AdaptBPE: From General Purpose to Specialized Tokenizers

Vijini Liyanage and François Yvon
Sorbonne-Université, CNRS, ISIR, Paris
{pilanaliyanage,yvon}@isir.upmc.fr

Abstract

Subword tokenization methods, such as Byte-Pair Encoding (BPE), significantly impact the performance and efficiency of large language models (LLMs). The standard approach involves training a general-purpose tokenizer that uniformly processes all textual data during both training and inference. However, the use of a generic set of tokens can incur inefficiencies when applying the model to specific domains or languages. To address this limitation, we propose a post-training adaptation strategy that selectively replaces low-utility tokens with more relevant ones based on their frequency in an adaptation corpus. Our algorithm identifies the token inventory that most effectively encodes the adaptation corpus for a given target vocabulary size. Extensive experiments on generation and classification tasks across multiple languages demonstrate that our adapted tokenizers compress test corpora more effectively than baselines using the same vocabulary size. This method serves as a lightweight adaptation mechanism, akin to a vocabulary fine-tuning process, enabling optimized tokenization for specific domains or tasks. Our code and data are available at https://github.com/vijini/Adapt-BPE.git.

1 Introduction

Subword tokenization has become a standard component in modern large language models (LLMs) such as GPT, BLOOM, Llama, Gemma and Qwen (Brown et al., 2020; BigScience et al., 2022; Touvron et al., 2023; Team et al., 2024; Bai et al., 2023). Techniques like Byte-Pair Encoding (BPE) (Gage, 1994; Sennrich et al., 2016) and SentecePiece (Kudo, 2018) represent rare and unknown words by decomposing them into frequently occurring subwords. These methods strike a balance between vocabulary size and coverage, enabling models to generalize across diverse inputs.

Despite their widespread adoption, most tokenizers operate statically. This means that once a BPE tokenizer is trained, its merge rules remain fixed throughout the model's lifecycle. This static nature can lead to suboptimal performance when the distribution of the test corpus differs from the training distribution (Ovadia et al., 2019). In fact, learning a BPE vocabulary is no different from other learning algorithms and is prone to overfitting, hence the need include regularization mechanisms (Provilkov et al., 2020) or to adapt to the test conditions (Sachidananda et al., 2021). Domain-specific corpora may benefit from different tokenization strategies that better capture local frequency statistics and require supplementary tokens (Liu et al., 2023, 2024). Likewise, adapting a large multilingual model to a restricted set of languages may cause many tokens (and their associated parameters) to become useless. Adapting a tokenizer offers two potential benefits: (a) it reduces the number of units and associated model parameters that need to be managed; (b) it results in shorter tokenized texts, thereby reducing inference costs.

In this paper, we propose a lightweight and practical approach to post-hoc tokenizer adaptation. Operating under a fixed merge budget, our algorithm optimizes tokenization by replacing low-utility merges with alternatives that offer greater compression benefits for test domain(s) and language(s). This strategy, similar to model fine-tuning but operating on vocabularies, computes non-canonical tokenizations that can readily be used *without updating the model weights*. In a nutshell, our algorithm starts by applying the first N merges of a pretrained BPE tokenizer. It then iteratively refines the vocabulary by replacing low-frequency tokens with high-frequency alternatives identified on a development set. This process *improves compression utility*, defined as the reduction in token count, while remaining fully compatible with the associated LLM.

Unlike prior work that trains augmented domain-specific tokenizers or exploits dynamic tokeniza-

Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics
Volume 1: Long Papers, pages 2607–2620
March 24-29, 2026 ©2026 Association for Computational Linguistics

tion at runtime (Gee et al., 2022, 2023; Da Dalt et al., 2024; Geng et al., 2025), our approach emphasizes a cost-benefit analysis that is particularly relevant for multilingual LLMs. These models often support vocabularies spanning thousands of tokens to accommodate as many scripts and languages as possible. However, large coverage introduces inefficiencies when processing language-specific or domain-specific corpora. By selectively refining the vocabulary, our method vastly reduces the number of active tokens, with minimal impact on performance. This reduction can yield smaller parameter sets, shorter tokenized sequences and better inference times. As it does not require any change in the model weights, it offers a practical, plug-and-play solution for downstream deployment—especially in constrained or high-throughput environments.

We showcase our approach across test corpora for several LLMs, languages, domains and tasks and demonstrate its superiority over baselines operating with the same vocabulary size. Our contributions can be summarized as follows (a) we introduce a refinement algorithm that operates on BPE merge lists and improves compression utility for test corpora under a fixed merge budget; (b) we run experiments showing the effectiveness of this method in multiple monolingual and bilingual settings.

2 Related Work

Subword Tokenization. Subword tokenization is foundational in modern NLP. BPE (Gage, 1994; Sennrich et al., 2016) is widely used in contemporary LLMs: starting with a character (or byte-based) segmention, it learns a finite set of subword units by recursively merging frequent bigrams into new units, balancing vocabulary compactness and generalization. Alternatives include Unigram LM (Kudo, 2018) and WordPiece (Wu et al., 2016).

BPE Variants. Extensions of BPE improve generalization or efficiency by amending the token set during learning, post-learning, or inference:

- **BPE-Dropout** (Provilkov et al., 2020) samples merge paths stochastically during training to improve robustness; (Zheng et al., 2025) samples random, *non-canonical* tokenizations, during inference, and observes a moderate loss of performance for fine-tuned models;
- **BPE Trimming** (Cognetta et al., 2024), **Picky BPE** (Chizhov et al., 2024) and **Scaffold-BPE** (Lian et al., 2025) attempt to solve a defect of the BPE algorithm, which tends to produce intermediate tokens ("junk" tokens) (Bostrom and Durrett, 2020; Li et al., 2024) that end up with a low frequency on the training corpus. In these variants, low-frequency tokens are removed ("trimmed") and also possibly replaced by larger units, based on bigram counts computed on the training corpus;
- Another well-known defect of BPE is that it generates tokens that inconsistent with morphologically motivated segmentations (Vania and Lopez, 2017; Hou et al., 2023; Mager et al., 2022); **BPE Knockout** (Bauwens and Delobelle, 2024) revises post-hoc the set of merge operations so as to improve the compatibility of tokens with linguistic morphemes.

Our approach also aims to replace low-utility tokens in the pretrained vocabulary; our goal is however different as we develop plug-and-play tokenization adapters that are optimized post-hoc for specific domains and languages; furthermore, it relies on the sole statistics of the adaptation corpus and does not require access to the pre-training data.

Vocabulary Adaptation. Multilingual tokenizer adaptation (Sachidananda et al., 2021; Feng et al., 2024) has primarily been studied in a scenario where new languages need to be added to an existing language model. This typically implies adding new units, to cover more scripts (Lin et al., 2024; Imani et al., 2023; Lu et al., 2024), then learn the corresponding embeddings via continued pretraining. In this scenario, finding the right initialization for these new parameters improves their estimation, especially in when the adaptation data is scarce (Minixhofer et al., 2022; Dobler and de Melo, 2023; Remy et al., 2024; Singh et al., 2025). By contrast, we consider a scenario where one want to *reduce* the language coverage of a pre-trained, so as to adapt to a restricted set of texts.

Vocabulary Refinement and Efficiency. Our approach reverses the scaling trend by identifying an effective subset of merges under a fixed budget. Unlike prior work (Sachidananda et al., 2021; Ushio et al., 2023; Lian et al., 2025; Cognetta et al., 2024), we do not require training data, retraining, or vocabulary heuristics. We operate directly on standard `tokenizer.json` files, using corpus-level statistics on adaptation data. Compared to (Bogoychev et al., 2024), which also uses heuristic vocabulary trimming in machine translation, our approach is more sound, as well as more effective.

