How Can We Effectively Expand the Vocabulary of LLMs with 0.01GB of Target Language Text?

Atsuki Yamaguchi[1*], Aline Villavicencio[1,2,3,4], Nikolaos Aletras[1]

[1]School of Computer Science, University of Sheffield, United Kingdom
ayamaguchi1@sheffield.ac.uk, n.aletras@sheffield.ac.uk
[2]Department of Computer Science, University of Exeter, United Kingdom
A.Villavicencio@exeter.ac.uk
[3]The Alan Turing Institute, United Kingdom
[4]Department of Bioinformatics, Federal University of Rio Grande do Norte, Brazil

Large language models (LLMs) have shown remarkable capabilities in many languages beyond English. Yet, LLMs require more inference steps when generating non-English text due to their reliance on English-centric tokenizers and vocabulary, resulting in higher usage costs to non-English speakers. Vocabulary expansion with target language tokens is a widely used cross-lingual vocabulary adaptation approach to remedy this issue. Despite its effectiveness in inference speedup, previous work on vocabulary expansion has focused on high-resource settings assuming access to a substantial amount of target language data to effectively initialize the embeddings of the new tokens and adapt the LLM to the target language. However, vocabulary expansion in low-resource settings has yet to be explored. In this article, we investigate vocabulary expansion in low-resource settings by considering embedding initialization methods and continual pre-training strategies. Through extensive experiments across typologically diverse languages, tasks, and models, we establish a set of strategies to perform vocabulary expansion for faster inference, while striving to maintain competitive downstream performance to baselines. This is achieved with only 30K sentences (∼0.01GB text data) from the target language.[1]

1. Introduction

Large language models (LLMs) have strong capabilities in English and other languages (OpenAI 2023; OpenAI et al. 2024; Touvron et al. 2023; Jiang et al. 2023; Groeneveld

* Corresponding author.

[1] Our code and models are available via GitHub.

Action Editor: Minlie Huang. Submission received: 3 March 2025; revised version received: 30 July 2025; accepted for publication: 24 October 2025.

https://doi.org/10.1162/COLI.a.581

et al. 2024; DeepSeek-AI et al. 2025). Yet, processing non-English texts with LLMs is challenging. They suffer from tokenization overfragmentation (see Figure 2 for quantitative analysis) and thus require more inference steps due to the reliance on English-centric tokenizers and vocabulary, resulting in higher utility costs for non-English speakers (Ahia et al. 2023; Petrov et al. 2023; Ali et al. 2024).

Cross-lingual vocabulary adaptation (CVA) via vocabulary expansion has been proposed to adapt LLMs (including their tokenizers) to specific target languages (Cui, Yang, and Yao 2023; Fujii et al. 2024; Choi et al. 2024; Tejaswi, Gupta, and Choi 2024; Mundra et al. 2024, inter alia). Vocabulary expansion approaches extend the vocabulary of a source model with tokens from a target language, followed by continual pre-training on target language data. A wide range of language-specific LLMs derived from an English-centric LLM such as Llama2 (Touvron et al. 2023) have been made available following this approach, including Chinese (Cui, Yang, and Yao 2023), Tamil (Balachandran 2023), Portuguese (Larcher et al. 2023), and Japanese (Fujii et al. 2024) models, inter alia. Vocabulary expansion improves inference speed but often assumes access to a substantial amount of target language data for adaptation. For example, Chinese and Tamil Llamas make use of 20 and 12GB of target language text, respectively (Cui, Yang, and Yao 2023; Balachandran 2023). Given the guaranteed large number of model updates in high-resource settings, the embeddings of new tokens are often randomly initialized, followed by the standard continual pre-training with a causal language modeling (CLM) objective (Cui, Yang, and Yao 2023; Balachandran 2023; Larcher et al. 2023; Choi et al. 2024). However, it is not clear how effective this approach to vocabulary expansion is in low-resource settings.

In this article, we seek to (i) *answer if this widely used adaptation approach under high-resource settings is as effective in low-resource settings*; and (ii) *identify the best possible vocabulary expansion strategies for language adaptation in low-resource settings, while striving to maintain similar performance to the source model with faster inference* (Figure 1). Our key contributions are as follows:

- We present the first systematic study of vocabulary expansion-based adaptation of generative LLMs (i.e., three English-centric models) in low-resource settings (i.e., assuming only 30K sentences, ∼0.01GB text data or up to approximately 5M tokens), on two generation tasks (i.e., machine translation and summarization) and two classification tasks (i.e., multiple-choice reading comprehension and general knowledge and reasoning) across ten typologically diverse languages.
- Our results show that the popular vocabulary expansion approach in high-resource settings (i.e., random initialization and fine-tuning the full model with a CLM objective) is not always optimal in low-resource settings.
- We find that target parameter initialization approaches that use heuristics information from the source and target tokenizers are more effective. Furthermore, fine-tuning the top and bottom two layers of the LLM using a multi-token prediction objective (Gloeckle et al. 2024) works better than fine-tuning the full model with CLM. Finally, using a short input sequence length by splitting longer text into multiple sentences allows for a larger number of model updates, mitigating underfitting.

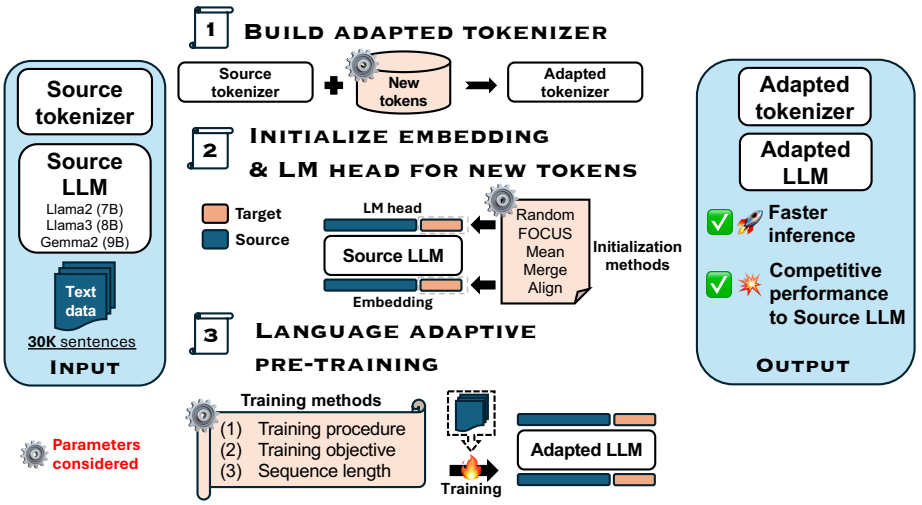

Figure 1
We address the challenge of effectively expanding vocabulary for LLMs in low-resource settings. This is crucial for reducing inference steps when generating non-English text, as LLMs often rely on English-centric tokenizers and vocabulary. Our approach explores various adaptation strategies (⚙) to achieve inference speedups while aiming to retain competitive performance. Our recommended strategy combines heuristic-based parameter initialization for new tokens with fine-tuning the top and bottom two layers of the model, using a short input sequence length and a multi-token prediction objective (Gloeckle et al. 2024).

- To better understand holistic aspects of vocabulary expansion in low-resource settings, we conduct a range of analyses, including the extent of source (English) knowledge retention and a direct comparison with vocabulary replacement.

2. Background

2.1 Text Overfragmentation

LLMs tend to overfragment text in underrepresented languages (Rust et al. 2021; Muller et al. 2021). Overfragmentation has significant implications for non-English speakers, including higher API costs (Ahia et al. 2023; Petrov et al. 2023), slower inference (Hofmann, Schuetze, and Pierrehumbert 2022; Sun et al. 2023; Petrov et al. 2023), and lower downstream performance (Bostrom and Durrett 2020; Rust et al. 2021; Toraman et al. 2023; Fujii et al. 2023; Ali et al. 2024).

To quantify this phenomenon, we calculate the average number of tokens on the FLORES-200 (NLLB Team et al. 2022) dev set across our target languages and models (Figure 2). Following the established assumption that lower average token counts indicate more efficient tokenization (Ahia et al. 2023),[2] our analysis reveals two key observations: (1) Burmese, Sinhala, and Telugu demonstrate the most severe overfragmentation

2 This assumption is valid specifically within the context of the FLORES-200 parallel machine translation corpus.

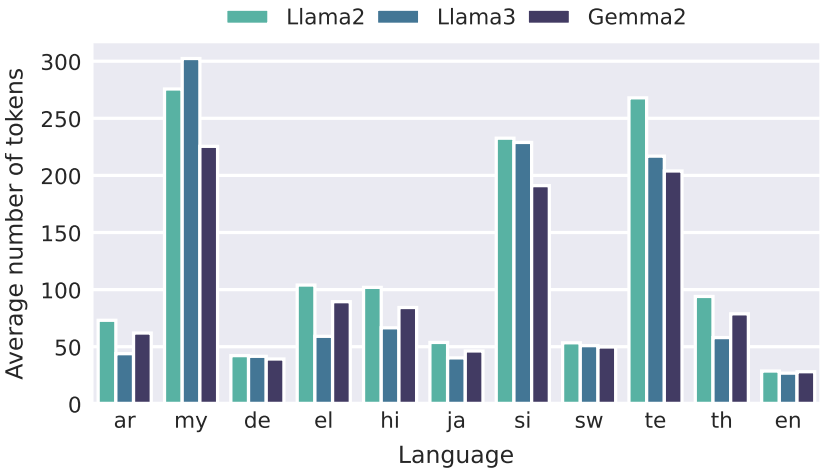

Figure 2
Average number of tokens on the FLORES-200 dev set across languages and models.

among our ten target languages, requiring at least 6.8x more tokens than English across all models, and (2) English consistently exhibits the most efficient tokenization.

2.2 Cross-lingual Vocabulary Adaptation

CVA methods have been proposed for adapting base LLMs to specific target languages for improving downstream performance and inference speed (Cui, Yang, and Yao 2023; Balachandran 2023; Fujii et al. 2024; Yamaguchi, Villavicencio, and Aletras 2024). CVA via **vocabulary expansion** incorporates new tokens into the source vocabulary (Balachandran 2023; Larcher et al. 2023; Pipatanakul et al. 2023; Lin et al. 2024; Cui, Yang, and Yao 2023; Kim, Choi, and Jeong 2024; Fujii et al. 2024; Choi et al. 2024; Nguyen et al. 2024; Tejaswi, Gupta, and Choi 2024; Mundra et al. 2024). CVA with **vocabulary replacement** replaces the entire or partial source vocabulary with a new one from the target (Ostendorff and Rehm 2023; Csaki et al. 2023; Da Dalt et al. 2024; Remy et al. 2024; Yamaguchi, Villavicencio, and Aletras 2024; Dobler and de Melo 2024; Cahyawijaya et al. 2024). More recent methods include a hypernetwork for tokenizer transfer (Minixhofer, Ponti, and Vulić 2024) and adapters for vocabulary alignment (Han et al. 2025). CVA is typically followed by continual pre-training on target language data, often called language adaptive pre-training (LAPT) (Chau, Lin, and Smith 2020).

Vocabulary expansion is widely used for developing language-specific generative LLMs from a source model, e.g., Chinese (Cui, Yang, and Yao 2023) and Tamil (Balachandran 2023) Llamas, and Swallow for Japanese (Fujii et al. 2024). This line of work assumes access to a substantial amount of target language data, e.g., 312B characters of Japanese text used for Swallow. This might not be feasible in low-resource settings with limited target language data or computing resources. To the best of our knowledge, our work is the first to investigate vocabulary expansion for efficient decoder-based LLM inference in extremely low-resource settings by assuming access to a small amount of target language data (30K sentences).

3. Problem Statement

The goal in this article is to expand the vocabulary of a source LLM to effectively support a target language in low-resource settings. This involves transitioning from a source model $\mathcal{M}_s$ (with vocabulary $\mathcal{V}_s$ and tokenizer $\mathcal{T}_s$) to a model $\mathcal{M}_t$ that supports an expanded target vocabulary $\mathcal{V}_t$ and tokenizer $\mathcal{T}_t$.

More specifically, given $\mathcal{M}_s$, $\mathcal{V}_s$, $\mathcal{T}_s$, and target language data $\mathcal{D}$, $\mathcal{M}_t$ is constructed as follows:

1. **Crafting the Auxiliary Tokenizer**: A target language-specific auxiliary tokenizer is firstly trained on $\mathcal{D}$. This tokenizer is built from scratch using $\mathcal{D}$, allowing it to capture language-specific nuances. It comes with its own vocabulary, $\mathcal{V}_{\text{aux}}$.
2. **Constructing the Target Vocabulary and Tokenizer**:
   (a) The new tokens $\mathcal{V}_{\text{new}}$ are identified by taking the top k most frequent tokens from $\mathcal{V}_{\text{aux}}$ that are *not* already present in $\mathcal{V}_s$ (i.e., $\mathcal{V}_{\text{new}} = \text{top } k \in \mathbb{N}$ tokens from $\mathcal{V}_{\text{aux}} \setminus (\mathcal{V}_s \cap \mathcal{V}_{\text{aux}})$).
   (b) The target vocabulary $\mathcal{V}_t$ is then formed by combining the original source vocabulary $\mathcal{V}_s$ with these newly identified tokens: $\mathcal{V}_t = \mathcal{V}_s \cup \mathcal{V}_{\text{new}}$.
   (c) The target tokenizer $\mathcal{T}_t$ is subsequently derived to operate on this expanded $\mathcal{V}_t$.
3. **Initializing and Adapting the Target Model**:
   (a) The target model $\mathcal{M}_t$ begins as a copy of $\mathcal{M}_s$, inheriting its identical architecture and pre-trained weights.
   (b) Its embedding and output layer matrices are then expanded to accommodate the larger $\mathcal{V}_t$. Specifically, the dimensionality of the embedding layer becomes $|\mathcal{V}_t| \times H_t$ and that of the output layer becomes $H_t \times |\mathcal{V}_t|$, where H_t is the hidden dimensionality of $\mathcal{M}_t$.
   (c) Each representation for a new token (i.e., tokens in $\mathcal{V}_{\text{new}}$) is initialized using a target parameter initialization method (detailed in §4). If the weights of both embeddings and language modeling head (i.e., output layer) are not tied, they are initialized separately.[3]
   (d) $\mathcal{M}_t$ undergoes continual pre-training (i.e., LAPT) on the target language data $\mathcal{D}$ using a causal language modeling (CLM) objective.

3 We follow the original configuration of the source model regarding weight tying for the embeddings and output layer to preserve its original behavior as closely as possible.

4. Target Parameter Initialization

After vocabulary expansion (step 2 in §3), the new embeddings should be initialized. Our aim is to investigate the effectiveness and robustness of different initialization approaches in low-resource settings. For that purpose, we evaluate: (1) random initialization; (2) initialization based on auxiliary models; and (3) heuristic-based initialization.[4]

4.1 Random Initialization

For new tokens, we randomly initialize the weights of their embeddings by sampling from $\mathcal{N}(\mu, \sigma^2)$. Here, μ and σ are the mean and standard deviation of the token embeddings from $\mathcal{M}_s$ (**Random**). This is the most simple and common approach when adapting English-centric LLMs to a target language via vocabulary expansion in high-resource settings (Cui, Yang, and Yao 2023; Balachandran 2023; Larcher et al. 2023; Choi et al. 2024).

