FlexiTokens: Flexible Tokenization for Evolving Language Models

Anonymous Authors[1]

Abstract

Language models (LMs) are challenging to adapt to new data distributions by simple finetuning due to the rigidity of their subword tokenizers, which typically remain unchanged during adaptation. This inflexibility often leads to inefficient tokenization, causing overfragmentation of out-of-distribution domains, unseen languages, or scripts. In this work, we develop byte-level LMs with learnable tokenizers to make tokenization adaptive. Our models include a submodule that learns to predict boundaries between the input byte sequence, encoding it into variable-length segments. Existing tokenizer-free methods train this boundary predictor using an auxiliary loss that enforces a fixed compression rate across the training corpus, introducing a new kind of rigidity. We propose FlexiTokens, a simplified training objective that enables significantly greater flexibility during adaptation. Evaluating across multiple multilingual benchmarks, morphologically diverse tasks, and domains, we demonstrate that FlexiTokens consistently reduces token over-fragmentation and achieves up to 10% improvements on downstream task performance compared to subword and other gradient-based tokenizers.

1. Introduction

Tokenization—the process of segmenting text into discrete units—has been shown to significantly influence language model performance (Ali et al., 2024; Geiping et al., 2024; Land & Bartolo, 2024). Widely used subword tokenization algorithms (Sennrich et al., 2016; Devlin et al., 2019) often overfragment sequences in unseen domains, languages, and scripts. This oversegmentation not only leads to poor downstream performance, increased sequence lengths contribute to higher computational overhead, memory usage, and inference costs (Ahia et al., 2023; Petrov et al., 2023). In addition, such tokenizers are inherently static and tightly coupled with the language model; they do not adapt when the language model is finetuned. As a result, even if a model is adapted to a new distribution, its tokenization remains fixed, limiting its performance, e.g., fine-tuning Llama 2 models is subpar for coding tasks (Dagan et al., 2024; Minixhofer et al., 2024), and unseen scripts (Li et al., 2023).

Eliminating the reliance on static subword tokenizers has, thus, gained momentum in recent literature by directly modeling bytes (Xue et al., 2022; Al-Rfou et al., 2018; Wang et al., 2024). To address the increase in sequence length in byte-level language models, various papers introduce a tokenization module within the LM to segment bytes into patches (Tay et al., 2021; Nawrot et al., 2022b; Ahia et al., 2024; Pagnoni et al., 2024; Nawrot et al., 2023; YU et al., 2023). As opposed to subword tokenizers, this module is typically learned via gradients along with the LM with an auxiliary loss to achieve a desired *compression rate* of the input sequence. This compression rate, while controllable, is predetermined and fixed during pretraining, which again hampers adaptation to new distributions (see Figure 1). For example, an LM trained with a fixed compression rate on a general domain may over-tokenize samples in specialized domains like Medicine or morphologically rich languages like Turkish that contain longer words. Conversely, it may undertokenize samples in programming languages or logographic languages like Chinese where distinct semantic units may be inappropriately merged.

To enable flexible adaptation of gradient-based tokenizers, we propose a new training objective, which relaxes the need to have a *fixed* compression rate. Instead of an expected compression rate, we define a lower bound on the compression rate that every input sequence should have. We introduce a hinge-like loss to optimize the tokenizer with this rate. By not penalizing the tokenizer when the compression rate is higher than this rate, our method allows for the segmentation to be flexible to the input sequence. When the LM is fine-tuned, this loss allows the tokenization to effectively adjust to the target distribution without leading to overfragmentation. We call our method FlexiTokens.

We evaluate our proposed approach on multiple multilin-

[1] Anonymous Institution, Anonymous City, Anonymous Region, Anonymous Country. Correspondence to: Anonymous Author <anon.email@domain.com>.

Preliminary work. Under review by the International Conference on Machine Learning (ICML). Do not distribute.

gual benchmarks and morphologically diverse tasks (Table 4). FlexiTokens consistently shows superior performance compared to baselines while improving average compression rate thereby improving inference runtime. We also show that while maintaining a fairer fragmentation rate across all our pretraining languages, FlexiTokens can be easily adapted to unseen languages and scripts without leading to overfragmentation. Our analysis shows that our method often updates the tokenizer to recover semantically meaningful tokens relevant to the task or domain after adaptation whereas the baselines, being not updatable, overtokenize.