Inference-Time Benefits. Test-time refinement yields practical efficiency: unused tokens can be pruned from the output projection matrix, reducing memory and compute costs. It also reduces off-target tokenization, improving compression and lowering the number of decoding steps—especially useful in domain-specific or low-latency settings.

3 Methodology

3.1 BPE Tokenizers Adaptation

Byte Pair Encoding (BPE) tokenizers are trained on very large corpora prior to model training. They provide an effective way to represent any running text as a sequence of tokens from a finite vocabulary, that is then used to estimate the model parameters. As they are associated with the model parameterization, tokenizers are usually kept fixed during all subsequent steps of the model lifecycle (supervised fine-tuning, adaptation, alignment, exploitation, etc). When specializing an LLM to a restricted number of languages or domains, though, the tokenization procedure may deliver suboptimal segmentations; it may also cause to load a lot of useless model parameters in memory. Our method addresses these shortcomings by revising *post hoc* the set of BPE merges on an adaptation corpus.

We assume access to a BPE tokenizer (e.g., Llama tokenizer), from which we extract the full list of M training merges. Given a target vocabulary size of N, we initially apply the first N merges and compute the resulting token frequencies on the adaptation corpus. We then iteratively revise this merge set by replacing low-utility merges with more frequent and beneficial merges from the remaining set of merges, aiming to reduce the overall length of the adaptation corpus in a greedy fashion.

This process is applied during a post-training step: it does not require to add new tokens to the vocabulary, nor does it require access to the original training data either. As it delivers a merge list that is fully compatible with the associated LLM, it can be used without any change in the model parameter. This procedure is formalized in section 3.3.

3.2 Concepts

Before presenting our method, we recall the main concepts of BPE-based tokenizers (Zouhar et al., 2023; Berglund and van der Merwe, 2023) and introduce useful notations. We denote Σ a base set of symbols and Σ^* its Kleene closure. Learning a BPE tokenizer on some training corpus $\mathcal{C}$ yields an ordered list of merges $\boldsymbol{\mu}(\mathcal{C}) = [\mu_0, \mu_1, \ldots, \mu_M]$, with $\mu_i = (x_i, y_i) \in \Sigma^* \times \Sigma^*$; x_i and y_i will are the *parent tokens* of μ_i. We use Python notations to denote subsequences (e.g. $\boldsymbol{\mu}[: i]$ for $[\mu_0, \ldots \mu_i]$). The tokenization process of text $\mathcal{W} \in \Sigma^*$ first[1]splits the text in its base symbols, then recursively applies merging rules in $\boldsymbol{\mu}(\mathcal{C})$ from μ_0 to μ_M, where applying the rule $\mu_i = (x_i, y_i)$ replaces all occurrences of bigram $(x_i y_i)$ with a new symbol μ_i. We denote $\text{apply}(\mu, \mathcal{W})$ the text resulting from applying rule μ to $\mathcal{W}$. Texts tokenized with BPE rules $\boldsymbol{\mu}$ are sequences of tokens in $(\Sigma \cup \Gamma)^*$, where $\Gamma = \text{set}(\boldsymbol{\mu})$.

A key property of a learned BPE vocabulary $\boldsymbol{\mu}(\mathcal{C})$ is *properness* (Berglund and van der Merwe, 2023): a merge sequence is *proper* if every compound token is only created after its parents have been created. Formally, a merge sequence $\mu = [\mu_0, \mu_1, \ldots, \mu_N]$ is *proper* if for all $\mu_i = (x_i, y_i)$: (a) $x_i \in \Sigma$ or $\exists j < i | x_i = \mu_j$; (b) $y_k \in \Sigma$ or $\exists k < i | y_i = \mu_k$. Note that any prefix of a proper merge sequence remains proper.

Following Lian et al. (2025), we extend these notions by distinguishing between *actual* and *virtual* (or scaffold) merges, with the following semantic: actual merges correspond to tokens that will appear in the tokenized text; virtual merges only exist to create larger units, and do not appear in the tokenized text. Given a list of merges $\boldsymbol{\mu}$ and their associated type, tokenization proceeds as follows: (a) merges in $\boldsymbol{\mu}$ are applied from first to last; (b) $\boldsymbol{\mu}$ is then processed *backwards*[2] and virtual tokens are replaced with their parents – a procedure denoted $\text{unapply}(\mu, \mathcal{W})$ below.

3.3 The AdaptBPE Algorithm

AdaptBPE is formalized in Algorithm 1. It maintains the following invariant: at any stage, $\boldsymbol{\mu}_\text{A}$ is a proper sequence of exactly N actual merges, containing only merges from the initial tokenizer. This is true initially (line 1). Exchanging μ_p for μ_q (lines 11-12)[3] does not change the number of actual

[1] Most implementations include a pre-tokenization step, which prevents tokens to cross word boundaries. Our presentation makes no such assumption; in our experiments, we use the same pre-tokenization as the tokenizers considered.

[2] This ensures that virtual tokens are split while their parents are still merged.

[3] In line 11, we append a new element to list $\boldsymbol{\mu}_\text{A}$; in line 12 we *remove* an element from the list $\boldsymbol{\mu}_\text{R}$.

Algorithm 1 BPE adaptation

```
Require: Merge list $\boldsymbol{\mu} = [\mu_0, \ldots, \mu_M]$, adaptation corpus $\mathcal{A}$, merge budget $N$
 1: $\boldsymbol{\mu}_{\text{A}} \leftarrow [: \mu_{N-1}]$, $\boldsymbol{\mu}_{\text{R}} \leftarrow [\mu_N : \mu_M]$
 2: $\mathcal{A}^t \leftarrow \text{apply}(\boldsymbol{\mu}_{\text{A}}, \mathcal{A})$
 3: Compute unigram frequencies:
 4:     $\text{Freq}(\mu, \mathcal{A}^t), \forall \mu \in \boldsymbol{\mu}_{\text{A}}$
 5: Compute bigram frequencies:
 6:     $\text{Freq}(\mu\mu', \mathcal{A}^t), \forall (\mu, \mu') \in \boldsymbol{\mu}_{\text{A}} \times \boldsymbol{\mu}_{\text{A}}$
 7: $\mu_p \leftarrow \arg\min_{\mu \in \boldsymbol{\mu}_{\text{A}}} \text{Freq}(\mu, \mathcal{A}^t)$
 8: $\mu_q \leftarrow \arg\max_{\mu=(x,y) \in \boldsymbol{\mu}_{\text{R}}} \text{Freq}(xy, \mathcal{A}^t)$
 9: while $\text{Freq}(\mu_p, \mathcal{A}^t) < \text{Freq}(\mu_q, \mathcal{A}^t)$ do
10:     Make $\mu_p$ a virtual merge
11:     $\mathcal{A}^t \leftarrow \text{unapply}(\mu_p, \mathcal{A}^t)$
12:     $\mathcal{A}^t \leftarrow \text{apply}(\mu_q, \mathcal{A}^t)$
13:     $\boldsymbol{\mu}_{\text{A}} \leftarrow \boldsymbol{\mu}_{\text{A}} + [\mu_q]$
14:     $\boldsymbol{\mu}_{\text{R}} \leftarrow \boldsymbol{\mu}_{\text{R}} - [\mu_q]$
15:     Update unigram and bigram frequencies
16:     $\mu_p \leftarrow \arg\min_{\mu \in \boldsymbol{\mu}_{\text{A}}} \text{Freq}(\mu, \mathcal{A}^t)$
17:     $\mu_q \leftarrow \arg\max_{\mu=(x,y) \in \boldsymbol{\mu}_{\text{R}}} \text{Freq}(xy, \mathcal{A}^t)$
18: end while
19: return $\boldsymbol{\mu}_{\text{A}}$
```

merges. Furthermore, as we only promote tokens from $\boldsymbol{\mu}_R$ when both parents exist in $\boldsymbol{\mu}_{\text{A}}$, this exchange also preserves the properness of the merge list. This procedure is illustrated in Appendix A.1.