4.2 Initialization Based on Auxiliary Models

FOCUS (Dobler and de Melo 2023) is a state-of-the-art CVA method that relies on auxiliary embeddings, i.e., fastText (Bojanowski et al. 2017), for initialization.[5] The main assumption is that semantic transfer of embeddings should result in better initialization over Random. We apply FOCUS by tokenizing $\mathcal{D}$ using $\mathcal{T}_t$ and train a fastText model for each language.[6]

4.3 Heuristic-based Initialization

Sophisticated methods such as FOCUS require auxiliary embeddings trained in the target language, which might not be available or hard to train in low-resource settings. Motivated by this, we evaluate a set of heuristic-based initialization methods that do not rely on any external data or model and can be applied to any language.

Mean. A straightforward approach is to initialize the weights of each new token in $\mathcal{V}_{new}$ by averaging the weights of their corresponding source tokens, which are identified using $\mathcal{T}_s$ (Yao et al. 2021). Koto, Lau, and Baldwin (2021) and Gee et al. (2022) have also followed a similar approach for vocabulary replacement in domain adaptation. More recently, Tejaswi, Gupta, and Choi (2024) and Mundra et al. (2024) have demonstrated the effectiveness of this mean initialization for vocabulary expansion under high-resource settings (i.e., at least 200M tokens and 2.5B tokens, respectively). However, our study specifically investigates low-resource settings with only 30K sentences, equivalent to at most 5M tokens.

4 Our primary focus in the subsequent discussion is on embedding initialization for simplicity. However, if the weights of the embeddings and the language modeling head are not tied, the language modeling head should also be initialized independently, using the identical process applied to the embeddings.

5 Due to resource constraints, we only report results using FOCUS as a representative initialization method that relies on auxiliary embeddings since it outperforms other similar methods such as WECHSEL (Minixhofer, Paischer, and Rekabsaz 2022) for vocabulary replacement.

6 Preliminary experiments with FOCUS using off-the-shelf pre-trained word-level fastText models yielded lower performance.

Merge. Mean uses solely $\mathcal{T}_s$, which might produce subtokens from $\mathcal{V}_s$ that are not semantically related to the new target token. To overcome this issue, we propose using merge rules from $\mathcal{T}_t$ to effectively initialize $\mathcal{V}_{new}$. Merge rules describe how $\mathcal{T}_t$ can combine two subtokens into one. Through these rules, each new token in $\mathcal{V}_{new}$ can be decomposed into several existing subtokens from $\mathcal{V}_s$.

For instance, consider a new token: 'superherohype' in $\mathcal{V}_{new}$. According to the merge rules in $\mathcal{T}_t$, it can first be decomposed into ('superhero', 'hype'). If 'superhero' is also a new token, it can be further decomposed into ('super', 'hero'). This process continues until all constituent parts are tokens from $\mathcal{V}_s$ (in this case, 'super', 'hero', and 'hype'). We hypothesize that such hierarchically derived subtokens from $\mathcal{V}_s$ are more semantically related to the new token than those obtained by simple averaging, as they leverage the specific tokenization information embedded in $\mathcal{T}_t$.

The initialization process is as follows:

1. We identify all merge rules in $\mathcal{T}_t$ that result in a token found in $\mathcal{V}_{new}$.
2. Using these identified merge rules, we generate a hierarchical mapping for each new target token down to its constituent source subtokens in $\mathcal{V}_s$.
3. For each new token, we compute the hierarchical mean of the embeddings of its associated source subtokens, guided by the mapping information.

Align. Initializing the weights of new tokens in $\mathcal{V}_t$ by simply averaging the weights of their constituent subtokens from $\mathcal{T}_s$ as in Mean can be suboptimal. This naive approach does not account for how tokenization might change in a full sequence, as opposed to a single token.

Consider the word: '_cup' and a new token: '_cu'.[7] While Mean averages the weights of '_c' and 'u' from $\mathcal{T}_s$ to initialize that of '_cu', this can be suboptimal. If $\mathcal{T}_s$ tokenizes '_cup' as '_c' + 'up', but $\mathcal{T}_t$ tokenizes it as '_cu' + 'p', '_cu' functions as a new, distinct subword unit. Its weight initialization should reflect this new role, perhaps focusing more on relevant overlapping components like '_c'.

To obtain a more fine-grained semantic representation of a new token, we propose token alignment initialization that leverages mapping information between tokens tokenized with $\mathcal{T}_s$ and $\mathcal{T}_t$ and the mapping frequency (i.e., more frequent mappings are more important for the final representation). This process allows us to consider different tokenization variants for the initialization of a given token.

Specifically, the weights of each new token $t \in \mathcal{V}_{new}$ are initialized as follows.

1. **Generate Mappings from $\mathcal{D}$**:
   (a) For each sentence $x \in \mathcal{D}$, we first tokenize it using both $\mathcal{T}_s$ and $\mathcal{T}_t$.
   (b) We then compare these two tokenized versions to identify how each new token t in $\mathcal{T}_t$'s output corresponds to

7 '_' stands for a whitespace.

sequences of subtokens from $\mathcal{T}_s$'s output. These correspondences are stored as a list of tuples, e.g., a token t from $\mathcal{V}_t$ might map to $[(t_1, t_2), (t_3, t_4)]$ from $\mathcal{V}_s$ in different contexts.

2. **Construct Unique Mappings and Frequencies**:

   (a) We concatenate all such mapping lists from $\mathcal{D}$ for each new token t.

   (b) From this combined list, we construct a unique set of constituent tuples for t, denoted as $\mathbf{T}_t = (\mathbf{t}_1, \mathbf{t}_2, \mathbf{t}_3, \ldots)$, where each $\mathbf{t}_i$ is a sequence of subtokens from $\mathcal{V}_s$.

   (c) We also create a corresponding frequency list $\mathbf{F}_t = (f_{\mathbf{t}_1}, f_{\mathbf{t}_2}, f_{\mathbf{t}_3}, \ldots)$, indicating how often each unique tuple $\mathbf{t}_i$ appeared in the mappings for token t on $\mathcal{D}$.

3. **Compute Initial Representation**: Finally, using $\mathbf{T}_t$ and $\mathbf{F}_t$, we aggregate the mapping information to generate the representation for token t by computing $\sum_{\mathbf{t} \in \mathbf{T}_t} \left[f_{\mathbf{t}} \frac{1}{|\mathbf{t}|} \sum_{t' \in \mathbf{t}} \mathbf{e}_{t'} \right]$. Here, $\mathbf{e}_{t'}$ is the embedding of the subtoken $t' \in \mathcal{V}_s$, $f_{\mathbf{t}}$ is the frequency of mapping $\mathbf{t}$, and $|\mathbf{t}|$ is the number of subtokens in $\mathbf{t}$.

5. Training Strategy

Continual pre-training on $\mathcal{D}$ (i.e., LAPT) is an integral part of vocabulary expansion to improve the alignment of the newly initialized embeddings. To this end, we explore: (i) the training procedure, (ii) the objective function, and (iii) the input sequence length.

Training Procedure.

- **LoRA**: We use by default low-rank adaptation (**LoRA**) (Hu et al. 2022) applied to all linear layers, following previous work on vocabulary expansion in high-resource settings (Cui, Yang, and Yao 2023; Balachandran 2023; Abbasi et al. 2023; Choi et al. 2024).
- **2-stage**: We also consider the two-stage tuning process (**2-stage**) (Cui, Yang, and Yao 2023), where only the embeddings and language modeling head are first updated, followed by tuning the LoRA modules. This process can help minimize the risk of overfitting to the initial embedding state, which might be suboptimal (Downey et al. 2023).
- **2×2 LS**: We train only the top and bottom two layers (**2×2 LS**) of the model, following Remy et al. (2024). This calibrates only the parts closely related to the encoding and decoding of the target language (Wendler et al. 2024; Tang et al. 2024), minimizing changes to the source model.

Note that none of these approaches have ever been compared to the standard LoRA or against each other. We tune the embeddings and language modeling head in each case.

Objective Function.

- **CLM**: We first evaluate a causal language modeling objective (CLM), which has been used by previous work in high-resource settings (Cui, Yang, and Yao 2023; Balachandran 2023; Larcher et al. 2023, inter alia), as a default training objective.
- **MTP**: We also consider a multi-token prediction (MTP) objective (Gloeckle et al. 2024), where the model must predict multiple consecutive tokens at each timestep rather than a single token. MTP has been proposed for pre-training and exhibited performance gains over CLM. However, it has yet to be explored for continual pre-training for cross-lingual transfer. We experiment with one additional language modeling head (i.e., predicting two consecutive tokens at a time) and initialize the additional weights with those from the original language modeling head.[8]

Input Sequence Length. We shorten the default input sequence length from **2,048** to **512**, thereby increasing the number of training batches. We hypothesize that this is critical in low-resource settings, as a small number of model updates could be prone to underfitting.

6. Experimental Setup

6.1 Source Models

We use **Llama2** 7B (Touvron et al. 2023), an English-centric, non-instruction-tuned model, with its $\mathcal{T}_s$ based on byte-fallback Byte Pair Encoding (BPE) (Sennrich, Haddow, and Birch 2016) and $|\mathcal{V}_s|$ set to 32K in the experiments. We also use **Llama3** 8B (Dubey et al. 2024) and **Gemma2** 9B (Riviere et al. 2024) for consistency in our analysis. These models have a far larger vocabulary size than Llama2, i.e., 128K and 256K, respectively. Note that Llama2 and Llama3 have untied embedding and language modeling head weights, while the original configuration of Gemma2 includes weight tying.

6.2 Target Languages and Data

We experiment with a typologically diverse set of ten languages with various scripts. This includes German (Indo-European) and Swahili (Niger–Congo) for the Latin script, and Arabic (Afroasiatic), Burmese (Sino-Tibetan), Greek (Indo-European), Hindi (Indo-European), Japanese (Japonic), Sinhala (Indo-European), Telugu (Dravidian), and Thai (Kra–Dai) for non-Latin scripts. We select these languages because of the availability of downstream task datasets with the same task formulation across languages, with a particular focus on generation tasks.

8 We find in our preliminary analysis that random initialization does not work well.

To simulate a realistic low-resource adaptation scenario, we follow Yong et al. (2023) in using 30K sentences per language. With $|\mathcal{D}|$ set to 30K, this equates to up to approximately 5M tokens. These sentences are randomly sampled from their language-specific subcorpus of CC-100 (Conneau et al. 2020). Note that previous work on vocabulary expansion for generative LLMs (Tejaswi, Gupta, and Choi 2024; Mundra et al. 2024) uses at least 200M tokens and 2.5B tokens respectively, which is at least 40 times larger than our training budget.[9]

6.3 Baselines

We use the following two methods as our baselines:

1. **Source**: We use the off-the-shelf source base (i.e., non-instruction-tuned) model $\mathcal{M}_s$ without any adaptation, following Tejaswi, Gupta, and Choi (2024). This provides a crucial reference point, allowing us to quantify the inherent performance in the target language before language-specific tuning, and thereby clearly measure specific gains from vocabulary expansion and LAPT.
2. **CPT-Only**: We continue pre-training (CPT) *Source* on $\mathcal{D}$, retaining its original vocabulary $\mathcal{V}_s$. This differs from vocabulary expansion approaches, which expand the original vocabulary $\mathcal{V}_s$ to include new terms $\mathcal{V}_t = \mathcal{V}_s \cup \mathcal{V}_{new}$ as mentioned in §3.

It is important to emphasize that these baselines do not offer any inference speedups.

6.4 Evaluation

Tasks. We use both generation and classification target language tasks to evaluate each approach. For generation tasks, we use (1) English-to-target machine translation (MT) using FLORES-200 (NLLB Team et al. 2022) and (2) summarization (SUM) including German MLSUM (Scialom et al. 2020), GreekSUM (Evdaimon et al. 2024), and XL-Sum (Hasan et al. 2021) for the rest. Note that SUM performance is not directly comparable between different languages as the data does not match across languages. For classification tasks, we use multiple-choice reading comprehension (MC) using Belebele (Bandarkar et al. 2024) and Global MMLU (GMMLU) (Singh et al. 2025) as a general knowledge and reasoning benchmark. Note that GMMLU does not support Burmese and Thai.

Number of Samples. Following Ahia et al. (2023), we use 500 random samples for generation tasks (SUM and MT). MC and GMMLU use their full test sets for evaluation. Specifically, MC has 900 samples per language subset, while GMMLU has 14K samples.

9 For additional context, the BabyLM challenge (Warstadt et al. 2023; Hu et al. 2024), where participants must pre-train a model from scratch under low-resource settings, utilizes training budgets of up to 100M words, motivated by the observation that children are exposed to fewer than 100M words by 13 years of age. While the BabyLM challenge is a challenging task, our setting, with access to less than 5M tokens, represents an even more extreme low-resource scenario, posing a distinct challenge for effective target language adaptation.

Prompt Templates. For SUM prompt templates, we translate the English templates from Ahia et al. (2023) using a machine translation API, as in Yong et al. (2023). For MT, we create an English template and then translate it into each target language. For MC and GMMLU, we follow the default template provided by HuggingFace LightEval (Habib et al. 2023). The complete prompt templates are listed in Table A.1 in the Appendix.

6.5 Evaluation Metrics

Task Performance. We use accuracy for MC and GMMLU, chrF (Popović 2015) for MT, and ROUGE-L (Lin 2004) for SUM. In the analysis (§7.3), we also use BLEURT (Sellam, Das, and Parikh 2020) as an auxiliary metric for SUM.

We report average zero-shot performance across five different runs for the generation tasks, namely SUM and MT. For the classification tasks, we report single-run three-shot performance for MC and five-shot performance for GMMLU as these tasks are deterministically evaluated with temperature set to zero.

Perplexity. We report perplexity on 100K language-specific held-out CC-100 sentences as an auxiliary metric for evaluating model performance.

Inference Efficiency. We measure inference efficiency as the number of tokens generated per second (Hong, Lee, and Cho 2024).

6.6 Implementation Details

Hyperparameters. We set $|\mathcal{V}_{\text{aux}}|$ to 50K across languages and the number of new target tokens $|\mathcal{V}_{\text{new}}|$ to 100 by default. We investigate the effect of varying $|\mathcal{V}_{\text{new}}|$ in §7.4. For LAPT, we train each model for two epochs with a batch size of 8, a maximum learning rate of 1e-4, and a sequence length of 2,048. We set a LoRA rank to 8 following previous work (Cui, Yang, and Yao 2023; Abbasi et al. 2023; Lin et al. 2024). Table A.2 in the Appendix details the hyperparameter configurations utilized during both the training and inference phases.

To make a fair comparison, we do not conduct any parameter tuning and use the same ones across all approaches. For SUM, we truncate an article whenever it exceeds the maximum prompt length of 4,096 to avoid the CUDA out-of-memory error.

Evaluation Metrics Computation. To compute ROUGE-L, we split sentences with an mT5 (Xue et al. 2021) tokenizer as preprocessing following Maynez, Agrawal, and Gehrmann (2023) and subsequently call `rouge_scorer`[10] to compute the metric. We use BLEURT-20 to compute BLEURT.[11]

Libraries and Hardware. For Llama2 and Gemma2, we train tokenizers using SentencePiece (Kudo and Richardson 2018) and convert them into the HuggingFace Tokenizers (Moi and Patry 2023) format. For Llama3, we train tokenizers using HuggingFace Tokenizers. We implement our models using PyTorch (Paszke et al. 2019), HuggingFace Transformers (Wolf et al. 2020) and PEFT (Mangrulkar et al. 2022). We preprocess

10 https://github.com/csebuetnlp/xl-sum/tree/master/multilingual_rouge_scoring.
11 https://github.com/google-research/bleurt.