2. FlexiTokens

We build a byte-level LM with a learnable tokenization module integrated within the model. FlexiTokens allows the model to adjust its learned tokenization strategy to the structure and distribution of the task and input data. Our model uses *hourglass transformers* (Nawrot et al., 2022a) as backbone, originally introduced to efficiently handle long sequences in tokenizer-free models (Nawrot et al., 2023; Ahia et al., 2024). Despite being learnable, the resulting tokenization modules in prior work remain bound to the decisions made during pretraining, even when the model is trained or finetuned further. This inherently limits their ability to adapt to new domains, languages, or evolving data distributions, where the originally learned segmentation might no longer be optimal.[1] Below, we describe the key components of the hourglass architecture (§2.1) and introduce the modifications we make to enable dynamic and equitable tokenization (§2.2).

2.1. Hourglass Architecture

The hourglass architecture (Nawrot et al., 2022a) was designed to scale byte-level language models to handle long sequences by incorporating an internal tokenization process. It consists of three modules; a tokenization submodule, a language modeling block, and an upsampling layer.

The tokenization submodule processes input byte sequences using a lightweight transformer that maps each byte in an input byte sequence $x_1, \ldots, x_N$ to hidden states. A boundary predictor then estimates the probability $\hat{b}_t \in [0,1]$ of predicting a segment boundary at each position t. It is implemented using an MLP followed by a sigmoid function. To obtain discrete boundary decisions $b_t \in \{0,1\}$ while preserving differentiability, we employ a hard Gumbel sigmoid re□parameterization of the Bernoulli distribution. Since this module is differentiable, the segmentations are learned along with the rest of the model.

Given the predicted boundaries, the **language modeling module** pools hidden states between segment boundaries to construct a sequence of token-level representations. These representations are then passed through the middle block of transformer layers to obtain another sequence of hidden representations.

Finally, the **upsampling module** converts the outputs from the middle LM block to byte-level probabilities. The token-level representations from the middle block are first upsampled to match the original input resolution via duplication and combined with initial byte-level representations using skip connections. These are then passed through a lightweight transformer, an unembedding layer, and a softmax to compute the language modeling loss. We refer the read to (Nawrot et al., 2023) for a detailed description.

To prevent the boundary predictor from collapsing and trivially predicting each position t as a boundary, prior work (Nawrot et al., 2023; Ahia et al., 2024) added a regularizer to the LM objective: $-\log \mathrm{Binomial}(\alpha; N, k)$ where,

$$\mathrm{Binomial}(\alpha; N, k) = \binom{N}{k}\alpha^k(1-\alpha)^{N-k}, \quad \text{and} \quad k = \sum_N b_t \quad (1)$$

$\alpha \in [0,1]$ is a hyperparameter that controls the expected boundary rate. This loss is lowest when k is close to αN which is the mode of the Binomial distribution. In other words, α controls the compression rate of the input sequence to approximately $\frac{1}{\alpha}\times$. Setting $\alpha = 0$ will cause no boundaries to be predicted and with $\alpha = 1$, the model learns to predict every position to be a boundary. This loss is added to a cross-entropy for next-byte prediction to train the model and tokenizer in an end-to-end fashion.

2.2. FlexiTokens

In contrast with subword based models like BPE, LMs with gradient-based tokenization can learn to segment input text in a way that best represents the underlying data distribution. Furthermore, prior work has shown that it allows better controllability over segmentation rates over different languages when training multilingual models by simply employing different boundary predictors with different compression rates per language or script (Ahia et al., 2024) leading to more equitable tokenization (Petrov et al., 2023). However, even within a language, different subsets such as different domains might require different compression rates to optimally encode the input. But the expected compression rate is predetermined by the hyperparameter α with little room for variation. Furthermore, when adapting the LM to new distributions such as a new domain or a new language, bound by the binomial loss in Equation 1, the compression rate does not update to the requirements of the target distribution.

[1] This issue is also present in subword tokenizers like BPE. Prior work typically handles this issue with heuristics like retraining and replacing the entire tokenizer during adaptation (Minixhofer et al., 2024).