As we rely on empirical counts the development corpus, the frequency computations can be unreliable, especially for the rarer events. The end condition (line 9) can be adapted to this uncertainty, for instance by ensuring the frequencies of the incoming and deleted unit differ by some margin.

The greedy merge procedure ensures that each replacement results in a net decrease of the total corpus size. Like for the original BPE algorithm, the complexity is dominated by the cost to compute and update sorted frequency lists, which can be done effectively using appropriate data structures.[4]

4 Experimental Setup

4.1 Overview

Our experiments evaluate the following scenarios.

Monolingual adaptation: In this setup, we adapt an LLM *to process text in only one language, possibly in a restricted domain*. For each language, we collect adaptation data that are used to trim the vocabulary. We then compare the original tokenizer with adapted versions on generation and classification tasks.

Language	Length	Language	Length
English (eng)	1943684	French (fra)	2725460
German (deu)	614542	Spanish (spa)	2443181
Occitan (oci)	693123	Manipuri (mni)	207892
Swahili (swa)	176289	Tamil (tam)	375334

Table 1: Corpus Extracted from Wikipedia. Length is a number of bytes.

Bilingual adaptation In this setup, we adapt the tokenizer to perform machine translation (MT) in *one single language pair*. We optimize the merge list using the two adaptation corpora for these languages, and evaluate the results using MT metrics.

4.2 Data

We use several data sets in our experiments: Wikipedia, PubMed and SIB for monolingual evaluations, FLORES and EMEA for bilingual ones.

Wikipedia. We collect 100 articles from the English, French, German, Manipuri, Occitan, Tamil, Spanish, and Swahili editions of Wikipedia. Articles are retrieved from curated "featured article" lists[5] to ensure content quality. For English, French, German, and Spanish, we restrict our collection to articles promoted between 2024 and 2025. This time window reduces the risk of overlap with the pretraining corpora of LLMs such as Llama-3 and BLOOM. For the low-resource languages (Manipuri, Occitan, Swahili and Tamil), no such time restriction applies. Even then, due to the scarcity of content in Manipuri, we were only able to collect 97 articles for that language. All articles were parsed, cleaned, and segmented into sentences using a custom extraction pipeline. Statistics are in Table 1. The resulting data is randomly split into 50% development and 50% test sets. The development set is used to compute the optimal list of BPE merges for each language. The test set is used to evaluate the resulting tokenizers in terms of compression utility and perplexity.

SIB. The SIB-200 multilingual benchmark is designed to evaluate topic classification across over 200 languages (Adelani et al., 2024). It consists of articles annotated with eight topics: *science*, *technology*, *travel*, *politics*, *sports*, *health*, *entertain-*

[4] https://guillaume-be.github.io/2021-09-16/byte_pair_encoding or https://github.com/marta1994/efficient_bpe_explanation.

[5] https://en.wikipedia.org/wiki/Wikipedia:Featured_articles

ment, and *geography*. We use it for zero-shot topic classification experiments in the same eight languages. In our experiments, we rely on the official train–test splits for SIB.[6] The English training set is used to fine-tune the Llama model; evaluations are performed with the official test sets.

FLORES. FLORES (NLLB Team et al., 2022) is a multilingual benchmark consisting of professionally translated sentence pairs across 200 languages. We focus on two translation tasks, **English–French** and **English–Occitan**, respectively illustrating a high-resource and a low-resource language pair.

For all the experiments on Wikipedia, SIB and FLORES, AdaptBPE relies on the corresponding Wikipedia development sets. We additionally consider specialized monolingual and bilingual corpora in the Medical domain, to showcase a setting where adaptation simultaneously targets a specialized domain and a subset of languages.

PubMed and EMEA PubMed[7] is a large English biomedical corpus containing scientific abstracts and articles, while EMEA (Tiedemann, 2012) consists of English–French parallel texts from the European Medicines Agency. We use a recent (2025) snapshot of PubMed and the latest version of EMEA to extract monolingual English data (PubMed) and bilingual English–French data (EMEA) for domain adaptation experiments.

4.3 Models and their Tokenizer

Our experiments use three tokenizers and models.[8].

BLOOM: Based on a byte level BPE tokenizer, BLOOM (BigScience et al., 2022) uses a vocabulary of 250,880 tokens and supports multilingual input. It was trained on ROOTS (Laurençon et al., 2022) a multilingual corpus spanning texts in 46 languages. In our experiments, we use the smallest BLOOM model (560m parameters).

Llama-3: Developed for the Llama-3 models (Grattafiori et al., 2024), this BPE tokenizer has a vocabulary size of approximately 128k tokens. We experiment with the 8b parameter models.

GPT-2: GPT-2 relies on byte-level BPE with a vocabulary of about 50k tokens. This tokenizer is primarily optimized for English and exhibits little multilingual coverage. We use it to contrast multilingual tokenizers with an English-centric one.

4.4 Baselines

Our experiments evaluate the cost-benefit trade-off of restricting the tokenizer vocabulary to a predefined size N. We compare our results to baselines that use a limited vocabulary of size N:

1. **Baseline 1 (First$_k$) - Merge-Truncated Tokenization:** We apply only the first N merges from the pretrained tokenizer to the corpus, without any refinement or filtering.

2. **Baseline 2 (First$_{k>0}$) - Merge-Truncated Tokenization with Data-aware Filtering:** We apply the first N merges that appear at least once in the dataset, thereby excluding zero-frequency merges before tokenization.[9]

3. **Baseline 3 (Top$_k$) - Full Merge with Selective Unmerging:** We first apply all merges defined by the original tokenizer to the corpus. We then record the N most frequent tokens. Less frequent tokens are recursively unmerged until their subparts belong to the top-N list or to the base tokens. The resulting vocabulary thus matches the size constraint.

For completeness, we also report the performance achieved with the full tokenizer vocabulary.

4.5 Setup and Metrics

4.5.1 Monolingual Generation

We report results using two main metrics.

Compression Utility (CU) (Zouhar et al., 2023), is defined as the relative reduction in corpus size (in characters) after tokenization:

$$\text{CU} = \frac{|\mathcal{W}| - |\,\text{apply}^*(\boldsymbol{\mu}, \mathcal{W})|}{|\mathcal{W}|}$$

where $|\mathcal{W}|$ is the length of text $\mathcal{W}$, and $\text{apply}^*(\boldsymbol{\mu}, \mathcal{W})$ recursively applies all the merges in $\boldsymbol{\mu}$ in their appearance order to text $\mathcal{W}$. CU measures the relative reduction in corpus size after applying the tokenizer merges and has been showed to correlate well with performance on downstream tasks (Goldman et al., 2024). Compression utility is also directly related to the *tokenizer fertility*[10] and to the length of the tokenized files; it thus has a direct impact on computation costs.