Table 1
Mean performance on generation tasks (MT and SUM) over five runs in low-resource settings (30K sentences) using Llama2 as source. Green shades indicate positive performance change over Source per language and task, respectively.

		Arabic		Burmese		German		Greek		Hindi		Japanese		Sinhala		Swahili		Telugu		Thai	
		Afroasiatic		Sino-Tibetan		Indo-European		Indo-European		Indo-European		Japonic		Indo-European		Niger–Congo		Dravidian		Kra–Dai	
	Model	MT	SUM	MT	SUM	MT	SUM	MT	SUM	MT	SUM	MT	SUM	MT	SUM	MT	SUM	MT	SUM	MT	SUM
	Source	$.08_{.03}$	$36_{0.0}$	$.03_{.00}$	$24_{0.1}$	$.25_{.01}$	$25_{0.1}$	$.08_{.03}$	$25_{0.2}$	$.12_{.03}$	$39_{0.2}$	$.03_{.03}$	$22_{0.1}$	$.05_{.00}$	$26_{0.1}$	$.01_{.00}$	$27_{0.4}$	$.06_{.00}$	$21_{0.2}$	$.06_{.02}$	$22_{0.2}$
	CPT-only	$.18_{.00}$	$36_{0.2}$	$.11_{.00}$	$24_{0.3}$	$.20_{.00}$	$21_{0.1}$	$.21_{.00}$	$25_{0.3}$	$.21_{.00}$	$38_{0.2}$	$.08_{.00}$	$23_{0.2}$	$.07_{.00}$	$23_{0.1}$	$.12_{.00}$	$30_{0.1}$	$.09_{.00}$	$29_{0.1}$	$.12_{.00}$	$23_{0.2}$
+Speedup	Random	$.07_{.00}$	$32_{0.3}$	$.05_{.00}$	$17_{0.3}$	$.28_{.00}$	$23_{0.3}$	$.13_{.00}$	$20_{0.1}$	$.09_{.00}$	$35_{0.2}$	$.18_{.00}$	$22_{0.2}$	$.07_{.00}$	$29_{0.1}$	$.13_{.00}$	$29_{0.1}$	$.06_{.00}$	$26_{0.2}$	$.07_{.00}$	$19_{0.2}$
	FOCUS	$.07_{.00}$	$32_{0.2}$	$.03_{.00}$	$16_{0.3}$	$.28_{.00}$	$22_{0.4}$	$.17_{.00}$	$18_{0.1}$	$.10_{.00}$	$34_{0.2}$	$.19_{.00}$	$22_{0.2}$	$.05_{.00}$	$29_{0.3}$	$.18_{.00}$	$29_{0.1}$	$.05_{.00}$	$26_{0.2}$	$.06_{.00}$	$17_{0.1}$
	Mean	$.06_{.00}$	$33_{0.1}$	$.04_{.00}$	$20_{0.4}$	$.24_{.01}$	$23_{0.2}$	$.14_{.00}$	$19_{0.1}$	$.12_{.00}$	$34_{0.2}$	$.20_{.00}$	$23_{0.1}$	$.06_{.00}$	$30_{0.3}$	$.12_{.00}$	$29_{0.1}$	$.06_{.00}$	$27_{0.1}$	$.09_{.00}$	$22_{0.1}$
	Merge	$.07_{.00}$	$33_{0.1}$	$.04_{.00}$	$9_{0.5}$	$.25_{.00}$	$23_{0.2}$	$.15_{.00}$	$18_{0.3}$	$.10_{.00}$	$34_{0.2}$	$.18_{.00}$	$23_{0.3}$	$.09_{.00}$	$30_{0.2}$	$.13_{.00}$	$29_{0.1}$	$.05_{.00}$	$26_{0.3}$	$.07_{.00}$	$21_{0.2}$
	Align	$.06_{.00}$	$33_{0.1}$	$.04_{.00}$	$15_{0.7}$	$.26_{.01}$	$21_{0.2}$	$.16_{.00}$	$17_{0.2}$	$.13_{.00}$	$35_{0.2}$	$.20_{.00}$	$23_{0.1}$	$.08_{.00}$	$30_{0.3}$	$.17_{.00}$	$29_{0.0}$	$.06_{.00}$	$27_{0.2}$	$.07_{.00}$	$22_{0.2}$

datasets with HuggingFace Datasets (Lhoest et al. 2021). For evaluation, we use HuggingFace LightEval (Habib et al. 2023). We use either four NVIDIA V100 (32GB) or a single A100 (80GB) for LAPT. Evaluation utilizes a single NVIDIA V100 (32GB) or A100 (80GB). Each analysis utilizes a consistent hardware configuration to ensure accurate measurement of inference efficiency.

7. Results and Analysis

7.1 Target Parameter Initialization

We analyze the effect of different target parameter initialization methods (§4) on task performance and inference efficiency using Llama2 as source.

7.1.1 Task Performance and Perplexity. Table 1 shows the performance of all methods on generation tasks, while Table 2 shows the corresponding perplexities on the held-out language-specific dataset.

Performance on Generation Tasks. Models initialized with Mean and Align generally exhibit performance comparable to or better than Source. Specifically, they outperform (by

Table 2
Perplexity on language-specific held-out dataset using Llama2 as source. Note that results are not comparable between models with gray and others due to their difference in vocabulary. **Bold** and underlined indicate the best and second-best perplexities among adapted models for each language.

Model	**ar**	**my**	**de**	**el**	**hi**	**ja**	**si**	**sw**	**te**	**th**
Source	8.3	5.0	35.7	4.8	6.4	20.4	3.5	47.2	2.4	9.4
CPT-only	4.2	2.7	10.9	2.9	3.2	5.4	2.3	12.7	1.8	4.3
Random	11.9	11.9	12.0	7.1	8.3	15.0	8.7	13.9	7.1	8.8
FOCUS	11.5	13.8	12.1	6.6	8.9	14.8	9.3	13.7	7.9	9.4
Mean	9.4	11.6	**11.7**	6.0	6.4	**14.7**	8.1	**13.5**	6.9	7.8
Merge	9.8	12.6	11.8	6.1	6.7	15.1	8.7	13.7	7.3	8.0
Align	**9.3**	**11.2**	**11.7**	**5.9**	**6.3**	15.1	**8.0**	**13.5**	**6.7**	**7.6**

>2 points) or match (within 2 points) Source in 16 and 15 out of 20 cases, respectively. Mean shows a positive gain over Source in 10 cases, while Align achieves this in 12 cases. These results align with their perplexity scores (Table 2), where Align generally yields the lowest perplexity across languages, closely followed by Mean. While Merge also matches or surpasses Source performance in 16 cases (with 10 positive gains), it often shows the largest perplexities among the three heuristic-based initialization methods. We speculate that Merge might be less informative than these two methods, as it does not rely on surface information for initialization, resulting in slightly larger perplexity.

The baseline Random and FOCUS models perform similarly (within 2 points) to or better than Source in 14 cases, with 10 and 7 cases showing positive gains, respectively. These results place them among the least effective approaches. FOCUS, in particular, shows the fewest cases with positive gains and the worst perplexity in six languages (i.e., Burmese, German, Hindi, Sinhala, Telugu, and Thai). These findings suggest that sophisticated initialization methods do not always guarantee superior performance, possibly due to underfitting of an auxiliary embedding model. Further, the popular Random approach is not always optimal in low-resource scenarios. Thus, we conclude that *models initialized with Mean and Align are more likely to perform competitively with Source on generation tasks.*

Analyzing performance by language and task, Mean and Align models show positive gains over Source in six and eight out of ten languages for MT, respectively. In cases where gains are not observed, any performance degradation is negligible (within 2 points). Notably, they substantially outperform Source in Greek, Japanese, and Swahili MT, with improvements ranging from 8 to 17 points.

However, for SUM, these adapted models often do not yield a performance gain over Source in the majority of the languages, except for Japanese, Sinhala, Swahili, and Telugu. In particular, Burmese and Greek show a substantial performance drop of up to 9 and 8 points, respectively. We hypothesize that SUM may necessitate more training data than MT because generating longer text (up to 128 tokens, using long context) requires models to have strong generative capabilities in the target language. We later address this challenge in §7.2 by showing that these performance gaps can be drastically narrowed using our alternative training strategies, further contributing to the competitiveness of vocabulary expansion approaches to Source.

Turning to CPT-only (i.e., continual pre-training without vocabulary expansion), it demonstrates strong performance, matching or improving upon Source in 17 out of 20 cases, a higher count by at least one case than achieved by the models with the heuristic-based initialization. A direct comparison against the best-performing Align[12] further confirms this. CPT-only outperforms Align in 10 out of 20 cases, whereas Align only wins in four. These results demonstrate the overall superiority of CPT-only over vocabulary expansion approaches. This trend aligns with previous work on CVA (Downey et al. 2023; Yamaguchi, Villavicencio, and Aletras 2024) suggesting that CPT-only can often perform better than vocabulary expansion approaches in low-resource settings, possibly due to its reliance on robust and well-aligned original embeddings.

Performance on Classification Tasks. Table 3 presents the performance of all methods on classification tasks. A distinct trend emerges compared to generation tasks: *adapted*

12 It is best-performing because it generally achieves the lowest perplexities across languages and is the most likely among different initialization methods to provide a positive gain over Source.

Table 3
Mean performance on classification tasks (MC and GMMLU) in low-resource settings (30K sentences) using Llama2 as source. Green shades indicate positive performance change over Source per language and task, respectively. Note that GMMLU does not support Burmese and Thai.

		Arabic		Burmese		German		Greek		Hindi		Japanese		Sinhala		Swahili		Telugu		Thai	
		MC	GMMLU	MC	GMMLU	MC	GMMLU	MC	GMMLU	MC	GMMLU	MC	GMMLU	MC	GMMLU	MC	GMMLU	MC	GMMLU	MC	GMMLU
	Source	.29	.29	.26	–	.43	.39	.27	.28	.25	.28	.40	.33	.24	.27	.31	.28	.28	.27	.29	–
	CPT-only	.30	.29	.22	–	.42	.38	.29	.29	.28	.28	.39	.34	.23	.27	.33	.27	.24	.27	.28	–
+Speedup	Random	.28	.25	.22	–	.42	.38	.22	.27	.27	.26	.35	.30	.22	.27	.29	.25	.28	.26	.28	–
+Speedup	FOCUS	.29	.26	.28	–	.41	.38	.24	.26	.28	.26	.36	.31	.29	.26	.28	.25	.24	.27	.28	–
+Speedup	Mean	.28	.25	.23	–	.42	.38	.30	.26	.27	.26	.38	.29	.28	.26	.29	.25	.23	.27	.30	–
+Speedup	Merge	.29	.25	.25	–	.43	.38	.27	.28	.26	.28	.37	.29	.27	.26	.28	.25	.28	.26	.29	–
+Speedup	Align	.28	.25	.26	–	.40	.37	.31	.26	.27	.27	.37	.28	.27	.27	.29	.25	.22	.27	.31	–

models rarely yield a positive performance gain over Source. While all vocabulary expansion approaches exhibit competitive or better performance than Source in at least 12 cases, their positive gains are limited to a maximum of four cases (22%) (Mean and Align). This is far fewer than the maximum of 12 cases (60%) observed on the generation tasks.

Notably, CPT-only performs on par with or outperforms Source in 16 out of 18 cases. Nonetheless, this contrasts with its strong performance on generation tasks: It shows positive gains in only 6 out of 18 cases (33%), compared to 65% on generation tasks. These results suggest that while continual pre-training (i.e., LAPT) can offer benefits, its impact on discriminative tasks in low-resource settings is limited and differs from its effect on generation tasks.

We hypothesize these differences arise because our adaptation process (LAPT) primarily optimizes a causal language modeling objective on unlabeled target language data $\mathcal{D}$. While this objective is highly beneficial for generative tasks requiring fluency and extended text production, it may not directly translate to immediate gains on classification tasks. Classification tasks, in contrast, often heavily depend on the inherent semantic and factual knowledge of a model, frequently requiring only a single token generation for prediction. Therefore, while previous work on vocabulary expansion with LAPT (Tejaswi, Gupta, and Choi 2024; Cui, Yang, and Yao 2023; Balachandran 2023; Choi et al. 2024; Yamaguchi et al. 2025) often reports performance improvements in both generation and classification tasks in resource-rich settings, our findings suggest that under extremely low-resource settings (i.e., 30K sentences per language, approximately up to 5M tokens), LAPT does not inherently guarantee an improvement in these discriminative capabilities.

7.1.2 Inference Efficiency. Table 4 shows the inference efficiency of adapted models across tasks and languages. We first see that FOCUS tends to exhibit the worst speedups across languages in SUM, followed by Merge. Specifically, its speedups are the worst in 8 out of 10 languages. These results are consistent with downstream performance and may be due to an inability to effectively generate newly added target tokens, which is essential for improving inference efficiency in a target language. For MT, we generally see smaller differences between models since its input is mostly in English, and thus, only the output contributes to inference speedups. Similarly, for classification tasks (MC and GMMLU), the observed speedups are generally similar across models. This is primarily

Table 4
Inference efficiency for: (a) generation tasks (MT and SUM) over five runs; and (b) classification tasks (MC and GMMLU). We measure inference efficiency by the number of tokens generated per second (T/S). The Source row shows both raw T/S and baseline 1.00x speedup per language and task. **Bold** and underlined indicate the best and second-best speedups across tasks and models with the same base model for each language.