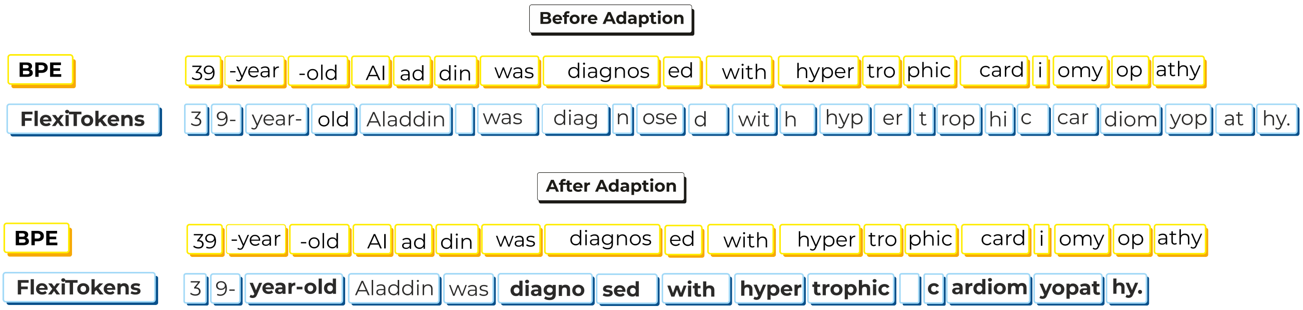
Before Adaption
BPE
39 -year -old Al ad din was diagnos ed with hyper tro phic card i omy op athy
FlexiTokens
3 9- year- old Aladdin was diag n ose d wit h hyp er t rop hi c car diom yop at hy.
After Adaption
BPE
39 -year -old Al ad din was diagnos ed with hyper tro phic card i omy op athy
FlexiTokens
3 9- year-old Aladdin was diagno sed with hyper trophic c ardiom yopat hy.

Figure 1: We present an example of tokenized medical text, where FlexiTokens produces a less fragmented sequence of tokens than BPE. Unlike BPE which applies a fixed tokenization, FlexiTokens adapts its tokenization to the medical domain, capturing domain-specific patterns more effectively.

The ideal solution to address this issue is to get rid of the hyperparameter α (and the binomial loss) and simply minimize the predicted number of boundaries per byte, that is, $\frac{k}{N}$. If optimized well, this loss will find the right balance between compression and minimizing the LM loss. However, in our early experiments, we observe that this loss quickly decreases to 0, predicting no boundaries. To prevent this behavior, we modify this loss to

$$\max\left(\frac{k}{N} - \beta, 0\right), \text{ where } \beta = \alpha - \lambda\sigma \leq \frac{k}{N} \leq \alpha \quad (2)$$

σ represents the standard deviation of tokenization rates over multiple samples in a given language. λ is a hyperparameter. This loss introduces a lower bound on the boundary rate at $\alpha - \lambda\sigma$. If the boundary rate reduces to less than this value, this loss will become 0 reducing further incentive to compress but does not penalize it. In contrast, the binomial loss forces the rate to be close to α penalizing both increase or decrease. Indeed, we observe in our experiments that there is higher variance in the segmentation rates of different samples. Furthermore, during finetuning, we observe changes in the compression rates showing that the tokenization indeed adapts to the task. We refer to the flexible tokens learned through our proposed loss and the resulting model that predicts flexible tokens as FlexiTokens.[2]

To encode the same information, different languages require different number of bytes, where non-Latin languages (e.g., Indian languages) may require up to 4 bytes per character. When training multilingual models, setting one α for all languages will lead to text in some languages getting segmented into much longer sequences. To alleviate this issue, Ahia et al. (2024) proposed adding a different boundary predictor per language with its own α defined to make the compression rates uniform across languages. A unique boundary predictor per language, however, requires determining or predicting the input language to route the input to the appropriate predictor. It also makes it challenging when the input text contains multiple languages (in case of code-mixed text). Our experiments reveal that training one shared boundary predictor with a different hyperparameter α_L for each language L leads to the same performance. Hence, we train a multilingual model with the following training objective objective.

$$\text{L} = \sum_{i=1}^{N} -\log p_\theta(x_i \mid x_{<i}) - \sum_{\mathcal{M}} \mathbb{I}(\text{language}(\mathbf{x}) = L) \max\left(\frac{k}{N} - \beta_L, 0\right) \quad (3)$$

where $\mathcal{M}$ is the set of all languages in the training set.