[6] https://huggingface.co/datasets/Davlan/sib200

[7] https://pubmed.ncbi.nlm.nih.gov

[8] Appendix A.2 additionally reports results obtained with Qwen-3 (Bai et al., 2023)

[9] In the terms of Section 3, this is done by making the zero-frequency tokens virtual, so as to preserve properness.

[10] Fertility is the average number of tokens per word, equal to $\frac{\text{CU}-|\mathcal{W}|}{L_W}$, with L_W the length in words.

Perplexity (Cover and Thomas, 1991) quantifies the model's uncertainty in predicting the next token. We report *word-level perplexity* computed with `lm-evaluation-harness` (Gao et al., 2024),[11] enabling cross-tokenizer comparisons.

4.5.2 Zero-shot Cross-lingual Classification

Following Sanh et al. (2022) and Adelani et al. (2024), we adopt a simple template for zero-shot cross-lingual classification. Each input sentence is wrapped in the following prompt:

```
“Is this a piece of news regarding
{science/ technology, travel, politics,
sports, health, entertainment, or
geography}? {INPUT}”
```

The task consists of assigning each input sentence to one of the seven predefined categories. We report classification accuracy on the SIB test partition as the main metric. We first fine-tune Llama-3.1-8b with the English training data, then contrast two experimental settings:

1. Using the fine-tuned Llama model with the full tokenizer to perform topic classification;
2. Using the fine-tuned Llama model with a language-adapted BPE tokenizer, considering a restricted vocabulary of N units.[12]

4.5.3 Machine translation

We use a 5-shot setup for translation. The first five sentence pairs in FLORES-200 test file are selected as in-context samples (and excluded from the BLEU computation) in the following template:

```
“Translate the following French
sentences to English:
French: [source text]
English:[target text]
...
French: [input text]
English:”
```

The last source sentence in the prompt will be translated. We evaluate translation quality using BLEU (Papineni et al., 2002) computed with `sacreBLEU` (Post, 2018).[13] We use greedy decoding with maximum generation limit of 128 tokens for all models.

Translation differs from the other tasks because we need to *actually generate texts with a reduced vocabulary*: to prevent the decoding of tokens that have been pruned, we apply logit masking in the unembedding layer so as to make the generation of such tokens impossible.

5 Results and Analysis

In preliminary experiments, we study the effect of varying the number of merges on the total corpus size. We experiment with BLOOM, for which the training set is precisely documented,[14] and consider two languages: French, which accounts for about 13% of the pretraining tokens, and Occitan, which is hardly represented (Occitan).[15]

We display the learning curves on the development sets on Figures 1(a), 1(b): each value represent the corpus size after running AdaptBPE (or the baselines) with a total budget of N.[16] In both cases, we observe a sharp decrease of the corpus size for the first thousands of merges for all methods, with AdaptBPE however always yielding shorter corpus sizes than alternative pruning methods. As the merge budget reaches 10k to 20k, the return of each additional merge gets smaller for all methods, slowly converging to the tokenization obtained with the complete tokenizer and its 250k merges.

For all follow-up experiments, we use a budget of 15k, which for BLOOM corresponds to a small fraction (about 6%) of the vocabulary and the corresponding embedding parameters: for the smallest model (with 560m parameters), embeddings account for about 40% of the parameters. The larger Llama (8b) only contains about 500m embeddings parameters, corresponding to about 6.5% of the total parameter set.

5.1 Monolingual Evaluations

Wikipedia Corpus Table 2 reports both compression utility and perplexity scores for four languages using tokenizers adapted under a merge budget of 15k.[17] As expected, using the complete vocabulary yields the best performance across the board. Among models relying on a constraint vocabulary, AdaptBPE consistently yields better compression utilities than the three baseline methods (see sec-

[11] Version: `0.4.8`, commit: `92d6139`

[12] Details regarding SIB are in appendix A.3.

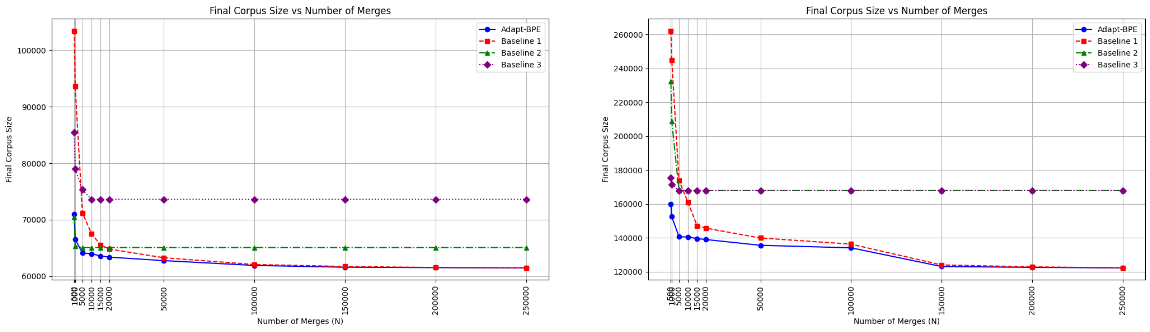
[13] Signature: `nrefs:1|case:mixed|eff:no|tok:13a|smooth:exp|version:2.5.1`

[14] `https://huggingface.co/bigscience/bloom#training-data`

[15] The processing of Occitan is facilitated by the large share of Romance languages in BLOOM's pretraining data.

[16] Corpus sizes are thus monotonically decreasing.

[17] Full results, for 8 languages, are in Appendix A.2.



(a) French

(b) Occitan

Figure 1: Corpus size with increasing number of merges (BLOOM Tokenizer).

	GPT-2	BLOOM	Llama-3	GPT-2	BLOOM	Llama-3	GPT-2	BLOOM	Llama-3	GPT-2	BLOOM	Llama-3
	English (1.32m)			French (0.97m)			Occitan (0.34m)			Manipuri (0.04m)		
Full vocab	0.66/09.9	0.66/05.9	0.66/04.5	0.62/14.9	0.63/08.5	0.63/06.7	0.59/04.8	0.60/05.8	0.66/04.0	0.07/02.2	0.07/03.6	0.08/01.9
First$_k$	0.62/13.2	0.61/07.7	0.62/05.1	0.56/17.9	0.59/10.6	0.57/08.4	0.55/06.0	0.56/06.1	0.60/05.9	0.03/03.2	0.03/03.7	0.03/02.3
First$_{k>0}$	0.63/11.3	0.63/07.0	0.64/05.0	0.59/16.1	0.61/09.9	0.61/07.8	0.56/05.9	0.58/06.5	0.63/05.3	0.05/03.3	0.05/03.8	0.05/02.0
Top$_k$	0.63/11.8	0.64/06.5	0.64/04.8	0.59/16.1	0.61/09.4	0.61/07.6	0.58/05.7	0.57/06.1	0.58/04.4	0.03/03.3	0.03/03.5	0.03/02.0
AdaptBPE	0.64/10.1	0.64/06.1	0.64/04.7	0.61/15.3	0.61/09.0	0.61/07.0	0.58/05.0	0.59/06.0	0.65/04.2	0.06/02.3	0.06/03.7	0.07/02.0

Table 2: Compression utility and perplexity for $N = 15$k merges on the Wikipedia test sets. The raw corpus size (before tokenization) is given in parentheses alongside each language. Each cell reports utility/perplexity.

tion 4.4), resulting in compact tokenizations for all languages and tokenizers, almost closing the gap with the full vocabulary.