(a) Generation tasks

Model	Arabic		Burmese		German		Greek		Hindi		Japanese		Sinhala		Swahili		Telugu		Thai	
	MT	SUM	MT	SUM	MT	SUM	MT	SUM	MT	SUM	MT	SUM	MT	SUM	MT	SUM	MT	SUM	MT	SUM
Source	42.2	20.1	28.6	15.0	42.6	24.8	42.4	13.3	42.2	21.5	42.8	21.9	28.1	14.8	43.0	24.1	27.7	10.0	42.6	17.9
	1.00x	1.00x	1.00x	1.00x	**1.00**x	1.00x	1.00x	1.00x	1.00x	1.00x	1.00x	1.00x	1.00x	1.00x	**1.00**x	1.00x	1.00x	1.00x	1.00x	1.00x
CPT-only	1.01x	1.13x	0.98x	1.01x	0.97x	**1.05**x	1.00x	0.97x	1.00x	1.01x	0.98x	1.03x	1.00x	1.00x	0.98x	1.21x	0.98x	0.99x	0.98x	1.01x
Random	**1.17**x	1.88x	**1.18**x	1.48x	0.95x	0.97x	1.32x	**1.79**x	**1.19**x	1.57x	1.04x	**1.13**x	1.56x	2.54x	0.92x	**1.27**x	**1.93**x	4.26x	1.06x	1.55x
FOCUS	1.14x	1.79x	1.13x	1.38x	0.89x	0.94x	**1.41**x	1.66x	1.11x	1.49x	1.04x	1.11x	1.13x	2.40x	0.93x	1.26x	1.79x	4.26x	1.06x	1.44x
Mean	1.08x	**1.99**x	1.02x	**2.15**x	0.95x	0.95x	1.32x	1.73x	1.15x	1.70x	1.08x	**1.13**x	1.29x	**2.67**x	0.91x	1.25x	1.84x	**4.58**x	**1.10**x	1.62x
Merge	**1.17**x	1.95x	1.08x	1.20x	0.94x	1.00x	1.40x	1.70x	1.07x	1.57x	1.05x	1.12x	**1.58**x	2.49x	0.92x	1.25x	1.71x	4.38x	1.05x	1.59x
Align	1.10x	1.98x	1.08x	1.90x	0.95x	1.04x	1.37x	1.67x	1.15x	**1.77**x	**1.09**x	**1.13**x	1.47x	2.58x	0.93x	1.26x	1.84x	4.42x	1.08x	**1.63**x

(b) Classification tasks

Model	Arabic		Burmese		German		Greek		Hindi		Japanese		Sinhala		Swahili		Telugu		Thai	
	MC	GMMLU	MC	GMMLU	MC	GMMLU	MC	GMMLU	MC	GMMLU	MC	GMMLU	MC	GMMLU	MC	GMMLU	MC	GMMLU	MC	GMMLU
Source	19.4	29.9	11.1	–	36.6	34.3	14.0	23.4	16.9	24.1	33.4	34.0	10.9	15.2	35.4	33.1	11.1	12.8	16.7	–
	1.00x	1.00x	1.00x	–	1.00x	1.00x	1.00x	1.00x	1.00x	1.00x	1.00x	**1.00**x	1.00x	1.00x	1.00x	1.00x	1.00x	1.00x	1.00x	–
CPT-only	1.17x	0.98x	1.24x	–	**1.04**x	0.97x	1.22x	0.92x	1.17x	0.99x	**1.03**x	0.93x	1.27x	1.10x	1.04x	1.02x	1.02x	1.11x	1.20x	–
Random	**1.71**x	**1.12**x	2.12x	–	1.03x	1.01x	**1.68**x	**1.29**x	**1.71**x	**1.25**x	1.02x	0.99x	2.10x	1.92x	1.05x	1.01x	**1.84**x	2.10x	**1.42**x	–
FOCUS	**1.71**x	**1.12**x	**2.13**x	–	0.99x	1.01x	**1.68**x	**1.29**x	**1.71**x	1.23x	1.02x	0.99x	**2.11**x	1.92x	1.05x	1.00x	**1.84**x	**2.11**x	**1.42**x	–
Mean	**1.71**x	1.10x	**2.13**x	–	1.02x	1.01x	**1.68**x	**1.29**x	**1.71**x	**1.25**x	**1.03**x	**1.00**x	2.10x	1.92x	1.05x	**1.03**x	**1.84**x	**2.11**x	**1.42**x	–
Merge	**1.71**x	**1.12**x	2.12x	–	1.02x	0.99x	**1.68**x	**1.29**x	**1.71**x	**1.25**x	1.02x	0.99x	2.10x	**1.93**x	**1.06**x	1.01x	**1.84**x	2.10x	**1.42**x	–
Align	**1.71**x	1.11x	2.12x	–	1.03x	1.02x	**1.68**x	**1.29**x	**1.71**x	**1.25**x	**1.03**x	0.98x	2.10x	1.92x	**1.06**x	**1.03**x	**1.84**x	2.10x	**1.42**x	–

because these tasks involve processing a long input context, and their output typically requires only a single token, limiting the room for producing the difference between different approaches.

Examining speedups by language and script, most non-Latin script languages (Arabic, Burmese, Hindi, Sinhala, Telugu, and Thai) exhibit substantial inference speedups, reaching up to 4.58×. In contrast, Japanese shows only moderate speedups, up to 1.13×. This is likely due to its prior representation in the Llama2 training corpus, which includes Japanese (0.1%) and Chinese (0.13%) data (Touvron et al. 2023), with Chinese sharing common scripts (i.e., *kanji*). Similarly, German shows negligible speedups across tasks. This can be attributed to its largest non-English language representation (0.17%) in the Llama2 training corpus, implying its vocabulary is already well-covered. As a Latin script language, German is also inherently well-tokenized by the primarily Latin-script-based vocabulary of Llama2. This further suggests that expanding the vocabulary of a source model with a small number of target language tokens (i.e., 100 in Table 4) does not substantially accelerate inference if the language is already well-represented in the vocabulary of the base model. Figure 2 further supports this trend, as Japanese and German both show far less text tokenization overfragmentation for Llama2 compared to Burmese, Greek, Hindi, Sinhala, Telugu, and Thai. Despite not being explicitly included in the Llama2 training corpus, Swahili shows limited speedups. This might be because its Latin script aligns well with the predominant Latin script pre-training data (91%), meaning its tokenization is already relatively efficient even without explicit target vocabulary expansion.

Recommendation. Using Mean or Align can provide better downstream performance on generation tasks and inference speedups across tasks. Vocabulary expansion contributes

Table 5
Mean performance of Align models on generation tasks over five runs with Llama2 as source. Green indicates positive performance change over Source.

		Arabic		Burmese		Greek		Hindi		Sinhala		Telugu	
	Model	MT	SUM	MT	SUM	MT	SUM	MT	SUM	MT	SUM	MT	SUM
	Source	$.08_{.03}$	$36_{0.0}$	$.03_{.00}$	$24_{0.1}$	$.08_{.03}$	$25_{0.2}$	$.12_{.03}$	$39_{0.2}$	$.05_{.00}$	$26_{0.1}$	$.06_{.00}$	$21_{0.2}$
	CPT-only	$.18_{.00}$	$36_{0.2}$	$.11_{.00}$	$24_{0.2}$	$.21_{.00}$	$25_{0.3}$	$.21_{.00}$	$38_{0.2}$	$.07_{.00}$	$23_{0.1}$	$.09_{.00}$	$29_{0.1}$
LoRA	CLM+2048	$.06_{.00}$	$33_{0.1}$	$.04_{.00}$	$15_{0.7}$	$.16_{.00}$	$17_{0.2}$	$.13_{.00}$	$35_{0.2}$	$.08_{.00}$	$30_{0.3}$	$.06_{.00}$	$27_{0.2}$
LoRA	MTP+2048	$.11_{.00}$	$32_{0.2}$	$.03_{.00}$	$14_{0.7}$	$.14_{.00}$	$20_{0.2}$	$.12_{.00}$	$34_{0.2}$	$.08_{.00}$	$29_{0.1}$	$.04_{.00}$	$25_{0.1}$
LoRA	CLM+512	$.12_{.00}$	$33_{0.1}$	$.12_{.00}$	$23_{0.1}$	$.23_{.00}$	$19_{0.4}$	$.17_{.00}$	$35_{0.1}$	$.11_{.00}$	$32_{0.1}$	$.10_{.00}$	$28_{0.1}$
LoRA	MTP+512	$.14_{.00}$	$33_{0.1}$	$.12_{.00}$	$23_{0.1}$	$.21_{.00}$	$21_{0.3}$	$.15_{.00}$	$35_{0.1}$	$.11_{.00}$	$30_{0.3}$	$.09_{.00}$	$28_{0.1}$
2-stage	CLM+2048	$.09_{.00}$	$33_{0.1}$	$.04_{.00}$	$17_{0.2}$	$.13_{.00}$	$21_{0.1}$	$.13_{.00}$	$36_{0.1}$	$.09_{.00}$	$30_{0.4}$	$.05_{.00}$	$28_{0.1}$
2-stage	MTP+2048	$.13_{.00}$	$33_{0.1}$	$.02_{.00}$	$17_{0.3}$	$.19_{.00}$	$21_{0.1}$	$.11_{.00}$	$36_{0.1}$	$.08_{.00}$	$29_{0.2}$	$.03_{.00}$	$25_{0.2}$
2-stage	CLM+512	$.12_{.01}$	$33_{0.1}$	$.07_{.00}$	$13_{0.5}$	$.22_{.00}$	$22_{0.3}$	$.16_{.00}$	$35_{0.1}$	$.10_{.00}$	$31_{0.3}$	$.07_{.00}$	$28_{0.1}$
2-stage	MTP+512	$.11_{.00}$	$33_{0.2}$	$.07_{.00}$	$17_{0.2}$	$.21_{.00}$	$22_{0.1}$	$.16_{.00}$	$35_{0.1}$	$.08_{.00}$	$31_{0.3}$	$.07_{.00}$	$28_{0.1}$
2×2 LS	CLM+2048	$.11_{.01}$	$33_{0.1}$	$.14_{.00}$	$25_{0.1}$	$.24_{.00}$	$22_{0.3}$	$.15_{.00}$	$36_{0.2}$	$.11_{.00}$	$32_{0.2}$	$.11_{.00}$	$29_{0.1}$
2×2 LS	MTP+2048	$.15_{.01}$	$33_{0.1}$	$.11_{.00}$	$23_{0.2}$	$.23_{.00}$	$23_{0.2}$	$.12_{.00}$	$36_{0.0}$	$.12_{.00}$	$32_{0.1}$	$.12_{.00}$	$28_{0.2}$
2×2 LS	CLM+512	$.11_{.00}$	$33_{0.1}$	$.15_{.00}$	$26_{0.2}$	$.25_{.00}$	$23_{0.3}$	$.19_{.00}$	$36_{0.1}$	$.11_{.00}$	$33_{0.2}$	$.13_{.00}$	$30_{0.1}$
2×2 LS	MTP+512	$.17_{.00}$	$33_{0.2}$	$.16_{.00}$	$26_{0.2}$	$.24_{.00}$	$23_{0.2}$	$.15_{.00}$	$36_{0.1}$	$.13_{.00}$	$33_{0.3}$	$.12_{.00}$	$29_{0.1}$

well to inference speedups when a target language is not included in pre-training data and is written in non-Latin scripts.[13]

7.2 Training Strategy

We analyze the effectiveness of each training strategy introduced in §5. Given that LAPT does not typically improve classification performance in low-resource settings, we focus on generation tasks for brevity.[14] Due to limited computational resources, we only use the best-performing Align as the target parameter initialization method and experiment with six languages with the largest speedups (i.e., Arabic, Burmese, Greek, Hindi, Sinhala, and Telugu) in SUM. Table 5 lists the performance of the adapted models.

Looking at the results by training procedure, we observe that 2×2 LS improves performance across tasks and languages. Gains range from 1 point (Hindi SUM) to 10 points (Burmese MT) compared with LoRA with CLM+2048 (our default approach in §7.1). The sole exception is Arabic SUM, where performance remains the same as the baseline. Notably, 2×2 LS also helps to partially mitigate performance drops observed in SUM for languages like Burmese and Greek. We hypothesize that 2×2 LS can reduce the risk of underfitting by focusing on calibrating only the parts closely related to the encoding and decoding of the target language. This suggests that the frequently used full LoRA approach in high-resource settings is not the best training approach in

13 If inference speedups do not matter, we recommend using CPT-only in general. Since it does not involve parameter updates from scratch, it can lead to more stable task performance in low-resource settings.

14 The corresponding classification results are presented in Table B.1 in the Appendix. While some fluctuations are observed, we confirm that no training strategy offers consistent improvements across tasks and languages.

low-resource settings. Although 2-stage does not consistently provide gains over LoRA, especially for MT, it does not lead to substantial performance degradation either, with a maximum drop of 3 points for Greek MT, compared with the default LoRA with CLM+2048.

Next, we examine the effectiveness of MTP. Although it does not consistently improve performance over CLM, it notably boosts Arabic MT performance across models, particularly when used with 2×2 LS. Similar to the 2-stage approach, any performance degradation observed with MTP is minor, staying within 2 points. The only exceptions across all cases are Burmese and Hindi MT with 2×2 LS, and Telugu SUM with 2-stage, where we observe moderate drops of 3 to 4 points. Based on these results, while MTP does not offer universal gains, its consistent benefit for a specific language like Arabic MT and its generally stable performance (i.e., avoiding substantial drops) across other tasks make it a valuable strategy, particularly when used with 2×2 LS.[15]

Finally, using a short sequence length (512) works well across models and languages, showing improvements of up to 9 points (LoRA+MTP in Burmese and 2-stage+CLM in Greek MT), confirming our hypothesis (§5) that increasing the number of model updates can help avoid underfitting.

Recommendation. 2×2 LS generally leads to performance improvements. The short training sequence length of 512 can also aid low-resource settings. While MTP does not always offer improvements, its notable boost for a certain language and its overall stable performance, with no substantial degradation elsewhere, make it a viable and beneficial strategy for adaptation in low-resource environments.

7.3 Experiments with Other Source LLMs

We investigate whether models other than Llama2 adapted based on our recommendations in §7.1 and §7.2 benefit from inference speedups while maintaining competitive performance to their base models. We use Llama3 and Gemma2 as source. We apply the 2×2 LS+MTP+512 training strategy and initialize models with either Mean or Align along with Random as a baseline. Due to resource constraints, we only experiment with Burmese, Sinhala, and Telugu in the remainder of the article, as they are the worst fragmented languages of our target languages (Figure 2). Table 6 shows the performance of adapted models using Llama3 and Gemma2 as source.

Overall, models adapted with Align generally either perform competitively or outperform their respective Llama3 and Gemma2 base models (Source) on both generation tasks. The only exception is Burmese SUM with Gemma2, which shows a substantial 9-point drop. In contrast, models adapted with Mean, along with Random, often underperform Align when using Llama3 as the base model. Specifically, they underperform Align in Burmese SUM, Sinhala, and Telugu MT tasks, and across almost all classification tasks. This underperformance is attributed to underfitting, as evidenced by their higher perplexities compared to Align (see Table 7).

A consistent challenge across vocabulary expansion approaches, including Align, is substantial performance degradation on classification tasks. This degradation ranges from an 8-point drop (Sinhala MC and GMMLU with Gemma2) to a significant 35-point

15 It is worth noting that MTP typically offers an additional benefit of improving inference efficiency through self-speculative decoding (Gloeckle et al. 2024), which can be a huge advantage in practical deployments, though this aspect is not the primary focus of our analysis and beyond the scope of this article.

Table 6
Mean performance and inference speedup with Llama3 and Gemma2 as source. Green indicates positive performance change over Source. The speedup ratio corresponds to Align.

