.co/datasets/facebook/belebele

[13] https://huggingface.co/datasets/alexandrainst/m_truthfulqa

[14] https://huggingface.co/datasets/google-research-datasets/paws-x

[15] https://huggingface.co/datasets/INK-USC/xcsr

[16] https://huggingface.co/datasets/INK-USC/xcsr

[17] https://huggingface.co/datasets/facebook/xnli

[18] https://huggingface.co/datasets/juletxara/xstory_cloze

[19] https://huggingface.co/datasets/allenai/winogrande

[20] https://huggingface.co/datasets/openai/MMMLU

[21] https://huggingface.co/datasets/CohereLabs/include-base-44

[22] https://huggingface.co/datasets/mhardalov/exams

[23] https://huggingface.co/datasets/SEACrowd/m3exam

score significantly outperforms random predictions. Specifically, we calculate the random baseline for each language as the average of the individual random baselines across all tasks that include language l:

$$\zeta_l = \frac{1}{|\mathcal{T}_l|} \sum_{t \in \mathcal{T}_l} \zeta_t$$

Language	Language Family	Script	Resource Level	30-lang	60-lang
English	Indo-European (Germanic)	Latin	High	✓	✓
German	Indo-European (Germanic)	Latin	High	✓	✓
French	Indo-European (Romance)	Latin	High	✓	✓
Italian	Indo-European (Romance)	Latin	High	✓	✓
Russian	Indo-European (Slavic)	Cyrillic	High	✓	✓
Spanish	Indo-European (Romance)	Latin	High	✓	✓
Japanese	Japonic	Kanji & Kana (CJK)	Medium	✓	✓
Polish	Indo-European (Slavic)	Latin	Medium	✓	✓
Portuguese	Indo-European (Romance)	Latin	Medium	✓	✓
Vietnamese	Austroasiatic	Latin	Medium	✓	✓
Turkish	Turkic	Latin	Medium	✓	✓
Dutch	Indo-European (Germanic)	Latin	High	✓	✓
Indonesian	Austronesian	Latin	Medium	✓	✓
Arabic	Afro-Asiatic (Semitic)	Perso-Arabic	Medium	✓	✓
Czech	Indo-European (Slavic)	Latin	Medium	✓	✓
Persian (Farsi)	Indo-European (Iranian)	Perso-Arabic	Medium	✓	✓
Greek	Indo-European (Hellenic)	Greek	Medium	✓	✓
Chinese (Mandarin)	Sino-Tibetan	Hanzi (CJK)	Medium	✓	✓
Hindi	Indo-European (Indo-Aryan)	Devanagari (Brahmic)	Medium	✓	✓
Korean	Koreanic	Hangugeo (CJK)	Medium	✓	✓
Thai	Kra–Dai (Tai)	Thai	Medium	✓	✓
Hebrew	Afro-Asiatic (Semitic)	Hebrew	Medium	✓	✓
Bengali	Indo-European (Indo-Aryan)	Bengali (Brahmic)	Medium	✓	✓
Tamil	Dravidian (Brahmic)	Tamil	Low	✓	✓
Georgian	Kartvelian	Georgian	Low	✓	✓
Marathi	Indo-European (Indo-Aryan)	Devanagari (Brahmic)	Medium	✓	✓
Filipino	Austronesian	Latin	Low	✓	✓
Telugu	Dravidian	Telugu (Brahmic)	Low	✓	✓
Norwegian	Indo-European (Germanic)	Latin	Medium	✓	✓
North Azerbaijani	Turkic	Latin	Low	✓	✓
Swedish	Indo-European (Germanic)	Latin	Medium	-	✓
Romanian	Indo-European (Romance)	Latin	Medium	-	✓
Ukrainian	Indo-European (Slavic)	Cyrillic	Medium	-	✓
Hungarian	Uralic (Ugric)	Latin	Medium	-	✓
Danish	Indo-European (Germanic)	Latin	Medium	-	✓
Finnish	Uralic (Finnic)	Latin	Medium	-	✓
Bulgarian	Indo-European (Slavic)	Cyrillic	Low	-	✓
Slovak	Indo-European (Slavic)	Latin	Low	-	✓
Catalan	Indo-European (Romance)	Latin	Low	-	✓
Malay	Austronesian	Latin	Low	-	✓
Urdu	Indo-European (Indo-Aryan)	Perso-Arabic	Low	-	✓
Belarusian	Indo-European (Slavic)	Cyrillic	Medium	-	✓
Basque	Language Isolate	Latin	Low	-	✓
Tajik	Indo-European (Iranian)	Cyrillic	Medium	-	✓
Sotho (Sesotho)	Niger–Congo (Bantu)	Latin	Low	-	✓
Yoruba	Niger–Congo	Latin	Low	-	✓
Swahili	Niger-Congo (Bantu)	Latin	Low	-	✓
Estonian	Uralic (Finnic)	Latin	Low	-	✓
Latvian	Indo-European (Slavic)	Latin	Low	-	✓
Galician	Indo-European (Romance)	Latin	Low	-	✓
Welsh	Indo-European (Celtic)	Latin	Low	-	✓
Albanian	Indo-European	Latin	Low	-	✓
Macedonian	Indo-European (Slavic)	Cyrillic	Low	-	✓
Malayalam	Dravidian	Malayalam (Brahmic)	Low	-	✓
Burmese	Sino-Tibetan	Mon–Burmese	Low	-	✓
Gujarati	Indo-European (Indo-Aryan)	Gujarati (Brahmic)	Low	-	✓
Afrikaans	Indo-European (Germanic)	Latin	Low	-	✓
Hawaiian	Austronesian	Latin	Low	-	✓
Northern Uzbek	Turkic	Latin	Low	-	✓