While the Full-Vocab tokenizer tends to achieve lower perplexity (and higher compression utility) on general-domain corpora (e.g., Wikipedia), Adapt-BPE consistently matches or exceeds performance on domain-specific corpora, such as PubMed and EMEA, as shown in Table 4.

In particular, our method improves over the $top_{k>0}$ baseline: even though both approaches are close, AdaptBPE still achieves better compression scores. This is because its merging policy is based on more reliable frequency estimates performed on the adaptation data. Using a reduced vocabulary does not seem to harm perplexity scores either: they are better or comparable than the baselines using a 15k vocabulary, demonstrating that our strategy to select merges better aligns tokenization with model expectations. The benefits of our method are especially clear for morphologically complex or low-resource languages such as Occitan and Manipuri, where other merge selections techniques seem to struggle.

Merge Depth Analysis We report in Table 3 the index of the last selected merge for each language after refinement under a fixed 15k merge budget. This index reflects how far in the BLOOM merge list the algorithm had to go to adapt the vocabulary.

eng	fra	deu	spa	oci	mni	swh	tam
18678	25784	22580	30047	115565	180908	84037	91135

Table 3: Index of last merge (BLOOM tokenizer).

High resource languages like English, French, German, and Spanish have relatively low indices, indicating that they are well covered with the frequent merges. For Occitan and Manipuri, these values are much higher, reflecting their small representation in the pretraining data.

Medical Domain Table 4 reports the performance of AdaptBPE on two medical corpora (PubMed, in English and EMEA, in English and French). Regarding compression, we again observe that our method does better than alternatives, and closes the gap with the full vocabulary setup. Perplexity scores show an even better trend as we observe values that are *significantly better than those obtained with the complete vocabulary*.

5.2 Zero-shot Text Classification

Table 5 presents zero-shot classification results when using either a full vocabulary in inference, or a language-adapted token set of 15k merges. We observe here a large drop in performance of En-

	Compression utility / Perplexity						BLEU scores	
	PubMed (1.29m)		EMEA-eng (39.48m)		EMEA-fra (45.76m)		EMEA eng→fra	
	BLOOM	**Llama-3**	**BLOOM**	**Llama-3**	**BLOOM**	**Llama-3**	**BLOOM**	**Llama-3**
Full vocab	0.62/18.2	0.62/16.6	0.62/13.2	0.60/12.6	0.60/16.1	0.63/14.3	42.9	45.4
First$_k$	0.60/18.6	0.59/17.8	0.60/14.1	0.59/13.3	0.58/16.9	0.61/15.9	40.3	43.8
First$_{k>0}$	0.60/17.3	0.60/16.8	0.60/13.3	0.59/13.1	0.59/16.6	0.62/15.8	40.6	44.5
Top$_k$	0.61/15.6	0.60/16.3	0.60/10.2	0.60/12.7	0.60/16.6	0.62/14.7	41.2	44.9
AdaptBPE	**0.62/11.4**	**0.65/12.5**	**0.64/6.7**	**0.60/8.0**	**0.60/12.2**	**0.63/11.8**	**42.9**	**45.6**

Table 4: Compression utility / perplexity scores for $N = 15$k merges on PubMed (English) and EMEA (English and French) test sets **(left side)**, and BLEU scores for 5-shot English–French translation on the EMEA test set **(right side)**. Corpus sizes in bytes are in parentheses.

Setting	**eng**	**fra**	**deu**	**spa**	**oci**	**mni**	**swh**	**tam**	**sin**
FT + Full vocab	54.9	52.0	48.1	43.2	32.1	16.3	25.2	23.9	17.2
FT + AdaptBPE	46.5	48.4	46.2	41.0	30.2	16.3	24.9	23.3	17.1
AdaptBPE+ FT	45.3	48.8	47.5	42.4	31.2	16.3	25.4	23.5	17.2

Table 5: Zero-shot classification accuracy for fine-tuned (FT) Llama-3 models with a full or trimmed vocabulary.

Method	**BLOOM**				**Llama-3**			
	eng-fra		**eng-oci**		**eng-fra**		**eng-oci**	
	15k	30k	15k	30k	15k	30k	15k	30k
Full vocab	39.5	40.2	25.3	26.2	40.4	40.8	27.0	27.8
First$_k$	30.3	34.9	18.2	20.5	32.0	36.5	19.0	21.1
First$_{k>0}$	35.2	36.9	21.7	23.4	36.0	38.2	23.5	24.6
Top$_k$	37.1	38.2	22.3	24.1	37.9	39.0	24.1	25.7
AdaptBPE	**38.9**	**39.1**	**24.8**	**25.9**	**39.7**	**40.2**	**26.3**	**27.2**

Table 6: BLEU scores on FLORES test with 5-shot MT with merge budgets of 15k and 30k.

glish, owing to the mismatch between training data (using the original tokenizer) and test data (using a reduced one): rarer tokens are helpful in topic classification and their removal hurts performance. Already for French, the impact is lessen, and progressively vanishes for Spanish and German, down to the low-resource language set, where the performance remains poor. This shows that *cross-lingual transfer is robust to vocabulary pruning*, and its effect subsist even when only using a small portion of the original vocabulary. In an additional experiment, we *combine Adapt*BPE *and fine-tuning*, recovering part of the accuracy loss for all languages but English and Tamil.

5.3 Machine Translation

Table 6 reports BLEU scores on FLORES datasets for English-French and English-Occitan translation using BLOOM and Llama-3 models (5-shot setting). We evaluate two merge budgets: 15k, corresponding to the monolingual vocabulary size of previous sections, and 30k, doubling the vocabulary for the bilingual setting. The refined tokenizer consistently improves translation quality relative to baselines and closely matches the performance of the full model at a reduced computational cost (details in Appendix A.4). These results illustrate the potential payoffs of using AdaptBPE.

Table 4 (right) displays additional MT results obtained with a corpus of medical texts, where we observe a similar trend: better results for AdaptBPE than for baselines, matching the scores obtained with the full tokenizer, at a reduced cost.

6 Conclusion

This work studies the adaptation of a pretrained BPE vocabulary in contexts where a model is specialized (e.g., by fine-tuning) on a small number of domains or languages. For this, we propose AdaptBPE, a vocabulary refinement algorithm that does not imply any change in the model weights.

We show that trimming the token set can act as a form of regularization: by constraining the vocabulary size, it biases the model toward more generalizable subword representations. A key property of our lightweight approach is that it minimally intervenes on the pretrained tokenizer by (a) marking merges that need to be undone; (b) truncating the set of merges to a prefix. As such, the resulting tokenizer remains fully compatible with the pretrained model and its derivatives (e.g., fine-tuned versions). Furthermore, the performance observed with AdaptBPE are achieved prior to model adaptation; they would likely increase after finetuning on the adaptation data.

In our future work, we would like to develop methods that optimize the choice of the budget N dynamically, potentially varying according to the targeted task(s). Using a predefined number of merges can yield suboptimal vocabularies. Another improvement will be to better take the uncertainty of counts estimates into account, as using raw count might cause our algorithm to overtrain its token inventory.

7 Limitations

AdaptBPE is a post-hoc refinement method that improves tokenizer efficiency without modifying model parameters.

However, it does not benefit from joint optimization with the model, which could unlock greater improvements. Additionally, the merge budget N is manually fixed; determining an optimal value remains an open question.

Finally, our experiments focus on a small set of multilingual BPE-based tokenizers and a limited set of models, languages and tasks, leaving generalization to other settings for future exploration.