Determining β_L We define an anchor language A[3] and set α_A as a hyperparameter. We assume access to an n-way parallel corpus[4] between A and every other language L in our training set.[5] We compute the mean sequence length (in bytes) μ_A, μ_L and standard deviation σ_A, σ_L over this dataset. We set α_L to be $\alpha_A \frac{\mu_A}{\mu_L}$, and define the lower bound β_L as $\alpha_L - \lambda\sigma_L$. Intuitively, if L uses more bytes to represent the same information as A, its compression rate should be higher (and hence α lower).

3. Experimental Setup

3.1. Datasets

We validate our proposed approach in a multilingual setting. We train models with four scripts and six languages: Latin script (English and Spanish), Cyrillic (Russian and Ukrainian), Devanagari (Hindi), and Telugu script (Telugu). These scripts cover a diverse range of typologies and byte complexities. For example, Latin script needs 1 byte per character in Unicode, whereas Russian and Telugu characters need up to 2 and 3 bytes respectively. To make tokeniza-

[2] We use the term interchangeably to refer to our model and proposed loss.

[3] We choose A as English in all our experiments. This choice is arbitrary; choosing another language will change the β values but will not influence the final results.

[4] This computation can also be done with pairwise parallel dataset with the anchor language with slight modifications.

[5] This parallel dataset is not used for training the model.

Table 1: α_L and σ_L values for each language in our training dataset, computed using FLORES-200. The upper bound β_L in Equation 3 is computed as $\alpha_L - \lambda\sigma_L$)

Configuration	en	es	ru	uk	hi	te
FlexiTokens 10×	0.1 / 10	0.08 / 12.12	0.05 / 19.92	0.053 / 18.70	0.039 / 25.62	0.037 / 26.91
FlexiTokens 5×	0.2 / 5	0.17 / 6.06	0.1 / 9.96	0.107 / 9.35	0.078 / 12.81	0.074 / 13.45
FlexiTokens 3×	0.333 / 3	0.28 / 3.64	0.167 / 5.98	0.178 / 5.61	0.13 / 7.68	0.124 / 8.07
σ	0.023	0.019	0.011	0.012	0.009	0.008

tion rates similar across all languages, all these languages require different amounts of compression.

For pretraining, we sample the first 2.06M documents from FineWeb (Penedo et al., 2024a) for English and Spanish, using the first 10K documents as the validation set. For all other languages, we sample the first 1.65M documents from FineWeb 2 (Penedo et al., 2024b), again using the first 10K documents for validation. A breakdown of the training set sizes is shown in Figure 5 (in Appendix D).

For downstream evaluations, we finetune on the following tasks: (1) *XNLI* (Conneau et al., 2018): natural language inference, (2) *SIB-200* (Adelani et al., 2023): topic classification, (3) *Multilingual Sentiment* (clapAI, 2024): multi-domain sentiment analysis, (4) *WikiANN* (Pan et al., 2017): named entity recognition, (5) *Indo-Aryan Language Identification (ILI)*[6] (Zampieri et al., 2018): dialect classification, (6) *Medical Abstracts Text Classification* (Schopf et al., 2022) and (7) *Irony detection* in Tweets containing emojis (Rohanian et al., 2018) We provide more details on each dataset in Appendix D.

3.2. Hyperparameters

To understand the impact of sequence compression on model's performance, we explore multiple compression rate configurations. Our main results use 3× compression rate for our anchor language, English (i.e. $\alpha = 1/3$). We also compare with 5× and 10×. The corresponding values of α_L and σ_L for all languages is in Table 1. We compute β_L using the FLORES-200 dataset (Costa-Jussà et al., 2022), which contains parallel sentences in 200 languages. We empirically set $\lambda = 3$; we show comparisons with other values in §5. In our experiment with adapting our model to an unseen script (for Urdu), we set it β to have the same value as Telugu, which has the highest compression rate of all the languages we experimented on, assuming no available training dataset in the unseen language.

Model Architecture and Pretraining We pretrain a model with $119M$ parameters. We follow Ahia et al. (2024) to create a 16-layer hourglass transformer. The tokenization and upsampling submodules each consist of 2 transformer layers, while the language modeling submodule contains 12 transformer layers. The input embedding dimension is 768. All transformer layers have a hidden size of 768, with a feed-forward intermediate dimension of 3072, and we use 12 attention heads in the self-attention mechanism. All other parameters follow Ahia et al. (2024), except for the boundary predictor: instead of multiple predictors, we use a single 2-layer MLP as the boundary predictor.