	Llama3	Burmese MT	Burmese SUM	Burmese MC	Burmese GMMLU	Sinhala MT	Sinhala SUM	Sinhala MC	Sinhala GMMLU	Telugu MT	Telugu SUM	Telugu MC	Telugu GMMLU
	Source	$.09_{.00}$	$28_{0.0}$	.41	–	$.19_{.00}$	$30_{0.0}$	.50	.37	$.24_{.00}$	$29_{0.0}$	.51	.40
	CPT-only	$.17_{.00}$	$28_{0.1}$	.31	–	$.21_{.00}$	$31_{0.1}$	.47	.36	$.30_{.00}$	$29_{0.1}$	.42	.38
+Speedup	Random	$.19_{.00}$	$23_{0.2}$	.28	–	$.14_{.00}$	$31_{0.2}$	.23	.25	$.13_{.00}$	$29_{0.1}$	.27	.26
+Speedup	Mean	$.19_{.00}$	$23_{0.2}$	.25	–	$.12_{.00}$	$32_{0.1}$	.23	.23	$.12_{.00}$	$29_{0.2}$	.26	.25
+Speedup	Align	$.21_{.00}$	$28_{0.3}$	.28	–	$.22_{.00}$	$33_{0.1}$	.30	.26	$.31_{.00}$	$31_{0.1}$	.38	.31
Speedup		2.60x	3.52x	2.30x	–	2.36x	3.19x	1.98x	1.87x	2.15x	3.55x	1.96x	1.85x

	Gemma2	Burmese MT	Burmese SUM	Burmese MC	Burmese GMMLU	Sinhala MT	Sinhala SUM	Sinhala MC	Sinhala GMMLU	Telugu MT	Telugu SUM	Telugu MC	Telugu GMMLU
	Source	$.21_{.01}$	$34_{0.1}$	.67	–	$.19_{.00}$	$34_{0.2}$	.71	.50	$.31_{.01}$	$30_{0.2}$	.74	.57
	CPT-only	$.28_{.00}$	$28_{0.3}$	.61	–	$.29_{.00}$	$34_{0.1}$	.69	.48	$.43_{.00}$	$29_{0.1}$	.71	.56
+Speedup	Random	$.24_{.00}$	$26_{0.3}$	.46	–	$.26_{.00}$	$33_{0.1}$	.63	.43	$.35_{.00}$	$28_{0.2}$	.65	.47
+Speedup	Mean	$.25_{.00}$	$25_{0.3}$	.49	–	$.27_{.00}$	$32_{0.2}$	.63	.42	$.33_{.00}$	$28_{0.1}$	.60	.43
+Speedup	Align	$.25_{.00}$	$25_{0.2}$	.32	–	$.28_{.00}$	$32_{0.1}$	.63	.42	$.32_{.00}$	$29_{0.1}$	.59	.43
Speedup		1.52x	1.57x	1.51x	–	1.26x	1.38x	1.31x	1.20x	1.07x	1.10x	1.06x	1.13x

Table 7
Perplexity on language-specific held-out dataset using Llama3 and Gemma2 as source. Results are not comparable between models with gray and others due to their difference in vocabulary.

(a) Llama3

Model	my	si	te
Source	3.7	3.5	3.0
CPT-only	1.9	2.0	1.8
Random	6.9	7.1	6.1
Mean	6.7	6.8	5.8
Align	6.0	5.5	4.6

(b) Gemma2

Model	my	si	te
Source	48.0	59.9	51.9
CPT-only	4.6	4.3	5.2
Random	9.0	6.9	6.7
Mean	9.0	6.9	6.6
Align	9.0	6.9	6.6

drop (Burmese MC with Gemma2) for the best-performing Align. This issue is not unique to these approaches, as CPT-only also frequently exhibits moderate to substantial degradation (3 to 10 points) on classification tasks across both models. This further highlights the inherent challenges of LAPT for classification tasks in low-resource settings, as discussed in §7.1.1. We later showcase in §8.2 that these substantial drops can be largely mitigated using a post-hoc, training-free method, achieving performance within 3 and 5 points of the Gemma2 base model for MC and GMMLU, respectively.

Beyond downstream performance, a core advantage of vocabulary expansion lies in inference efficiency. Align models, for instance, consistently exhibit inference speedups, reaching up to 3.52x for Llama3 and 1.57x for Gemma2.[16] These results confirm the

16 For context, speedups in the range of 1.3x to 3x are commonly reported for speculative decoding techniques (Agrawal, Jeon, and Lee 2024; Timor et al. 2025; Huang, Guo, and Wang 2025). Thus, the observed speedups generally represent a meaningful gain in inference efficiency for LLMs.

Table 8
Mean SUM performance in ROUGE-L and BLEURT over five runs. Green indicates positive performance change over Source.

		Burmese		Sinhala		Telugu	
Llama2		ROUGE-L	BLEURT	ROUGE-L	BLEURT	ROUGE-L	BLEURT
	Source	$24_{0.1}$	$.03_{.00}$	$26_{0.1}$	$.19_{.00}$	$21_{0.2}$	$.13_{.00}$
	CPT-only	$24_{0.3}$	$.04_{.00}$	$23_{0.1}$	$.25_{.00}$	$29_{0.1}$	$.22_{.00}$
+Speedup	Random	$24_{0.1}$	$.07_{.00}$	$32_{0.2}$	$.26_{.00}$	$29_{0.1}$	$.23_{.00}$
+Speedup	Mean	$25_{0.1}$	$.06_{.00}$	$32_{0.1}$	$.25_{.00}$	$29_{0.1}$	$.23_{.01}$
+Speedup	Align	$26_{0.2}$	$.08_{.01}$	$33_{0.3}$	$.26_{.01}$	$29_{0.1}$	$.24_{.00}$
Llama3							
	Source	$28_{.00}$	$.06_{.00}$	$30_{.00}$	$.24_{.00}$	$29_{.00}$	$.22_{.00}$
	CPT-only	$28_{0.1}$	$.06_{.00}$	$31_{0.1}$	$.25_{.00}$	$30_{0.1}$	$.29_{.00}$
+Speedup	Random	$23_{0.2}$	$.05_{.00}$	$31_{0.2}$	$.27_{.00}$	$29_{0.1}$	$.24_{.01}$
+Speedup	Mean	$23_{0.2}$	$.05_{.00}$	$32_{0.1}$	$.29_{.01}$	$29_{0.2}$	$.26_{.00}$
+Speedup	Align	$28_{0.3}$	$.07_{.00}$	$33_{0.1}$	$.31_{.00}$	$31_{0.1}$	$.28_{.00}$
Gemma2							
	Source	$34_{0.1}$	$.10_{.00}$	$34_{0.2}$	$.32_{.00}$	$30_{0.2}$	$.30_{.00}$
	CPT-only	$28_{0.3}$	$.10_{.00}$	$34_{0.1}$	$.32_{.01}$	$29_{0.1}$	$.26_{.00}$
+Speedup	Random	$26_{0.3}$	$.10_{.00}$	$33_{0.1}$	$.34_{.01}$	$28_{0.2}$	$.26_{.00}$
+Speedup	Mean	$25_{0.3}$	$.09_{.00}$	$32_{0.2}$	$.33_{.00}$	$28_{0.1}$	$.27_{.00}$
+Speedup	Align	$25_{0.2}$	$.10_{.00}$	$32_{0.1}$	$.33_{.01}$	$29_{0.1}$	$.28_{.00}$

versatility of our approaches in optimizing inference speed across different base model architectures. However, we note a relatively moderate inference speedup of up to 1.13x for Telugu with Gemma2, despite its higher text fragmentation rate than Sinhala (Figure 2). This can be attributed to lower target token ratios across tasks for both input and output contexts (Figure 4), suggesting an underutilization of newly added tokens and highlighting potential avenues for improvement in the selection of new tokens for specific languages.

Finally, examining the results by source model and generation task, we observe that adapted Llama3 and Gemma2 models show substantially better MT performance than those with Llama2 despite their similar model size. In particular, the Telugu-adapted models initialized with Align with Llama3 and Gemma2 as source obtain 19 and 20 points better performance than those Llama2 counterparts. This trend suggests a successful cross-lingual transfer of the generative capabilities of the base models. However, SUM does not seem to follow the trend, i.e., the adapted models with Llama3 and Gemma2 do not outperform those with Llama2 when evaluated with ROUGE-L. In fact, their semantic-level performance measured by BLEURT (Sellam, Das, and Parikh 2020) improves up to 7 points from those with Llama2 (see Table 8). Thus, adapted models actually enjoy their base model capabilities to improve performance at the semantic level rather than the surface level measured by ROUGE-L.

7.4 Target Vocabulary Size

The extent to which a target language benefits from inference speedups varies across languages, models, and tasks, as observed from Tables 4 and 6. Although larger $|\mathcal{V}_{\text{new}}|$

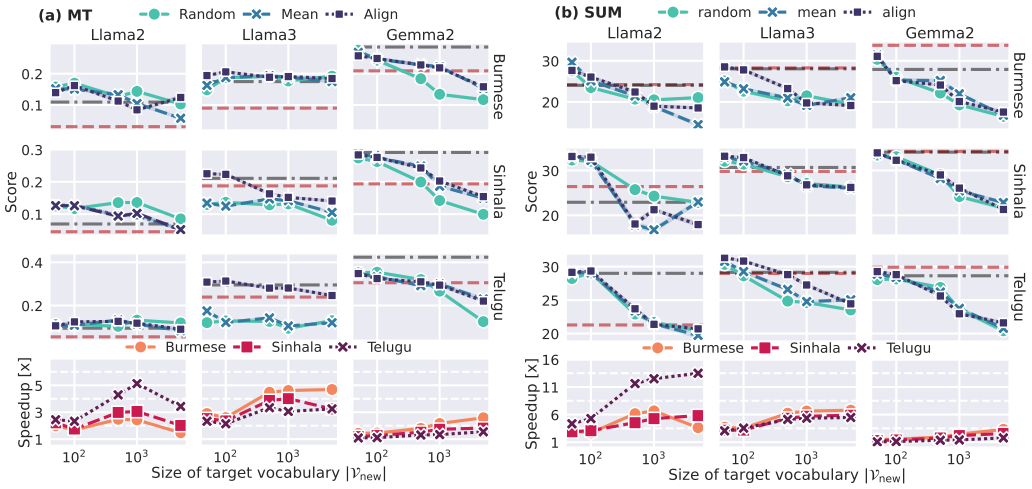

Figure 3
Downstream performance and inference speedup in (a) MT and (b) SUM across different $|\mathcal{V}_{\text{new}}|$. Red and gray dotted lines denote Source and CPT-only.

can lead to faster inference, it comes with the risk of underfitting in low-resource settings.[17] To help select an optimal $|\mathcal{V}_{\text{new}}|$ for obtaining substantial speedups while retaining competitive performance to Source, we conduct a cost-benefit analysis with respect to $|\mathcal{V}_{\text{new}}|$. We experiment with $|\mathcal{V}_{\text{new}}| = \{50, 100, 500, 1\text{K}, 5\text{K}\}$ while previous studies set $|\mathcal{V}_{\text{new}}|$ to over 10K, e.g., 17,953 for Chinese Llama. For comprehensiveness, we use all three source models and follow the same setup as in §7.3.[18]

7.4.1 Generation Tasks. Figure 3 shows the performance changes and corresponding inference speedups for different $|\mathcal{V}_{\text{new}}|$ on generation tasks. We first observe that the larger $|\mathcal{V}_{\text{new}}|$, the worse the task performance across different source models, languages, and tasks in general. Notably, SUM appears to be less robust to the changes in $|\mathcal{V}_{\text{new}}|$ than MT across different models and languages. Adapted models underperform Source even at $|\mathcal{V}_{\text{new}}| = 500$ in the majority of the cases, while they, especially when initialized with Align, still outperform or rival Source in MT in almost all the cases. This difference can be due to the difficulty of each task as discussed in §7.1, i.e., SUM requires more $\mathcal{D}$ than MT to perform well. These suggest that $|\mathcal{V}_{\text{new}}|$ should be set smaller (e.g., $|\mathcal{V}_{\text{new}}| = 100$ in our setup) for tasks like SUM to be competitive with Source in task performance.

Next, we observe that the larger $|\mathcal{V}_{\text{new}}|$, the faster the inference up to around $|\mathcal{V}_{\text{new}}| = 1\text{K}$ in all the cases. After that point, the inference speedups tend to plateau or sometimes deteriorate. This can be partially due to underfitting, which hinders a model from effectively using $\mathcal{V}_{\text{new}}$. Indeed, the target token ratio in output at $|\mathcal{V}_{\text{new}}| = 5\text{K}$ drops by 59.7% (Burmese), 9.5% (Sinhala), and 30.0% (Telugu) from its peak at $|\mathcal{V}_{\text{new}}| = 500$ or 1K when using Llama2 on SUM (Figure 4). Therefore, although larger $|\mathcal{V}_{\text{new}}|$ can shorten

17 Given a small and limited $\mathcal{D}$, the larger $|\mathcal{V}_{\text{new}}|$, the less fragmentation with respect to $\mathcal{D}$, the fewer the number of training tokens, and therefore the fewer model updates.
18 Table B.2 in the Appendix provides a qualitative example of how tokenization in Burmese changes with respect to $|\mathcal{V}_{\text{new}}|$ using Llama2 as source.

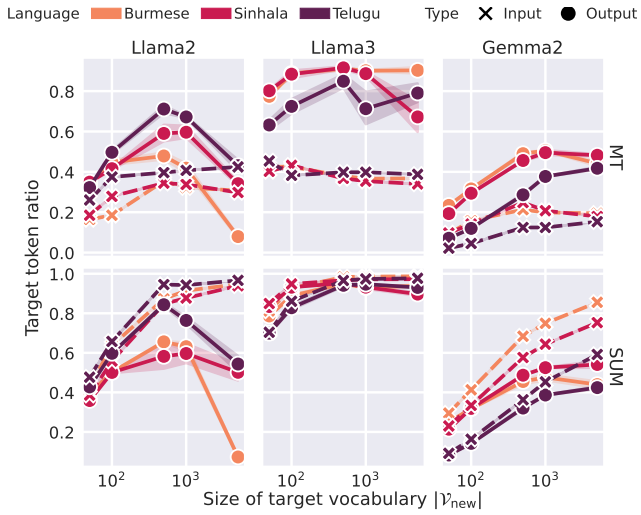

Figure 4
Average target token ratio in input (×) and output (•) with respect to $|\mathcal{V}_{\text{new}}|$ across models, languages, and tasks.

prompt length in a target language, it does not always guarantee inference speedups unless a model is well-trained to handle $\mathcal{V}_{\text{new}}$.

7.4.2 Classification Tasks. Figure 5 shows the performance changes and corresponding inference speedups with respect to different $|\mathcal{V}_{\text{new}}|$ on classification tasks. Unlike generation tasks, larger $|\mathcal{V}_{\text{new}}|$ does not always result in worse task performance. This is especially evident in Llama2 and Llama3 across tasks. Nonetheless, Gemma2 adapted

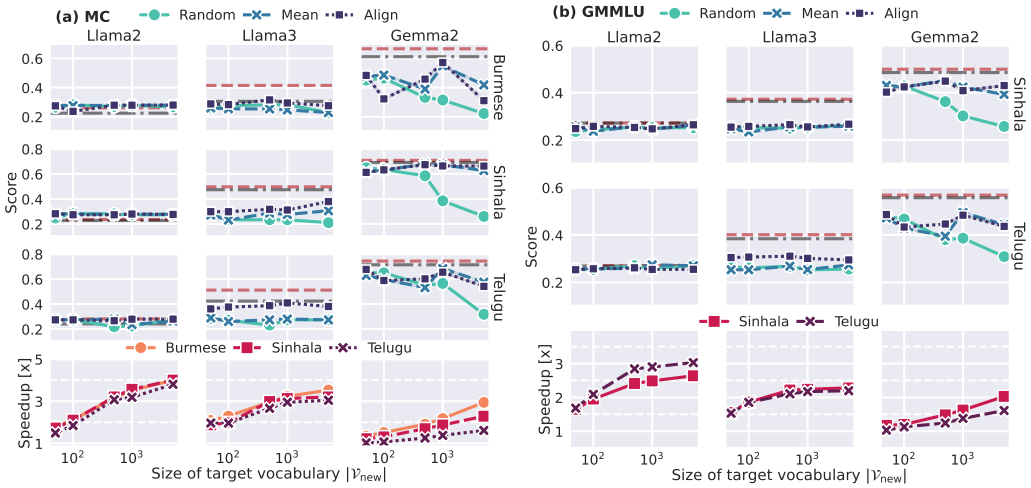

Figure 5
Downstream performance and inference speedup in (a) MC and (b) GMMLU across different $|\mathcal{V}_{\text{new}}|$. Red and gray dotted lines denote Source and CPT-only.

models with Random often exhibit performance degradation as $|\mathcal{V}_{\text{new}}|$ becomes larger across tasks and languages, suggesting the importance of target parameter initialization.