Acknowledgments

This work was funded by the French Agence Nationale de la Recherche (ANR) under the project TraLaLaM ("ANR-23-IAS1-0006"). This work was also granted access to the HPC resources of IDRIS under the allocation 2025-AD011015117R2 made by GENCI. The authors wish to thank the ACL/ARR reviewers for their insightful comments and suggestions.

References

David Ifeoluwa Adelani, Hannah Liu, Xiaoyu Shen, Nikita Vassilyev, Jesujoba O. Alabi, Yanke Mao, Haonan Gao, and En-Shiun Annie Lee. 2024. SIB-200: A simple, inclusive, and big evaluation dataset for topic classification in 200+ languages and dialects. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 226–245, St. Julian's, Malta. Association for Computational Linguistics.

Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, Binyuan Hui, Luo Ji, Mei Li, Junyang Lin, Runji Lin, Dayiheng Liu, Gao Liu, Chengqiang Lu, Keming Lu, and 29 others. 2023. Qwen technical report. *Preprint*, arXiv:2309.16609.

Thomas Bauwens and Pieter Delobelle. 2024. BPE-knockout: Pruning pre-existing BPE tokenisers with backwards-compatible morphological semi-supervision. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 5810–5832, Mexico City, Mexico. Association for Computational Linguistics.

Martin Berglund and Brink van der Merwe. 2023. Formalizing BPE tokenization. *Electronic Proceedings in Theoretical Computer Science*, 388:16–27.

Workshop BigScience, Teven Le Scao, Angela Fan, Christopher Akiki, Ellie Pavlick, Suzana Ilić, Daniel Hesslow, Roman Castagné, Alexandra Sasha Luccioni, François Yvon, Matthias Gallé, Jonathan Tow, Alexander M. Rush, Stella Biderman, Albert Webson, Pawan Sasanka Ammanamanchi, Thomas Wang, Benoît Sagot, Niklas Muennighoff, and 373 others. 2022. BLOOM: A 176B-Parameter Open-Access Multilingual Language Model. *Preprint*, arXiv:2211.05100.

Nikolay Bogoychev, Pinzhen Chen, Barry Haddow, and Alexandra Birch. 2024. The ups and downs of large language model inference with vocabulary trimming by language heuristics. In *Proceedings of the Fifth Workshop on Insights from Negative Results in NLP*, pages 148–153, Mexico City, Mexico. 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.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, and 12 others. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Pavel Chizhov, Catherine Arnett, Elizaveta Korotkova, and Ivan P. Yamshchikov. 2024. BPE gets picky: Efficient vocabulary refinement during tokenizer training. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 16587–16604, Miami, Florida, USA. Association for Computational Linguistics.

Marco Cognetta, Tatsuya Hiraoka, Rico Sennrich, Yuval Pinter, and Naoaki Okazaki. 2024. An analysis of BPE vocabulary trimming in neural machine translation. In *Proceedings of the Fifth Workshop on Insights from Negative Results in NLP*, pages 48–50, Mexico City, Mexico. Association for Computational Linguistics.

Thomas M. Cover and Joy A. Thomas. 1991. *Elememts of Infomation Theory*. John Wiley and Sons.

Severino Da Dalt, Joan Llop, Irene Baucells, Marc Pamies, Yishi Xu, Aitor Gonzalez-Agirre, and Marta

Villegas. 2024. FLOR: On the effectiveness of language adaptation. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 7377–7388, Torino, Italia. ELRA and ICCL.

Konstantin Dobler and Gerard de Melo. 2023. FOCUS: Effective embedding initialization for monolingual specialization of multilingual models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 13440–13454, Singapore. Association for Computational Linguistics.

Zhili Feng, Tanya Marwah, Lester Mackey, David Alvarez-Melis, and Nicolo Fusi. 2024. Adapting language models via token translation. In *Workshop on Machine Learning and Compression, NeurIPS 2024*.

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

Leo Gao, Jonathan Tow, Baber Abbasi, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Alain Le Noac'h, Haonan Li, Kyle McDonell, Niklas Muennighoff, Chris Ociepa, Jason Phang, Laria Reynolds, Hailey Schoelkopf, Aviya Skowron, Lintang Sutawika, and 5 others. 2024. The language model evaluation harness.

Leonidas Gee, Leonardo Rigutini, Marco Ernandes, and Andrea Zugarini. 2023. Multi-word tokenization for sequence compression. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing: Industry Track*, pages 612–621, Singapore. Association for Computational Linguistics.

Leonidas Gee, Andrea Zugarini, Leonardo Rigutini, and Paolo Torroni. 2022. Fast vocabulary transfer for language model compression. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing: Industry Track*, pages 409–416, Abu Dhabi, UAE. Association for Computational Linguistics.

Saibo Geng, Nathan Ranchin, Yunzhen Yao, Maxime Peyrard, Chris Wendler, Michael Gastpar, and Robert West. 2025. Zip2zip: Inference-time adaptive vocabularies for language models via token compression. In *ES-FoMo III: 3rd Workshop on Efficient Systems for Foundation Models*.

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.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, and 542 others. 2024. The Llama 3 herd of models. *Preprint*, arXiv:2407.21783.

Jue Hou, Anisia Katinskaia, Anh-Duc Vu, and Roman Yangarber. 2023. Effects of sub-word segmentation on performance of transformer language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 7413–7425, Singapore. Association for Computational Linguistics.

Edward J Hu, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*.

Ayyoob Imani, Peiqin Lin, Amir Hossein Kargaran, Silvia Severini, Masoud Jalili Sabet, Nora Kassner, Chunlan Ma, Helmut Schmid, André Martins, François Yvon, and Hinrich Schütze. 2023. Glot500: Scaling multilingual corpora and language models to 500 languages. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1082–1117, Toronto, Canada. Association for Computational Linguistics.

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 (Volume 1: Long Papers)*, pages 66–75, Melbourne, Australia. Association for Computational Linguistics.

Hugo Laurençon, Lucile Saulnier, Thomas Wang, Christopher Akiki, Albert Villanova del Moral, Teven Le Scao, Leandro Von Werra, Chenghao Mou, Eduardo González Ponferrada, Huu Nguyen, Jörg Frohberg, Mario Šaško, Quentin Lhoest, Angelina McMillan-Major, Gérard Dupont, Stella Biderman, Anna Rogers, Loubna Ben allal, Francesco De Toni, and 35 others. 2022. The BigScience ROOTS Corpus: A 1.6TB Composite Multilingual Dataset. In *Thirty-Sixth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Yuxi Li, Yi Liu, Gelei Deng, Ying Zhang, Wenjia Song, Ling Shi, Kailong Wang, Yuekang Li, Yang Liu, and Haoyu Wang. 2024. Glitch tokens in large language models: Categorization taxonomy and effective detection. *Proceedings of the ACM Software Engineering Conference*, 1(FSE).

Haoran Lian, Yizhe Xiong, Jianwei Niu, Shasha Mo, Zhenpeng Su, Zijia Lin, Hui Chen, Jungong Han, and Guiguang Ding. 2025. Scaffold-BPE: Enhancing byte pair encoding for large language models with simple and effective scaffold token removal. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 24539–24548.

Peiqin Lin, Shaoxiong Ji, Jörg Tiedemann, André F. T. Martins, and Hinrich Schütze. 2024. Mala-500: Massive language adaptation of large language models. *CoRR*, abs/2401.13303.

Chengyuan Liu, Shihang Wang, Lizhi Qing, Kun Kuang, Yangyang Kang, Changlong Sun, and Fei Wu. 2024. Gold panning in vocabulary: An adaptive method for vocabulary expansion of domain-specific LLMs. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 7442–7459, Miami, Florida, USA. Association for Computational Linguistics.