During pretraining, we use a chunk size of 512 bytes. We train for 100K steps with a cumulative batch size of 512 across 2 H100 GPUs with 9000 warmup steps. Optimization is performed with Adam (Kingma & Ba, 2014), a cosine learning rate scheduler (with maximum learning rate of 5e-5), and gradient clipping set to 0.25.

Finetuning During finetuning, we increase the sequence length to 2048 bytes to better capture longer sequences in the finetuning dataset.[7] For the NER task, we first concatenate token sequences using whitespaces before tokenization and label whitespaces as non-entity. We set gradient clipping to 1.0 and apply a warmup ratio of 10%. All tasks are finetuned for 5 epochs, using task-specific batch sizes (Table 2) based on data availability. We perform monolingual finetuning on each language.

3.3. Baselines

We consider two baselines: (1) a model trained with a BPE tokenizer and (2) a byte-level model whose boundary predictor is trained with a binomial loss as described in Nawrot et al. [2023] (Nawrot et al., 2023) (binomial). For fair comparison with the BPE-based model, we match its overall parameter size with FlexiTokens. We train a BPE tokenizer with a vocab size of 50K on the same amount of dataset from each language. This achieves a compression rate of

[6] `https://github.com/kmi-linguistics/vardial2018`

[7] We use a shorter sequence length during pretraining due to computational constraints.

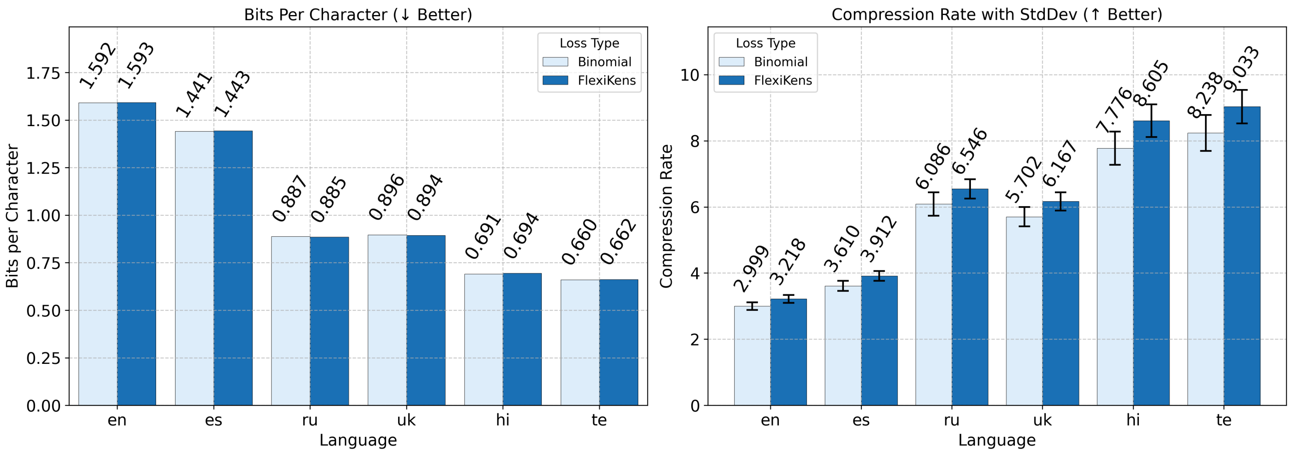

Figure 2: FineWeb Test BPB (↓), Compression rate (↑) and Compression variance (↑) of FlexiTokens compared to the binomial variant with $\alpha_A = 0.3$ and $\lambda = 3$. Higher compression rates result in fewer tokens, which in turn leads to a more efficient model.

Table 2: Batch Sizes per Dataset and Language

Dataset	en	es	ru	uk	hi	te	ur
XNLI	64	64	64	64	64	64	64
SIB-200	8	8	8	8	8	8	-
WikiANN	16	16	16	16	16	16	-
Multi. Sentiment	128	32	32	-	8	-	-
ILI	-	-	-	-	32	-	-
Medical Abstract	16	-	-	-	-	-	-
Irony detection	32	-	-	-	-	-	-

4.4× on English.[8] To match total parameters (embeddings + transformer layers), we train the language model with 5 Transformer layers.[9]

4. Results and Analyses

We evaluate our pretrained model using bits per byte (BPB) (Graves, 2013) and the finetuned models using task specific metrics, mostly accuracy and F1-score. We provide a summary of the results for the pretrained models in Figure 2 and Figure 3, and for the finetuned models inTable 3, Table 4, and Figure 4, with details in Appendix E.