We hypothesize that the observed difference between classification and generation tasks stems from the fundamental nature of the task, as discussed in §7.1.1 and §7.3. Classification often involves predicting only a single token, which requires less extensive and nuanced generation capabilities compared to generation tasks, where models must produce multi-token sequences. Therefore, a model might not need to be as well-trained to effectively utilize $\mathcal{V}_{\text{new}}$ for inference in classification, as it does not generate sequences incorporating those new tokens. While performance degradation is substantial in Llama3 and Gemma2 (as observed in §7.3), classification tasks generally appear more robust to changes in $|\mathcal{V}_{\text{new}}|$. This also explains why inference speedups tend not to plateau in most classification scenarios, as larger $|\mathcal{V}_{\text{new}}|$ can effectively shorten prompt length in a target language (Table B.2 for an qualitative example), leading to continued speedup gains without the performance trade-offs seen on generative tasks.

Recommendation. Setting $|\mathcal{V}_{\text{new}}|$ between 500 and 1K for MT and 100 for SUM can be a suitable starting point to maintain competitive performance to Source while benefiting greatly from inference speedups under low-resource settings. While classification tasks are more robust to changes in $|\mathcal{V}_{\text{new}}|$, it is often preferable to utilize either CPT-only or Source for better performance if: (i) inference speedups are not the primary priority, and (ii) a post-hoc, training-free performance degradation mitigation technique, as discussed later in §8.2, is not applied.

8. Discussion

8.1 Source Knowledge Retention

Previous work (Tejaswi, Gupta, and Choi 2024; Mundra et al. 2024) has reported that vocabulary expansion followed by LAPT can lead to catastrophic forgetting of the original capabilities of a source model. We measure the extent to which adapted models in low-resource settings (30K sentences) suffer from this phenomenon by evaluating them on English-centric reading comprehension and general knowledge and reasoning benchmarks. To this end, we employ the English subset of Belebele (MC) and MMLU (Hendrycks et al. 2021). We use all three source models and follow the same setup as in §7.3 and §7.4. However, for brevity, we evaluate only the best-performing Align initialization method. Table 9 presents the corresponding results on MC and MMLU.

Overall, we observe varying degrees of source knowledge retention depending on the adaptation method. CPT-only (i.e., continual per-training without vocabulary expansion) generally preserves original source model capabilities, showing negligible performance drops of up to 2 points across different base models. In contrast, adapted models initialized with Align exhibit moderate to substantial performance degradation on these English tasks, with average drops ranging from 5.3 points for Gemma2, to 10 points for Llama2, and a substantial 18.7 points for Llama3. These results corroborate the findings in Mundra et al. (2024), which noted that performance degradation in source language tasks tends to be more severe during early LAPT stages with vocabulary expansion. This suggests that while LAPT with Align is effective for target language adaptation, especially on generation tasks, it comes with a considerable trade-off in the retention of the original source capabilities.

Table 9
Source knowledge retention evaluation on English benchmarks. Each language name indicates that a model has undergone LAPT on its corresponding target language data (30K sentences).

	Burmese		Sinhala		Telugu	
Llama2	MC	MMLU	MC	MMLU	MC	MMLU
Source	.52	.46	.52	.46	.52	.46
CPT-only	.53	.47	.52	.47	.52	.46
Align	.38	.37	.42	.39	.41	.37

	Burmese		Sinhala		Telugu	
Llama3	MC	MMLU	MC	MMLU	MC	MMLU
Source	.88	.67	.88	.67	.88	.67
CPT-only	.86	.66	.86	.66	.87	.66
Align	.75	.46	.65	.39	.78	.50

	Burmese		Sinhala		Telugu	
Gemma2	MC	MMLU	MC	MMLU	MC	MMLU
Source	.91	.72	.91	.72	.91	.72
CPT-only	.90	.72	.91	.72	.90	.72
Align	.88	.67	.89	.66	.84	.63

8.2 Can We Recover Performance Degradation from Vocabulary Expansion in Low-Resource Settings Without Training?

As discussed in §7, vocabulary expansion in low-resource settings offers inference speedups and largely retains or even improves performance on generation tasks. However, it leads to performance degradation in (i) target language classification tasks (§7.3 and §7.4) and (ii) source language capabilities (§8.1). This section investigates whether such performance degradation can be recovered without additional training. Indeed, our work, ElChat (Yamaguchi et al. 2025), provides a post-hoc, training-free method to recover the original capabilities of adapted models.

ElChat: A Post-hoc Method to Mitigate Catastrophic Forgetting. ElChat (Yamaguchi et al. 2025) is a post-hoc, training-free method designed to restore the original capabilities of an LLM after it has undergone vocabulary expansion and continual pre-training on target language data. The method requires access to an instruction-tuned model that has been further supervised fine-tuned on labeled conversational data, enabling the source base model (i.e., $\mathcal{M}_s$) to follow instructions. This prerequisite is readily met in practice, as frontier models like Llama3 and Gemma 2 commonly offer both base and instruction-tuned variants. ElChat addresses catastrophic forgetting of original capabilities by utilizing a strategic combination of two core steps: model merging and copying special token weights. This allows for robust recovery of degraded performance without incurring further training costs.[19]

Results. Constrained by resources, we apply ElChat to adapted Gemma2 models initialized with Align to evaluate its impact on both target language and English tasks.[20] We observe from Table 10(a) that ElChat consistently improves performance across target language tasks. Specifically, for classification tasks, the performance drop relative to Source is substantially reduced within 5 points (Telugu GMMLU), a notable

19 For more technical details, please refer to Yamaguchi et al. (2025).
20 Gemma2 is chosen for its notable performance degradation in target language tasks (not only in classification tasks but also in Burmese SUM) and as a representative of more recent models than Llama2 and Llama3, enabling a practical and meaningful comparison within our budget.

Table 10
Gemma2 performance and inference speedup using ElChat, a post-hoc training-free method to mitigate catastrophic forgetting. Green indicates positive performance change over Source.

(a) Target language tasks

		Burmese				Sinhala				Telugu			
		MT	SUM	MC	GMMLU	MT	SUM	MC	GMMLU	MT	SUM	MC	GMMLU
	Source	$.21_{.01}$	$34_{0.1}$	.67	–	$.19_{.00}$	$34_{0.2}$	.71	.50	$.31_{.01}$	$30_{0.2}$	.74	.57
	CPT-only	$.28_{.00}$	$28_{0.3}$	.61	–	$.29_{.00}$	$34_{0.1}$	.69	.48	$.43_{.00}$	$29_{0.1}$	.71	.56
+Speedup	Align	$.25_{.00}$	$25_{0.2}$	.32	–	$.28_{.00}$	$32_{0.1}$	.63	.42	$.32_{.00}$	$29_{0.1}$	.59	.43
+Speedup	+ ElChat	$.30_{.00}$	$33_{0.1}$	.64	–	$.31_{.00}$	$35_{0.1}$	.73	.46	$.41_{.00}$	$31_{0.1}$	.74	.52
Speedup		1.52x	1.57x	1.51x	–	1.26x	1.38x	1.31x	1.20x	1.07x	1.10x	1.06x	1.13x
	(ElChat)	1.53x	1.73x	1.48x	–	1.20x	1.39x	1.30x	1.18x	1.09x	1.14x	1.06x	1.02x

(b) English tasks

	Burmese		Sinhala		Telugu	
Gemma2	MC	MMLU	MC	MMLU	MC	MMLU
Source	.91	.72	.91	.72	.91	.72
CPT-only	.90	.72	.91	.72	.90	.72
Align	.88	.67	.89	.66	.84	.63
+ ElChat	.93	.72	.93	.71	.93	.71

improvement from up to 35 points (Burmese MC) without ElChat. Furthermore, ElChat not only recovers degraded performance but also maintains the superior generative capabilities of the adapted models, even showing improvements in some cases (e.g., Sinhala SUM). Crucially, these performance recoveries are achieved while largely retaining the inference speedups gained from vocabulary expansion. The effect of ElChat is even more pronounced in English tasks (Table 10(b)). The adapted models almost fully recover their original source capabilities, matching Source performance with a negligible drop of at most 1 point. This highlights the effectiveness of using a post-hoc, training-free method to mitigate performance degradation from vocabulary expansion without requiring any additional training.

Recommendation. Using a post-hoc, training-free method like ElChat enables an adapted model to restore the original capabilities of the corresponding source model without additional training. For target language tasks, this can lead to enhanced performance while preserving the inference speedups gained from vocabulary expansion.

8.3 Comparison with Vocabulary Replacement

Intuitively, vocabulary replacement, which typically replaces the entire source vocabulary with a new one from a target language, necessitates a greater number of training tokens than vocabulary expansion. This is attributed to its substantially larger number of new parameters requiring alignment. For instance, Dagan, Synnaeve, and Roziere (2024) found that over 50 billion tokens were necessary to swap a tokenizer at no performance cost in their domain adaptation experiments. To illustrate the challenges

Table 11
Vocabulary replacement performance and inference speedup on target language tasks using Gemma2 as source. Green indicates positive performance change over Source. The speedup ratio corresponds to Mean.

		Burmese				Sinhala				Telugu			
		MT	SUM	MC	GMMLU	MT	SUM	MC	GMMLU	MT	SUM	MC	GMMLU
	Source	$.21_{.01}$	$34_{0.1}$	.67	–	$.19_{.00}$	$34_{0.2}$	.71	.50	$.31_{.01}$	$30_{0.2}$	.74	.57
	CPT-only	$.28_{.00}$	$28_{0.3}$	.61	–	$.29_{.00}$	$34_{0.1}$	.69	.48	$.43_{.00}$	$29_{0.1}$	.71	.56
+Speedup	Random	$.00_{.00}$	$0_{0.0}$	.22	–	$.00_{.00}$	$0_{0.0}$	.27	.25	$.00_{.00}$	$1_{0.1}$	.24	.23
+Speedup	FOCUS	$.00_{.00}$	$4_{0.1}$	.22	–	$.02_{.00}$	$12_{0.3}$	.27	.27	$.00_{.00}$	$3_{0.0}$	.23	.27
+Speedup	Mean	$.06_{.00}$	$16_{0.1}$	.26	–	$.01_{.00}$	$23_{0.3}$	.27	.26	$.04_{.00}$	$12_{0.3}$	.34	.26
Speedup		1.14x	2.27x	2.95x	–	1.01x	1.36x	2.70x	2.44x	0.89x	0.57x	1.77x	1.91x

of vocabulary replacement for target language adaptation in low-resource settings compared to vocabulary expansion, we conduct a brief comparative study using Gemma2 as the source model.

Experimental Setup. We first train a target tokenizer with a vocabulary size of 32K using target language data $\mathcal{D}$ and the same training configurations as the base Gemma2 tokenizer. For target parameter initialization, we consider Random, FOCUS, and Mean. We then continually pre-train each initialized model using the same approach as in Sections 7.3 and 7.4 (i.e., 2×2 LS+MTP+512).

Results. Table 11 presents the performance and inference speedups of models adapted using vocabulary replacement. Overall, these models exhibit notably poor performance across all initialization approaches and tasks in low-resource settings. Specifically, Random and FOCUS largely fail to perform well on generation tasks, yielding near-zero scores in MT and low scores of up to 12 points in SUM. While Mean shows slight improvement over Random and FOCUS, its performance remains substantially lower than that of Source, CPT-only, and crucially, the vocabulary expansion counterpart (Table 6). For instance, in Telugu MC, the best vocabulary replacement performance (Mean at 34 points) is still far below Source (74) and vocabulary expansion with Mean (60). The only advantage of vocabulary replacement lies in its superior inference speedups in some tasks (e.g., Burmese MC at 2.95x, Sinhala MC at 2.70x, and Telugu GMMLU at 1.91x), which stems from its entirely optimized vocabulary for the target language. These results clearly demonstrate the significant challenges of effectively applying vocabulary replacement for target language adaptation under low-resource settings, when compared to vocabulary expansion.

8.4 Limitations

Target Language Classification Performance. While we demonstrate that ElChat, a post-hoc and training-free method, helps adapted models recover degraded performance in target language classification tasks, a moderate performance drop can still persist, as observed in Telugu GMMLU (Table 10). Furthermore, ElChat is not applicable in scenarios where only base models are available as source for adaptation (i.e., lacking an instruction-tuned variant). These limitations represent ongoing challenges that extend beyond the immediate scope of this article.

Comparison with Fully Multilingual LLMs. Due to resource constraints, this article does not include a direct comparison with fully multilingual models like MaLA-500 (Lin et al. 2024) and EMMA-500 (Ji et al. 2024, 2025). Such a comparison would help contextualize our results.

Model Size. Our experiments use models of up to 9B parameters due to resource constraints. Scaling these experiments to larger LLMs is a valuable direction for future work to confirm the generalizability of our findings.

Tokenizer. Heuristic-based target parameter initialization with Merge assumes the use of a BPE-based tokenizer which is a common choice in recent LLMs, e.g., Gemma2, Llama3, Llama2, Mistral (Jiang et al. 2023), inter alia. Experimenting with other tokenizers, such as Unigram (Kudo 2018), falls outside the scope of this article.

9. Conclusion

We investigated cross-lingual vocabulary expansion in low-resource settings across target parameter initialization approaches and training strategies. Our extensive experiments reveal that a widely used approach in high-resource settings is not always optimal in low-resource settings. In contrast, models adapted by our alternative strategies achieve faster inference while rivaling their base models in task performance, especially on generation tasks. We supplement our analysis with specific recommendations for effective vocabulary expansion.

Appendix A. Details on Experimental Setup

Table A.1 lists prompt templates for all tasks, while Table A.2 lists the hyperparameters used for both (a) LAPT and (b) inference.

Appendix B. Supplementary Results

Table B.1 lists the performance of the adapted models with different training strategies on classification tasks.

Table B.2 visualizes how tokenization changes with vocabulary expansion in Burmese using Llama2 as source.

Table A.1
Prompt template for each task and language. For MC and GMMLU, we omit text breaks (\n) after context, question, and each option for readability.