Table 9: Details on the languages used to train and evaluate tokenizers.

Language	INCLUDE	Belebele	Exams	M3Exam	MMMLU	mTruthfulQA	PAWS-X	XCodah	XCSQA	XNLI	XStoryCloze	XWinoGrande
English	-	✓	-	✓	✓	✓	✓	✓	✓	✓	✓	✓
Chinese	✓	✓	-	✓	✓	✓	✓	✓	✓	✓	✓	✓
Vietnamese	✓	✓	✓	✓	✓	✓	-	✓	✓	✓	-	-
Arabic	✓	✓	✓	-	✓	✓	-	✓	✓	✓	✓	-
German	✓	✓	✓	-	✓	✓	✓	✓	✓	-	-	-
Spanish	✓	✓	✓	-	✓	✓	-	✓	✓	✓	✓	-
French	✓	✓	-	-	✓	✓	✓	✓	✓	✓	-	✓
Portuguese	✓	✓	✓	-	✓	✓	-	✓	✓	-	-	✓
Hindi	✓	✓	-	-	✓	✓	-	✓	✓	✓	✓	-
Russian	✓	✓	-	-	✓	✓	-	✓	✓	✓	✓	✓
Indonesian	✓	-	-	-	✓	✓	-	-	-	-	✓	-
Italian	✓	✓	✓	✓	✓	✓	-	✓	✓	-	-	-
Japanese	✓	-	-	-	-	-	✓	✓	✓	-	-	✓
Swahili	-	✓	-	✓	-	-	-	✓	✓	✓	✓	-
Tamil	✓	-	-	-	-	-	-	-	-	-	-	-
Telugu	✓	✓	-	-	✓	✓	-	-	-	-	✓	-
Thai	-	✓	-	✓	-	-	-	-	-	✓	-	-
Basque	✓	-	-	-	✓	✓	-	-	-	-	✓	-
Turkish	✓	✓	✓	-	-	-	-	-	-	✓	-	-
Bulgarian	✓	✓	✓	-	-	-	-	-	-	✓	-	-
Albanian	✓	✓	✓	-	-	-	-	-	-	-	-	-
Polish	✓	✓	-	-	-	-	-	✓	-	-	-	-
Bengali	✓	-	-	-	✓	✓	-	-	-	-	-	-
Serbian	✓	-	✓	-	-	✓	-	-	-	-	-	-
Estonian	✓	-	-	-	-	-	-	-	-	-	-	-
Macedonian	✓	✓	-	-	-	-	-	-	-	-	-	-
Lithuanian	✓	-	✓	-	-	-	-	-	-	-	-	-
Greek	✓	-	-	-	-	-	-	-	-	✓	-	-
Urdu	✓	-	-	-	-	-	-	-	-	✓	-	-
Catalan	-	-	-	-	✓	✓	-	-	-	-	-	-
Persian	✓	-	-	-	-	-	-	-	-	-	-	-
Finnish	✓	-	-	-	-	-	-	-	-	-	-	-
Korean	✓	-	-	-	-	-	-	-	-	-	-	-
Quechua	-	-	-	-	-	-	-	-	-	-	-	-
Haitian Creole	-	-	-	-	-	-	-	-	-	-	-	-
Malay	-	-	-	-	-	-	-	-	-	-	✓	-

Table 10: Coverage of downstream benchmarks across languages.