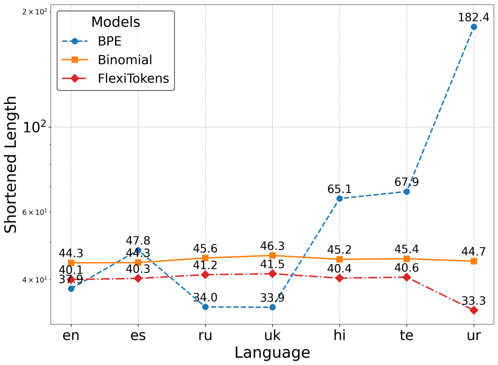

Figure 3: Average number of tokens per sample obtained in the FLORES dataset with different tokenization algorithms. FlexiTokens consistently produces the least number of tokens while maintaining balance across languages, even for the unseen language Urdu. BPE over-fragments seen (Hindi, Telugu) as well as unseen languages (Urdu).

Pretraining with FlexiTokens leads to better compression As shown in Figure 2, our method maintains the BPB performance as binomial on the FineWeb test sets while achieving a substantially higher average compression rate, which in turn increases inference speed by requiring fewer tokens.

We also observe a higher variance in compression rates of FlexiTokens implying higher flexibility in how input sequences are fragmented. This variation—which is much lower in baseline models—alongside the higher compression rate on average underscores FlexiTokens' ability to dynamically adapt its tokenization patterns to its input. In Figure 3, we compare average number of tokens required

[8] Note that BPE models cannot be controlled to have desired compression rates across all languages due to their inherent frequency based training process (Ahia et al., 2023).

[9] We conducted early experiments with training BPE-based models by matching English's compression rate to 3× compression rate but they resulted in vocabulary sizes of 10K which performed poorly in early experiments.

Table 3: WikiANN (NER), XNLI and SIB-200 F1 Score and Accuracy and for 3× Compression Rate. FlexiTokens outperforms all baselines on XNLI and NER respectively. Notably, it achieves approximately a 3 point gain on XNLI for Urdu—an unseen language script—compared to BPE.

Model	**en**	**es**	**ru**	**uk**	**hi**	**te**	**Avg**
NER F1 Score							
BPE	52.30	67.7 0	64.94	74.99	60.23	48.18	61.39
binomial	63.80	75.06	67.59	**78.06**	61.21	48.31	65.67
FlexiTokens $\lambda 1$	63.07	76.12	**68.30**	77.94	**62.26**	**51.74**	**66.57**
FlexiTokens $\lambda 2$	**63.96**	**76.23**	67.55	77.99	62.24	48.13	66.02
FlexiTokens $\lambda 3$	63.73	75.45	68.25	78.01	61.97	50.88	66.38
XNLI Accuracy							
Model	**en**	**es**	**ru**	**hi**	**te**	**ur** (OOD)	**Avg**
BPE	73.09	69.9	65.95	61.48	**68.00**	54.11	65.42
binomial	72.87	70.28	65.93	62.26	66.11	54.79	65.37
FlexiTokens $\lambda 1$	**73.51**	70.22	66.47	**62.42**	67.11	56.99	66.12
FlexiTokens $\lambda 2$	73.21	**70.84**	**66.97**	62.16	66.71	**57.58**	**66.25**
FlexiTokens $\lambda 3$	73.35	70.22	66.75	62.36	67.82	57.33	66.31
SIB-200 Accuracy							
Model	**en**	**es**	**ru**	**uk**	**hi**	**te**	**Avg**
BPE	**80.88**	**81.37**	**81.37**	**76.96**	60.78	**72.55**	**75.65**
binomial	79.41	74.02	71.08	68.63	64.71	69.61	71.24
FlexiTokens $\lambda 1$	78.92	72.55	75.49	69.61	61.27	66.18	70.67
FlexiTokens $\lambda 2$	77.94	75.98	74.51	71.57	69.12	66.18	72.55
FlexiTokens $\lambda 3$	**80.88**	77.45	73.04	72.55	71.08	**71.08**	74.35

to represent the same information in different languages by different tokenization methods. Our method remains as equitable as binomial using a similar number of tokens for all languages. In comparison, BPE shows high variability with included languages like Hindi and Telugu requiring twice as many tokens. An unseen language (Urdu) requires 6 times as much.