Siyang Liu, Naihao Deng, Sahand Sabour, Yilin Jia, Minlie Huang, and Rada Mihalcea. 2023. Task-adaptive tokenization: Enhancing long-form text generation efficacy in mental health and beyond. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 15264–15281, Singapore. Association for Computational Linguistics.

Yinquan Lu, Wenhao Zhu, Lei Li, Yu Qiao, and Fei Yuan. 2024. LLaMAX: Scaling Linguistic Horizons of LLM by Enhancing Translation Capabilities Beyond 100 Languages. *CoRR*, abs/2407.05975.

Manuel Mager, Arturo Oncevay, Elisabeth Mager, Katharina Kann, and Thang Vu. 2022. BPE vs. morphological segmentation: A case study on machine translation of four polysynthetic languages. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 961–971, Dublin, Ireland. Association for Computational Linguistics.

Benjamin Minixhofer, Fabian Paischer, and Navid Rekabsaz. 2022. WECHSEL: Effective initialization of subword embeddings for cross-lingual transfer of monolingual language models. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3992–4006, Seattle, United States. 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, and 20 others. 2022. No language left behind: Scaling human-centered machine translation. *arXiv preprint arXiv:2207.04672*.

Yaniv Ovadia, Emily Fertig, Jie Ren, Zachary Nado, D. Sculley, Sebastian Nowozin, Joshua Dillon, Balaji Lakshminarayanan, and Jasper Snoek. 2019. Can you trust your model's uncertainty? evaluating predictive uncertainty under dataset shift. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Matt Post. 2018. A call for clarity in reporting BLEU scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191, Brussels, Belgium. Association for Computational Linguistics.

Ivan Provilkov, Dmitrii Emelianenko, and Elena Voita. 2020. BPE-dropout: Simple and effective subword regularization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1882–1892, Online. Association for Computational Linguistics.

François Remy, Pieter Delobelle, Hayastan Avetisyan, Alfiya Khabibullina, Miryam de Lhoneux, and Thomas Demeester. 2024. Trans-tokenization and cross-lingual vocabulary transfers: Language adaptation of LLMs for low-resource NLP. In *First Conference on Language Modeling*.

Vin Sachidananda, Jason Kessler, and Yi-An Lai. 2021. Efficient domain adaptation of language models via adaptive tokenization. In *Proceedings of the Second Workshop on Simple and Efficient Natural Language Processing*, pages 155–165, Virtual. Association for Computational Linguistics.

Victor Sanh, Albert Webson, Colin Raffel, Stephen H. Bach, Lintang Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Teven Le Scao, Arun Raja, Manan Dey, M. Saiful Bari, Canwen Xu, Urmish Thakker, Shanya Sharma, Eliza Szczechla, Taewoon Kim, Gunjan Chhablani, Nihal V. Nayak, and 22 others. 2022. Multitask prompted training enables zero-shot task generalization. In *Proceedings of the Tenth International Conference on Learning Representations*, ICLR 2022.

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.

Pranaydeep Singh, Eneko Agirre, Gorka Azkune, Orphee De Clercq, and Els Lefever. 2025. EnerGIZAr: Leveraging GIZA++ for effective tokenizer initialization. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 2124–2137, Vienna, Austria. Association for Computational Linguistics.

Gemma Team, Thomas Mesnard, Cassidy Hardin, Robert Dadashi, Surya Bhupatiraju, Shreya Pathak, Laurent Sifre, Morgane Rivière, Mihir Sanjay Kale, Juliette Love, Pouya Tafti, Léonard Hussenot, Pier Giuseppe Sessa, Aakanksha Chowdhery, Adam Roberts, Aditya Barua, Alex Botev, Alex Castro-Ros, Ambrose Slone, and 89 others. 2024. Gemma: Open models based on Gemini research and technology. *Preprint*, arXiv:2403.08295.

Jörg Tiedemann. 2012. Parallel data, tools and interfaces in OPUS. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)*, pages 2214–2218, Istanbul, Turkey. European Language Resources Association (ELRA).

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023. LLaMA: Open and Efficient Foundation Language Models. *Preprint*, arXiv:2302.13971.

Asahi Ushio, Yi Zhou, and Jose Camacho-Collados. 2023. Efficient multilingual language model compression through vocabulary trimming. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 14725–14739, Singapore. Association for Computational Linguistics.

Clara Vania and Adam Lopez. 2017. From characters to words to in between: Do we capture morphology? In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2016–2027, Vancouver, Canada. Association for Computational Linguistics.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Łukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, and 12 others. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *Preprint*, arXiv:1609.08144.

Brian Siyuan Zheng, Alisa Liu, Orevaoghene Ahia, Jonathan Hayase, Yejin Choi, and Noah A. Smith. 2025. Broken Tokens? Your Language Model can Secretly Handle Non-Canonical Tokenizations. *arXiv preprint*. ArXiv:2506.19004 [cs].

Vilém Zouhar, Clara Meister, Juan Gastaldi, Li Du, Tim Vieira, Mrinmaya Sachan, and Ryan Cotterell. 2023. 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 Appendix

A.1 AdaptBPE at work

Figure 2 illustrates the merge / unmerge procedure for various pruning strategies and Figure 3 exhibit the merge order change with and without AdaptBPE.

A.2 Monolingual Evaluation

We report in Table 7 the compression utility and perplexity for the full set of eight languages considered in our experiments, also reporting results obtained with Qwen-3 (Bai et al., 2023). For Qwen, our experiments use the 8 billion parameter model; the associated tokenizer contains 151,656 pieces.

A.3 Text classification: SIB

We fine tune the Llama model with 701 training instances as follows. We first replace the expected category labels in the training data with numbers (from 1 to 7), so that the model outputs are not affected by changes in tokenization. Fine-tuning is performed with LoRA (Hu et al., 2022) implemented in the HuggingFace library[18], with training parameters of 3 epochs, 2e-4 learning rate. In inference, we generate 5 tokens with greedy decoding, and match the expected label with the first token.

A.4 Computational Costs

Trimming the Llama vocabulary down to 15k units enables a potential saving of about 900m parameters in the embedding and the final layers when generating texts; as the projection matrix needs to be stored in memory, pruning could also significantly impact the memory total footprint. For the experiments with Machine Translation (EMEA, English into French) of section 5.3, we monitored the cost-benefit tradeoff of using AdaptBPE compared to the full model: the number of tokens, consequently the average sentence length, increases by 3.6%; yet, the total computation time for 1k sentences is reduced by approximately 2 minutes (out of 23.9, a reduction of 8.4%), owing to faster computations of the softmax in the output layer. Even without making any change to the inference code, or trying to optimize the vocabulary size in inference, we still see a clear gain in using AdaptBPE over the full model. Details are in Table 8.