Task	Language	Template
MT	English	Translate English to {X: a target language}: {sentence} =
	Arabic	ترجم العربية إلى العربية: {sentence} =
	Burmese	အင်္ဂလိပ်မှ မြန်မာသို့ ဘာသာပြန်ပါ။: {sentence} =
	German	Übersetzen Sie Englisch ins Deutsche: {sentence} =
	Greek	Μεταφράστε τα αγγλικά στα ελληνικά: {sentence} =
	Hindi	अंग्रेजी से हिंदी में अनुवाद करें: {sentence} =
	Japanese	英語から日本語へ翻訳しなさい: {sentence} =
	Sinhala	ඉංග්‍රීසි සිංහලයට පරිවර්තනය කරන්න: {sentence} =
	Swahili	Tafsiri Kiingereza hadi Kiswahili: {sentence} =
	Telugu	ఆంగ్లం నుండి తెలుగుకు అనువదించండి: {sentence} =
	Thai	แปลภาษาอังกฤษเป็นไทย: {sentence} =
SUM	English	Write a short summary of the following text in {language}. Article: {text} Summary:
	Arabic	اكتب ملخصًا قصيرًا للنص التالي باللغة العربية. المقالة: {text} الملخص:
	Burmese	အောက်ပါစာသားကို မြန်မာဘာသာဖြင့် အကျဉ်းချုပ်ရေးပါ။ ဆောင်းပါး: {text} အကျဉ်းချုပ်:
	German	Schreiben Sie eine kurze Zusammenfassung des folgenden Textes auf Deutsch. Artikel: {text} Zusammenfassung:
	Greek	Γράψε μια σύντομη περίληψη του παρακάτω κειμένου στα ελληνικά. Άρθρο: {text} Περίληψη:
	Hindi	निम्नलिखित का संक्षेप हिंदी में लिखे। लेख: {text} संक्षेप:
	Japanese	次の文章の要約を日本語で書きなさい。記事: {text} 要約:
	Sinhala	පහත පාඨයේ සාරාංශය සිංහලෙන් ලියන්න. ලිපිය: {text} සාරාංශය:
	Swahili	Andika muhtasari mfupi wa maandishi yafuatayo kwa Kiswahili. Makala: {text} Muhtasari:
	Telugu	క్రింది వచనం యొక్క సారాంశం తెలుగులో రాయండి. వ్యాసం: {text} సారాంశం:
	Thai	เขียนสรุปสั้น ๆ ของข้อความต่อไปนี้เป็นภาษาไทย บทความ: {text} สรุป:
MC GMMLU	English	{context} Question: {question} A. {option A} B. {option B} C. {option C} D. {option D} Answer:
	Arabic	{context} سؤال: {question} A. {option A} B. {option B} C. {option C} D. {option D} إجابة:
	Burmese	{context} မေးခွန်း: {question} A. {option A} B. {option B} C. {option C} D. {option D} အဖြေ:
	German	{context} Frage: {question} A. {option A} B. {option B} C. {option C} D. {option D} Antwort:
	Greek	{context} Ερώτηση: {question} A. {option A} B. {option B} C. {option C} D. {option D} Απάντηση:
	Hindi	{context} सवाल: {question} A. {option A} B. {option B} C. {option C} D. {option D} उत्तर:
	Japanese	{context} 質問: {question} A. {option A} B. {option B} C. {option C} D. {option D} 回答:
	Sinhala	{context} ප්‍රශ්නය: {question} A. {option A} B. {option B} C. {option C} D. {option D} පිළිතුර:
	Swahili	{context} Swali: {question} A. {option A} B. {option B} C. {option C} D. {option D} Jibu:
	Telugu	{context} ప్రశ్న: {question} A. {option A} B. {option B} C. {option C} D. {option D} జవాబు:
	Thai	{context} คำถาม: {question} A. {option A} B. {option B} C. {option C} D. {option D} คำตอบ:

Table A.2
Hyperparameter configurations for LAPT and inference.

(a) LAPT

Hyperparameters	Values
Batch size	8
Maximum number of training epochs	2
Adam ϵ	1e-8
Adam β_1	0.9
Adam β_2	0.999
Sequence length	2,048
Learning rate	1e-4
Learning rate scheduler	cosine
Warmup steps	100
Weight decay	0.01
Attention dropout	0.0
Dropout	0.05
LoRA rank r	8
LoRA dropout	0.05
LoRA α	32

(b) Inference

Parameters	Values
Maximum prompt length	4,096
Temperature	0.8
Repetition penalty	1.1
Top k	40
Top p	0.9
Beam width	5
Sampling	True
Early stopping	True
Maximum number of generated tokens	128

Table B.1
Mean performance of Align models on classification tasks with Llama2 as source. Green indicates positive performance change over Source.

		Arabic		Burmese		Greek		Hindi		Sinhala		Telugu	
	Model	MC	GMMLU	MC	GMMLU	MC	GMMLU	MC	GMMLU	MC	GMMLU	MC	GMMLU
	Source	.29	.29	.26	–	.27	.28	.25	.28	.24	.27	.28	.27
	CPT-only	.30	.29	.22	–	.29	.29	.28	.28	.23	.27	.24	.27
LoRA	CLM+2048	.28	.25	.26	–	.31	.26	.27	.27	.27	.27	.22	.27
LoRA	MTP+2048	.29	.26	.28	–	.26	.28	.28	.25	.28	.26	.27	.27
LoRA	CLM+512	.28	.25	.24	–	.33	.26	.28	.27	.27	.26	.27	.26
LoRA	MTP+512	.29	.28	.24	–	.27	.27	.30	.26	.24	.25	.27	.27
2-stage	CLM+2048	.30	.27	.24	–	.26	.25	.27	.26	.27	.27	.27	.26
2-stage	MTP+2048	.29	.27	.22	–	.23	.24	.28	.26	.29	.26	.27	.26
2-stage	CLM+512	.25	.28	.22	–	.29	.27	.27	.27	.27	.25	.26	.27
2-stage	MTP+512	.23	.28	.22	–	.27	.28	.29	.26	.28	.25	.22	.27
2×2 LS	CLM+2048	.29	.27	.22	–	.29	.24	.28	.26	.27	.26	.28	.26
2×2 LS	MTP+2048	.30	.27	.26	–	.32	.25	.28	.27	.28	.24	.29	.27
2×2 LS	CLM+512	.31	.29	.22	–	.28	.24	.28	.28	.27	.26	.28	.26
2×2 LS	MTP+512	.28	.27	.24	–	.23	.23	.28	.26	.27	.26	.27	.26

Table B.2
Example of tokenization changes with vocabulary expansion in Burmese using Llama2 as source. '_' stands for a whitespace.

$\lvert\mathcal{V}_{\text{new}}\rvert$	# tokens	Tokenization
		English (Reference text)
Source	15	<s> _The _Amazon _River _is _the _second _longest _and _the _biggest _river _on _Earth .
		Burmese
Source	134	<s> _ <0xE1> <0x80> <0xA1> မ ေ <0xE1> <0x80> <0x87> ု နဲ ့ မ ြ <0xE1> <0x80> <0x85> ် သ <0xE1> <0x80> <0x8A> ် _ က မ <0xE1> <0x80> <0xB9> <0xE1> <0x80> <0x98> ာ ပ ေ <0xE1> <0x80> <0xAB> ် တ <0xE1> <0x80> <0xBD> င ် _ <0xE1> <0x80> <0x92> ု တ ိ <0xE1> <0x80> <0x9A> မ ြ ေ ာ က ် <0xE1> <0x80> <0xA1> ရ <0xE1> <0x80> <0xBE> <0xE1> <0x80> <0x8A> ် <0xE1> <0x80> <0x86> ု <0xE1> <0x80> <0xB6> း နဲ <0xE1> <0x80> <0xBE> င <0xE1> <0x80> <0xB7> ် _ <0xE1> <0x80> <0xA1> က ြ <0xE1> <0x80> <0xAE> း <0xE1> <0x80> <0x86> ု <0xE1> <0x80> <0xB6> း မ ြ <0xE1> <0x80> <0x85> ် <0xE1> <0x80> <0x96> ြ <0xE1> <0x80> <0x85> ် ပ <0xE1> <0x80> <0xAB> သ <0xE1> <0x80> <0x8A> ် <0xE1> <0x81> <0x8B>
50	72	<s> _အ မ ေ <0xE1> <0x80> <0x87> ု နဲ့ မ ြ စ် သ ည် _ က မ <0xE1> <0x80> <0xB9> <0xE1> <0x80> <0x98> ာ ပ ေ ါ် ် တ ွ င် _ <0xE1> <0x80> <0x92> ု တ ိ ယ မ ြ ော က် အ ရ ည် <0xE1> <0x80> <0x86> ုံ း နဲ့ င ့် _အ ကြ ီး <0xE1> <0x80> <0x86> ုံ း မ ြ စ် ဖြ စ် ပါ သ ည် <0xE1> <0x81> <0x8B>
100	51	<s> _အ မ ေ <0xE1> <0x80> <0x87> ု နဲ့ မြ စ် သည် _က မ <0xE1> <0x80> <0xB9> <0xE1> <0x80> <0x98> ာ ပ ေါ် ် တ ွ င် _ဒ ု တ ိ ယ မြ ော က် အ ရ ည် ဆ ုံး နဲ့ င ့် _အ ကြ ီး ဆ ုံး မြ စ် ဖြ စ် ပါ သည် ။
500	31	<s> _အ မေ ဇ ုန် မြ စ် သည် _က မ ဘာ ပေါ် တွင် _ဒ ု တိ ယ မြ ော က် အ ရ ည် ဆုံး နှင့် _အ ကြီး ဆုံး မြ စ် ဖြစ် ပါသည်။
1,000	25	<s> _အ မေ ဇ ုန် မြ စ် သည် _ကမ္ဘာ ပေါ် တွင် _ဒုတိယ မြောက် အ ရ ည် ဆုံး နှင့် _အ ကြီး ဆုံး မြ စ် ဖြစ် ပါသည်။
5,000	18	<s> _အ မေ ဇုန် မြစ် သည် _ကမ္ဘာ ပေါ်တွင် _ဒုတိယ မြောက် အ ရှည် ဆုံး နှင့်အ ကြီး ဆုံး မြစ် ဖြစ်ပါသည်။

Acknowledgments

We thank the anonymous reviewers and Action Editor, Minlie Huang, for their constructive and detailed feedback. We are also grateful to Miles Williams, Huiyin Xue, and Constantinos Karouzos for their valuable feedback on the initial draft. We acknowledge (1) IT Services at the University of Sheffield for providing high-performance computing services, (2) the EuroHPC Joint Undertaking for awarding us access to MeluXina at LuxProvide, Luxembourg, and (3) the use of time on Tier 2 HPC facility JADE2, funded by the Engineering and Physical Sciences Research Council (EPSRC) (EP/T022205/1). AY is supported by EPSRC [grant number EP/W524360/1] and the Japan Student Services Organization (JASSO) Student Exchange Support Program (Graduate Scholarship for Degree Seeking Students). AV research is partly supported by MRC-FAPESP [AIM-Health] and CNPq [406926/2025-5].

References

Abbasi, Mohammad Amin, Arash Ghafouri, Mahdi Firouzmandi, Hassan Naderi, and Behrouz Minaei Bidgoli. 2023. PersianLLaMA: Towards building first Persian large language model. *arXiv*, abs/2312.15713. https://doi.org/10.21203/rs.3.rs-3789059/v1

Agrawal, Sudhanshu, Wonseok Jeon, and Mingu Lee. 2024. AdaEDL: Early draft stopping for speculative decoding of large language models via an entropy-based lower bound on token acceptance probability. In *Proceedings of The 4th NeurIPS Efficient Natural Language and Speech Processing Workshop*, volume 262 of *Proceedings of Machine Learning Research*, pages 355–369.

Ahia, Orevaoghene, Sachin Kumar, Hila Gonen, Jungo Kasai, David Mortensen, Noah 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*, pages 9904–9923. https://doi.org/10.18653/v1/2023.emnlp-main.614

Ali, Mehdi, Michael Fromm, Klaudia Thellmann, Richard Rutmann, Max Lübbering, Johannes Leveling, Katrin Klug, Jan Ebert, Niclas Doll, Jasper Buschhoff, et al. 2024. Tokenizer choice for LLM training: Negligible or crucial? In *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 3907–3924. https://doi.org/10.18653/v1/2024.findings-naacl.247

Balachandran, Abhinand. 2023. Tamil-LLaMA: A new Tamil language model based on LLaMA 2. *arXiv*, abs/2311.05845. https://doi.org/10.48550/arXiv.2311.05845

Bandarkar, Lucas, 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. https://doi.org/10.18653/v1/2024.acl-long.44

Bojanowski, Piotr, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146. https://doi.org/10.1162/tacl_a_00051

Bostrom, Kaj 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. https://doi.org/10.18653/v1/2020.findings-emnlp.414

Cahyawijaya, Samuel, Holy Lovenia, Fajri Koto, Rifki Putri, Wawan Cenggoro, Jhonson Lee, Salsabil Akbar, Emmanuel Dave, Nuurshadieq Nuurshadieq, Muhammad Mahendra, et al. 2024. Cendol: Open instruction-tuned generative large language models for Indonesian languages. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 14899–14914. https://doi.org/10.18653/v1/2024.acl-long.796

Chau, Ethan C., Lucy H. Lin, and Noah A. Smith. 2020. Parsing with multilingual BERT, a small corpus, and a small treebank. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1324–1334. https://doi.org/10.18653/v1/2020.findings-emnlp.118

Choi, ChangSu, Yongbin Jeong, Seoyoon Park, Inho Won, HyeonSeok Lim, SangMin Kim, Yejee Kang, Chanhyuk Yoon, Jaewan Park, Yiseul Lee, et al. 2024. Optimizing language augmentation for multilingual

large language models: A case study on Korean. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 12514–12526.

Conneau, Alexis, 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. https://doi.org/10.18653/v1/2020.acl-main.747

Csaki, Zoltan, Pian Pawakapan, Urmish Thakker, and Qiantong Xu. 2023. Efficiently adapting pretrained language models to new languages. *arXiv*, abs/2311.05741. https://doi.org/10.48550/arXiv.2311.05741

Cui, Yiming, Ziqing Yang, and Xin Yao. 2023. Efficient and effective text encoding for Chinese LLaMA and Alpaca. *arXiv*, abs/2304.08177. https://doi.org/10.48550/arXiv.2304.08177

Da Dalt, Severino, 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.

Dagan, Gautier, Gabriel Synnaeve, and Baptiste Roziere. 2024. Getting the most out of your tokenizer for pre-training and domain adaptation. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 9784–9805.

DeepSeek-AI, Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, et al. 2025. DeepSeek-R1: Incentivizing reasoning capability in LLMs via reinforcement learning. *arXiv*, abs/2501.12948. https://doi.org/10.48550/arXiv.2501.12948

Dobler, Konstantin 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. https://doi.org/10.18653/v1/2023.emnlp-main.829

Dobler, Konstantin and Gerard de Melo. 2024. Language adaptation on a tight academic compute budget: Tokenizer swapping works and pure bfloat16 is enough. In *Proceedings of the 2nd Workshop on Advancing Neural Network Training: Computational Efficiency, Scalability, and Resource Optimization (WANT@ICML 2024)*.

Downey, C. M., Terra Blevins, Nora Goldfine, and Shane Steinert-Threlkeld. 2023. Embedding structure matters: Comparing methods to adapt multilingual vocabularies to new languages. In *Proceedings of the 3rd Workshop on Multi-lingual Representation Learning (MRL)*, pages 268–281. https://doi.org/10.18653/v1/2023.mrl-1.20

Dubey, Abhimanyu, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. 2024. The Llama 3 herd of models. *arXiv*, abs/2407.21783. https://doi.org/10.48550/arXiv.2407.21783

Evdaimon, Iakovos, Hadi Abdine, Christos Xypolopoulos, Stamatis Outsios, Michalis Vazirgiannis, and Giorgos Stamou. 2024. GreekBART: The first pretrained Greek sequence-to-sequence model. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 7949–7962.

Fujii, Kazuki, 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 *Proceedings of the First Conference on Language Modeling*.