Table 4: Accuracy on ILI, Medical Abstracts, and Irony tasks. FlexiTokens outperforms across all tasks.

Model	**ILI (hi)**	**Med. Abs. (en)**	**Irony (en)**
BPE	89.06	57.68	67.86
binomial	89.47	62.81	67.60
FlexiTokens $\lambda 1$	89.58	62.92	68.37
FlexiTokens $\lambda 2$	**90.33**	62.74	68.75
FlexiTokens $\lambda 3$	89.55	**63.19**	**69.26**

FlexiTokens adapts tokenization and boosts performance across tasks and domains. In Tables 4 and 3, we report task-specific metrics after finetuning our pretrained models on several downstream tasks across different domains and the corresponding compression rates per language and task in Figure 4. FlexiTokens outperforms all baselines on majority of tasks, even the BPE baseline with a much higher compression rate. Our method obtains performance improvements of up to 4 absolute points on some tasks compared with binomial while improving compression across all tasks. Moreover, as we increase λ, performance tends to also increase. This is because a higher λ allows a wider margin for model to find the optimal compression rate resulting in an up to 2 points improvements in some tasks.

Analyzing compression rates across tasks and languages in Figure 4, we observe that binomial maintains rates closer to the initial α, but this effect diminishes for non-Latin languages such as Hindi and Telugu, which are structurally distant from Latin scripts. These languages show both higher average compression and greater variance with FlexiTokens.