[18] https://huggingface.co/docs/peft/main/en/conceptual_guides/lora

	GPT-2	BLOOM	Llama-3	Qwen-3	GPT-2	BLOOM	Llama-3	Qwen-3	GPT-2	BLOOM	Llama-3	Qwen-3	GPT-2	BLOOM	Llama-3	Qwen-3
	English (1.32m)				French (0.97m)				German (0.30m)				Spanish (1.18m)			
Full vocab	0.66/09.9	0.66/5.9	0.66/4.5	0.69/3.87	0.62/14.9	0.63/8.5	0.63/6.7	0.66/4.44	0.66/20.8	0.69/17.6	0.71/16.1	0.70/5.23	0.63/12.6	0.64/6.8	0.63/5.2	0.65/3.98
First$_k$	0.62/13.2	0.61/7.7	0.62/5.1	0.63/4.31	0.56/17.9	0.59/10.6	0.57/8.4	0.61/5.42	0.59/29.7	0.61/24.1	0.61/20.3	0.67/7.32/	0.58/16.1	0.60/8.6	0.58/6.3	0.61/4.86
First$_{k>0}$	0.63/11.3	0.63/7.0	0.64/5.0	0.65/4.03	0.59/16.1	0.61/9.9	0.61/7.8	0.62/5.14	0.65/23.6	0.64/23.5	0.67/20.0	0.68/6.92	0.61/14.2	0.61/8.1	0.61/6.0	0.62/4.56
Top$_k$	0.63/11.8	0.64/6.5	0.64/4.8	0.66/3.91	0.59/16.1	0.61/9.4	0.61/7.6	0.63/4.85	0.65/23.6	0.67/20.3	0.67/19.0	0.69/6.76	0.61/14.3	0.62/7.5	0.61/5.4	0.63/4.10
AdaptBPE	0.64/10.1	0.64/6.1	0.64/4.7	0.67/3.90	0.61/15.3	0.61/9.0	0.61/7.0	0.65/4.65	0.65/21.1	0.68/18.6	0.70/17.3	0.70/6.31	0.62/13.0	0.62/7.0	0.61/5.3	0.64/4.02
	Occitan (0.34m)				Manipuri (0.04m)				Swahili (0.38m)				Tamil (0.07m)			
Full vocab	0.59/4.8	0.60/5.8	0.66/4.0	0.63/3.42	0.07/2.2	0.07/3.6	0.08/1.9	0.08/3.27	0.60/5.7	0.64/6.3	0.63/5.3	0.67/7.29	0.05/1.9	0.06/3.2	0.07/2.6	0.06/3.85
First$_k$	0.55/6.0	0.56/6.1	0.60/5.9	0.59/4.56	0.03/3.2	0.03/3.7	0.03/2.3	0.03/3.93	0.56/6.2	0.58/6.8	0.53/5.8	0.62/9.10	0.04/2.2	0.05/3.7	0.04/2.7	0.04/4.26
First$_{k>0}$	0.56/5.9	0.58/6.5	0.63/5.3	0.60/4.55	0.05/3.3	0.05/3.8	0.05/2.3	0.05/3.87	0.58/6.1	0.61/6.8	0.61/5.7	0.64/8.32	0.05/2.2	0.05/3.7	0.05/2.7	0.05/4.12
Top$_k$	0.58/5.7	0.57/6.1	0.58/4.4	0.62/4.12	0.03/3.3	0.03/3.5	0.03/2.1	0.05/3.56	0.59/5.9	0.61/6.5	0.61/5.6	0.65/8.20	0.05/2.1	0.05/3.4	0.05/2.6	0.05/4.01
AdaptBPE	0.58/5.0	0.59/6.0	0.65/4.2	0.62/3.73	0.06/2.3	0.06/3.7	0.07/2.0	0.06/3.34	0.60/5.8	0.62/6.4	0.62/5.5	0.66/8.03	0.05/2.0	0.05/3.4	0.06/2.6	0.06/3.85

Table 7: Compression utility and perplexity for $N = 15$k merges on the Wikipedia test sets. Each cell reports utility/perplexity.

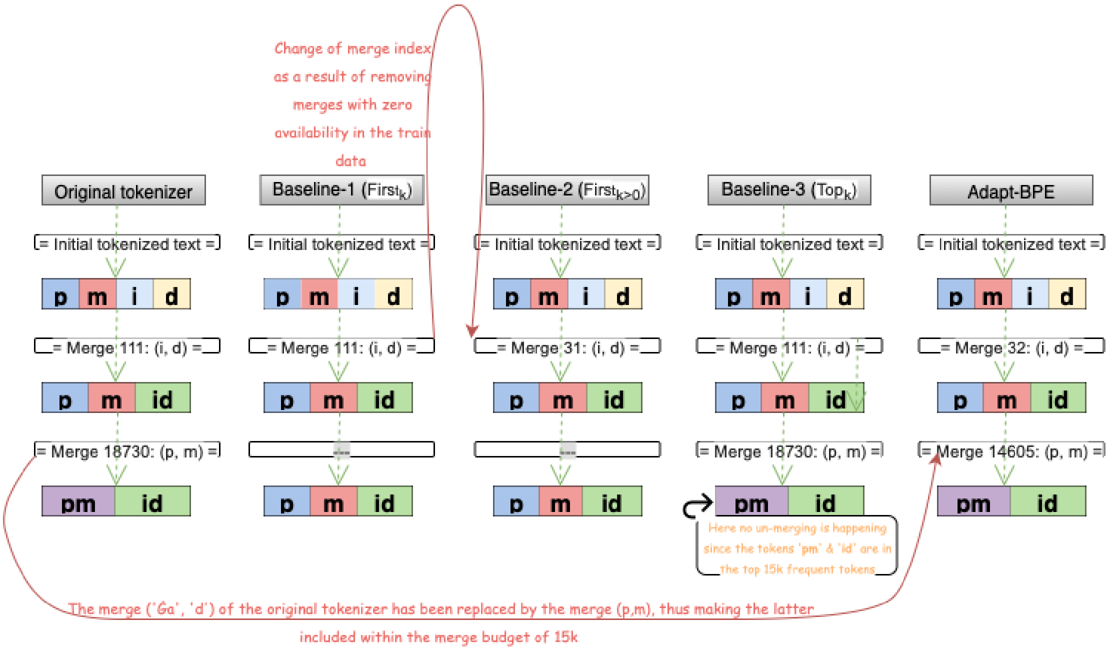

Change of merge index as a result of removing merges with zero availability in the train data
Original tokenizer
Baseline-1 (First$_k$)
Baseline-2 (First$_{k>0}$)
Baseline-3 (Top$_k$)
Adapt-BPE
= Initial tokenized text =
p m i d
= Merge 111: (i, d) =
= Merge 31: (i, d) =
= Merge 32: (i, d) =
p m id
= Merge 18730: (p, m) =
= Merge 14605: (p, m) =
pm id
Here no un-merging is happening since the tokens 'pm' & 'id' are in the top 15k frequent tokens
The merge ('Ġa', 'd') of the original tokenizer has been replaced by the merge (p,m), thus making the latter included within the merge budget of 15k

Figure 2: An illustration of BPE merging unmerging procedures The selected token is **"pmid"** extracted from PubMed data and the original tokenizer is BLOOM.

Tokenizer	**Time (s)**	**Tokens**	**Tok/s**
Original	1548.10	30,804	19.90
AdaptBPE	1432.88	31,904	22.27

Table 8: Inference times for Llama computing the translation of 1k EMEA sentence pairs, comparing the original and AdaptBPE tokenizers.

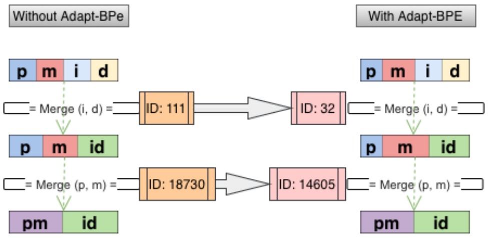


Figure 3: An illustration of merge order change with AdaptBPE. The selected token is **"pmid"** extracted from PubMed data and the original tokenizer is BLOOM