Fujii, Takuro, Koki Shibata, Atsuki Yamaguchi, Terufumi Morishita, and Yasuhiro Sogawa. 2023. How do different tokenizers perform on downstream tasks in scriptio continua languages?: A case study in Japanese. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 4: Student Research Workshop)*, pages 39–49. https://doi.org/10.18653/v1/2023.acl-srw.5

Gee, Leonidas, 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. https://doi.org/10.18653/v1/2022.emnlp-industry.41

Gloeckle, Fabian, Badr Youbi Idrissi, Baptiste Roziere, David Lopez-Paz, and Gabriel Synnaeve. 2024. Better & faster large language models via multi-token prediction. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 15706–15734.

Groeneveld, Dirk, Iz Beltagy, Evan Walsh, Akshita Bhagia, Rodney Kinney, Oyvind Tafjord, Ananya Jha, Hamish Ivison, Ian Magnusson, Yizhong Wang, et al. 2024. OLMo: Accelerating the science of language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15789–15809. https://doi.org/10.18653/v1/2024.acl-long.841

Habib, Nathan, Clémentine Fourrier, Hynek Kydlícek, Thomas Wolf, and Lewis Tunstall. 2023. LightEval: A lightweight framework for LLM evaluation. https://github.com/huggingface/lighteval

Han, HyoJung, Akiko Eriguchi, Haoran Xu, Hieu Hoang, Marine Carpuat, and Huda Khayrallah. 2025. Adapters for altering LLM vocabularies: What languages benefit the most? In *Proceedings of the 13th International Conference on Learning Representations*.

Hasan, Tahmid, Abhik Bhattacharjee, Md. Saiful Islam, Kazi Mubasshir, Yuan-Fang Li, Yong-Bin Kang, M. Sohel Rahman, and Rifat Shahriyar. 2021. XL-Sum: Large-scale multilingual abstractive summarization for 44 languages. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 4693–4703. https://doi.org/10.18653/v1/2021.findings-acl.413

Hendrycks, Dan, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021. Measuring massive multitask language understanding. In *Proceedings of the 9th International Conference on Learning Representations*.

Hofmann, Valentin, Hinrich Schuetze, and Janet Pierrehumbert. 2022. An embarrassingly simple method to mitigate undesirable properties of pretrained language model tokenizers. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 385–393. https://doi.org/10.18653/v1/2022.acl-short.43

Hong, Jimin, Gibbeum Lee, and Jaewoong Cho. 2024. Accelerating multilingual language model for excessively tokenized languages. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 11095–11111. https://doi.org/10.18653/v1/2024.findings-acl.660

Hu, Edward J., 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 *Proceedings of the 10th International Conference on Learning Representations*.

Hu, Michael Y., Aaron Mueller, Candace Ross, Adina Williams, Tal Linzen, Chengxu Zhuang, Leshem Choshen, Ryan Cotterell, Alex Warstadt, and Ethan Gotlieb Wilcox, editors. 2024. *The 2nd BabyLM Challenge at the 28th Conference on Computational Natural Language Learning*. Association for Computational Linguistics.

Huang, Kaixuan, Xudong Guo, and Mengdi Wang. 2025. SpecDec++: Boosting speculative decoding via adaptive candidate lengths. In *Proceedings of the Second Conference on Language Modeling*.

Ji, Shaoxiong, Zihao Li, Jaakko Paavola, Indraneil Paul, Hengyu Luo, and Jörg Tiedemann. 2025. Massively multilingual adaptation of large language models using bilingual translation data. *arXiv*, abs/2506.00469. https://doi.org/10.48550/arXiv.2506.00469

Ji, Shaoxiong, Zihao Li, Indraneil Paul, Jaakko Paavola, Peiqin Lin, Pinzhen Chen, Dayyán O'Brien, Hengyu Luo, Hinrich Schütze, Jörg Tiedemann, and Barry Haddow. 2024. EMMA-500: Enhancing massively multilingual adaptation of large language models. *arXiv*, abs/2409.17892. https://doi.org/10.48550/arXiv.2409.17892

Jiang, Albert Qiaochu, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de Las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. 2023. Mistral 7B. *arXiv*, abs/2310.06825. https://doi.org/10.48550/arXiv.2310.06825

Kim, Seungduk, Seungtaek Choi, and Myeongho Jeong. 2024. Efficient and effective vocabulary expansion towards

multilingual large language models. *arXiv*, abs/2402.14714. https://doi.org/10.48550/arXiv.2402.14714

Koto, Fajri, Jey Han Lau, and Timothy Baldwin. 2021. IndoBERTweet: A pretrained language model for Indonesian Twitter with effective domain-specific vocabulary initialization. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 10660–10668. https://doi.org/10.18653/v1/2021.emnlp-main.833

Kudo, Taku. 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. https://doi.org/10.18653/v1/P18-1007

Kudo, Taku and John Richardson. 2018. SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 66–71. https://doi.org/10.18653/v1/D18-2012

Larcher, Celio H. N., Marcos Piau, Paulo Finardi, Pedro Gengo, Piero Esposito, and Vinicius F. Caridá. 2023. Cabrita: Closing the gap for foreign languages. *arXiv*, abs/2308.11878. https://doi.org/10.48550/arXiv.2308.11878

Lhoest, Quentin, Albert Villanova del Moral, Yacine Jernite, Abhishek Thakur, Patrick von Platen, Suraj Patil, Julien Chaumond, Mariama Drame, Julien Plu, Lewis Tunstall, et al. 2021. Datasets: A community library for natural language processing. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 175–184. https://doi.org/10.18653/v1/2021.emnlp-demo.21

Lin, Chin Yew. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81.

Lin, Peiqin, Shaoxiong Ji, Jörg Tiedemann, André F. T. Martins, and Hinrich Schütze. 2024. MaLA-500: Massive language adaptation of large language models. *arXiv*, abs/2401.13303. https://doi.org/10.48550/arXiv.2401.13303

Mangrulkar, Sourab, Sylvain Gugger, Lysandre Debut, Younes Belkada, Sayak Paul, and Benjamin Bossan. 2022. PEFT: State-of-the-art parameter-efficient fine-tuning methods. https://github.com/huggingface/peft

Maynez, Joshua, Priyanka Agrawal, and Sebastian Gehrmann. 2023. Benchmarking large language model capabilities for conditional generation. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 9194–9213. https://doi.org/10.18653/v1/2023.acl-long.511

Minixhofer, Benjamin, 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. https://doi.org/10.18653/v1/2022.naacl-main.293

Minixhofer, Benjamin, Edoardo M. Ponti, and Ivan Vulić. 2024. Zero-shot tokenizer transfer. In *Advances in Neural Information Processing Systems*, volume 37, pages 46791–46818. Curran Associates, Inc. https://doi.org/10.52202/079017-1484

Moi, Anthony and Nicolas Patry. 2023. HuggingFace's Tokenizers. https://github.com/huggingface/tokenizers

Muller, Benjamin, Antonios Anastasopoulos, Benoît Sagot, and Djamé Seddah. 2021. When being unseen from mBERT is just the beginning: Handling new languages with multilingual language models. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 448–462. https://doi.org/10.18653/v1/2021.naacl-main.38

Mundra, Nandini, Aditya Nanda Kishore Khandavally, Raj Dabre, Ratish Puduppully, Anoop Kunchukuttan, and Mitesh M Khapra. 2024. An empirical comparison of vocabulary expansion and initialization approaches for language models. In *Proceedings of the 28th Conference on Computational Natural Language Learning*, pages 84–104. https://doi.org/10.18653/v1/2024.conll-1.8

Nguyen, Xuan Phi, Wenxuan Zhang, Xin Li, Mahani Aljunied, Zhiqiang Hu, Chenhui Shen, Yew Ken Chia, Xingxuan Li, Jianyu Wang, Qingyu Tan, et al. 2024. SeaLLMs - large language models for Southeast Asia. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics*

(Volume 3: System Demonstrations), pages 294–304. https://doi.org/10.18653/v1/2024.acl-demos.28

NLLB Team, Marta R. Costa-jussà, James Cross, Onur Çelebi, Maha Elbayad, Kenneth Heafield, Kevin Heffernan, Elahe Kalbassi, Janice Lam, Daniel Licht, et al. 2022. No language left behind: Scaling human-centered machine translation. *arXiv*, abs/2207.04672. https://doi.org/10.48550/arXiv.2207.04672

OpenAI. 2023. GPT-4 technical report. *arXiv*, abs/2303.08774. https://doi.org/10.48550/arXiv.2303.08774

OpenAI, Aaron Hurst, Adam Lerer, Adam P. Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, A. J. Ostrow, Akila Welihinda, Alan Hayes, et al. 2024. GPT-4o system card. *arXiv*, abs/2410.21276. https://doi.org/10.48550/arXiv.2410.21276

Ostendorff, Malte and Georg Rehm. 2023. Efficient language model training through cross-lingual and progressive transfer learning. *arXiv*, abs/2301.09626. https://doi.org/10.48550/arXiv.2301.09626

Paszke, Adam, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. 2019. PyTorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Petrov, Aleksandar, Emanuele La Malfa, Philip Torr, and Adel Bibi. 2023. Language model tokenizers introduce unfairness between languages. In *Advances in Neural Information Processing Systems*, volume 36, pages 36963–36990.

Pipatanakul, Kunat, Phatrasek Jirabovonvisut, Potsawee Manakul, Sittipong Sripaisarnmongkol, Ruangsak Patomwong, Pathomporn Chokchainant, and Kasima Tharnpipitchai. 2023. Typhoon: Thai large language models. *arXiv*, abs/2312.13951. https://doi.org/10.48550/arXiv.2312.13951

Popović, Maja. 2015. chrF: character n-gram F-score for automatic MT evaluation. In *Proceedings of the Tenth Workshop on Statistical Machine Translation*, pages 392–395. https://doi.org/10.18653/v1/W15-3049

Remy, François, 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 *Proceedings of the First Conference on Language Modeling*.

Riviere, Morgane, Gemma Team, Shreya Pathak, Pier Giuseppe Sessa, Cassidy Hardin, Surya Bhupatiraju, Léonard Hussenot, Thomas Mesnard, Bobak Shahriari, Alexandre Ramé, et al. 2024. Gemma 2: Improving open language models at a practical size. *arXiv*, abs/2408.00118. https://doi.org/10.48550/arXiv.2408.00118

Rust, Phillip, Jonas Pfeiffer, Ivan Vulić, 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 (Volume 1: Long Papers)*, pages 3118–3135. https://doi.org/10.18653/v1/2021.acl-long.243

Scialom, Thomas, Paul-Alexis Dray, Sylvain Lamprier, Benjamin Piwowarski, and Jacopo Staiano. 2020. MLSUM: The multilingual summarization corpus. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8051–8067. https://doi.org/10.18653/v1/2020.emnlp-main.647

Sellam, Thibault, Dipanjan Das, and Ankur Parikh. 2020. BLEURT: Learning robust metrics for text generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7881–7892. https://doi.org/10.18653/v1/2020.acl-main.704

Sennrich, Rico, 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. https://doi.org/10.18653/v1/P16-1162

Singh, Shivalika, Angelika Romanou, Clémentine Fourrier, David Ifeoluwa Adelani, Jian Gang Ngui, Daniel Vila-Suero, Peerat Limkonchotiwat, Kelly Marchisio, Wei Qi Leong, Yosephine Susanto, et al. 2025. Global MMLU: Understanding and addressing cultural and linguistic biases in multilingual evaluation. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long*

Papers), pages 18761–18799. https://doi.org/10.18653/v1/2025.acl-long.919

Sun, Jimin, Patrick Fernandes, Xinyi Wang, and Graham Neubig. 2023. A multi-dimensional evaluation of tokenizer-free multilingual pretrained models. In *Findings of the Association for Computational Linguistics: EACL 2023*, pages 1725–1735. https://doi.org/10.18653/v1/2023.findings-eacl.128

Tang, Tianyi, Wenyang Luo, Haoyang Huang, Dongdong Zhang, Xiaolei Wang, Xin Zhao, Furu Wei, and Ji-Rong Wen. 2024. Language-specific neurons: The key to multilingual capabilities in large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5701–5715. https://doi.org/10.18653/v1/2024.acl-long.309

Tejaswi, Atula, Nilesh Gupta, and Eunsol Choi. 2024. Exploring design choices for building language-specific LLMs. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 10485–10500. https://doi.org/10.18653/v1/2024.findings-emnlp.614

Timor, Nadav, Jonathan Mamou, Daniel Korat, Moshe Berchansky, Gaurav Jain, Oren Pereg, Moshe Wasserblat, and David Harel. 2025. Accelerating LLM inference with lossless speculative decoding algorithms for heterogeneous vocabularies. In *Proceedings of the 42nd International Conference on Machine Learning*.

Toraman, Cagri, Eyup Halit Yilmaz, Furkan Sahinuç, and Oguzhan Ozcelik. 2023. Impact of tokenization on language models: An analysis for Turkish. *ACM Transactions on Asian and Low-Resource Language Information Processing*, 22(4). https://doi.org/10.1145/3578707

Touvron, Hugo, Louis Martin, Kevin R. Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. LLaMA 2: Open foundation and fine-tuned chat models. *arXiv*, abs/2307.09288. https://doi.org/10.48550/arXiv.2307.09288

Warstadt, Alex, Aaron Mueller, Leshem Choshen, Ethan Wilcox, Chengxu Zhuang, Juan Ciro, Rafael Mosquera, Bhargavi Paranjabe, Adina Williams, Tal Linzen, and Ryan Cotterell. 2023. Findings of the BabyLM challenge: Sample-efficient pretraining on developmentally plausible corpora. In *Proceedings of the BabyLM Challenge at the 27th Conference on Computational Natural Language Learning*, pages 1–34. https://doi.org/10.18653/v1/2023.conll-babylm.1

Wendler, Chris, Veniamin Veselovsky, Giovanni Monea, and Robert West. 2024. Do llamas work in English? On the latent language of multilingual transformers. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15366–15394. https://doi.org/10.18653/v1/2024.acl-long.820

Wolf, Thomas, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, et al. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45. https://doi.org/10.18653/v1/2020.emnlp-demos.6

Xue, Linting, Noah Constant, Adam Roberts, Mihir Kale, Rami Al-Rfou, Aditya Siddhant, Aditya Barua, and Colin Raffel. 2021. 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. https://doi.org/10.18653/v1/2021.naacl-main.41

Yamaguchi, Atsuki, Terufumi Morishita, Aline Villavicencio, and Nikolaos Aletras. 2025. Adapting chat language models using only target unlabeled language data. *Transactions on Machine Learning Research*.

Yamaguchi, Atsuki, Aline Villavicencio, and Nikolaos Aletras. 2024. An empirical study on cross-lingual vocabulary adaptation for efficient language model inference. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 6760–6785. https://doi.org/10.18653/v1/2024.findings-emnlp.396

Yao, Yunzhi, Shaohan Huang, Wenhui Wang, Li Dong, and Furu Wei. 2021. Adapt-and-distill: Developing small, fast and effective pretrained language models for domains. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 460–470. https://doi.org/10.18653/v1/2021.findings-acl.40

Yong, Zheng Xin, Hailey Schoelkopf, Niklas Muennighoff, Alham Fikri Aji, David

Ifeoluwa Adelani, Khalid Almubarak, M. Saiful Bari, Lintang Sutawika, Jungo Kasai, et al. 2023. BLOOM+1: Adding language support to BLOOM for zero-shot prompting. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 11682–11703. https://doi.org/10.18653/v1/2023.acl-long.653