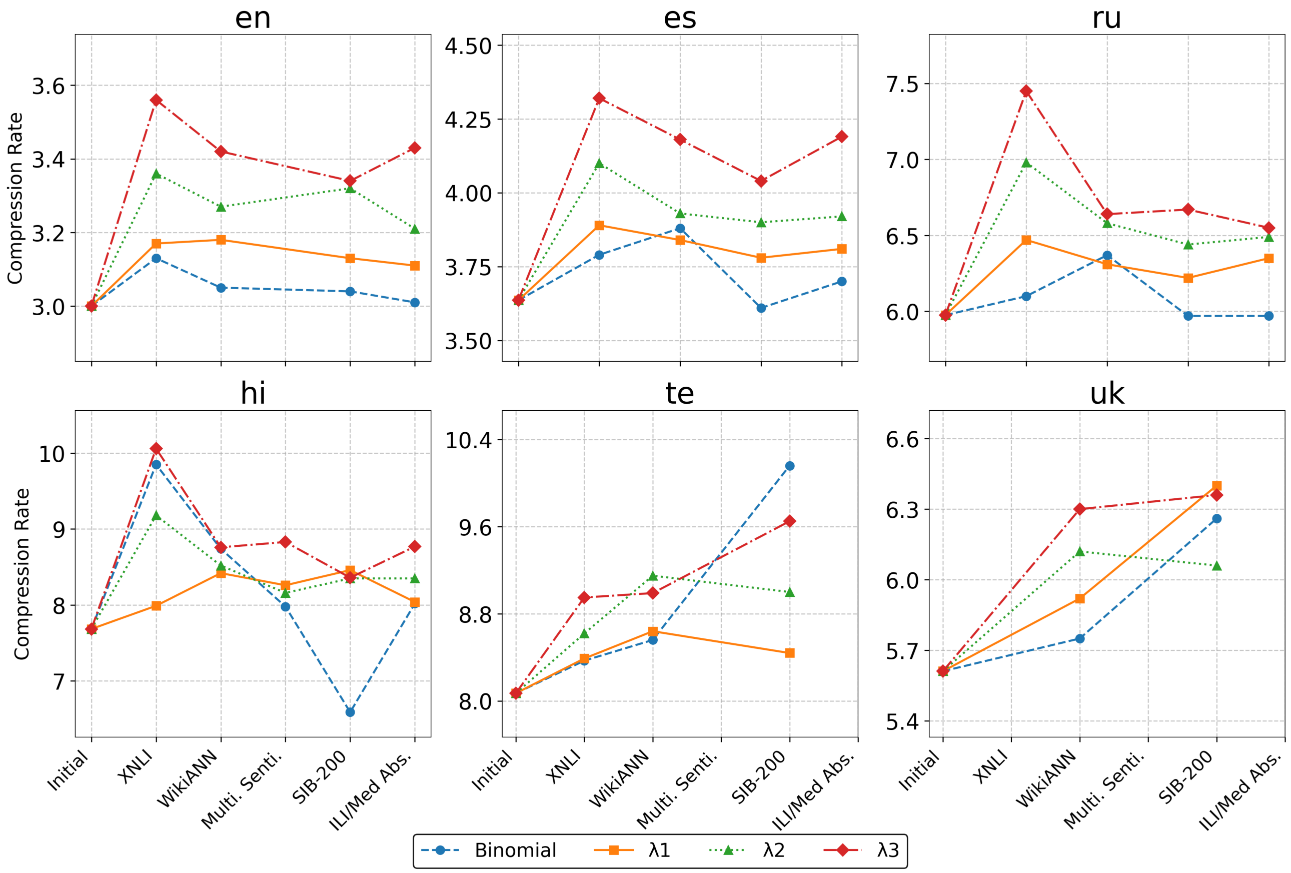

Figure 4: Compression rate changes with FlexiTokens across multiple tasks. *Initial* is the base compression rate before pretraining. Compression rate for binomial remains relatively low while while we also see a spike for task like XNLI

Qualitative analysis reveals consistent tokenization patterns across topic classification tasks like SIB-200 and Medical Abstracts, where compression remains stable across examples. In contrast, tasks such as XNLI exhibit compression spikes across all languages, indicating that some tasks benefit from more compression than others. In the Irony Classification task, FlexiTokens effectively tokenizes emojis with higher compression, preserving their semantic meaning. Following adaptation to the medical domain (Figure 1), we also find that medical terms are tokenized in unison as whole words, reducing fragmentation and better aligning with expected domain-specific vocabulary.

Adaptive tokenization to unseen scripts boosts performance without overfragmentation In Table 3, we extend our evaluation to Urdu, a low-resource Indo-Aryan language that shares linguistic commonalities with Hindi but uses a different script, not included in our pretraining dataset. We see that FlexiTokens outperforms BPE with more than 3 points after finetuning. Qualitative evaluation on the XNLI inputs (Table 6) reveals that our approach finds more compressed and semantically meaningful tokens compared to baselines (numbers and words). BPE tokenizer tokenizes Urdu with more tokens 6× than FlexiTokens which is follows the same pattern results from Figure 3. Note that FlexiTokens adapts well to unseen scripts because we use a script-agnostic boundary predictor as opposed to Ahia et al. (2024) which introduced the idea of equitable tokenization but requires a different boundary predictor for every language or script included during pretraining. Also, compound or rare words (especially medical terms or foreign-origin words like “hypertrophic“) are split into meaningful subwords.

Tradeoff between compression and model performance: We explore various configurations of α and how it impacts performance and show average results across all tasks in Table 5 (see Appendix F for a breakdown of performance on each language). As we scale the compression rate from 3× to 5 and 10, we observe slight decline in performance indicating that too much compression may result in loss of information hurting the model. We speculate that this issue

Table 5: Ablation for α: Average Accuracy and Compression Results Across Multiple Languages

Model	SIB-200	WikiANN	Multi. Senti.	XNLI	ILI	Med. Abs.	Avg
			Accuracy				
FlexiTokens 10x	53.76	64.35	**72.99**	65.23	89.07	62.95	68.06
FlexiTokens 5x	71.16	64.92	72.54	65.48	89.28	**63.47**	71.14
FlexiTokens 3x	**72.55**	**66.02**	72.74	**66.25**	**90.33**	62.74	**71.77**
			Compression Rate ± Std				
FlexiTokens 10x	28.89 ± 11.06	28.01 ± 14.14	27.41 ± 12.12	29.06 ± 8.55	38.80 ± 38.80	13.22 ± 2.15	27.56 ± 14.47
FlexiTokens 5x	10.72 ± 1.54	11.17 ± 3.69	11.25 ± 2.86	12.15 ± 1.76	14.82 ± 14.82	5.63 ± 0.33	10.96 ± 4.17
FlexiTokens 3x	6.19 ± 0.53	6.26 ± 1.33	6.17 ± 1.03	6.83 ± 0.60	8.35 ± 8.35	3.21 ± 0.15	6.17 ± 2.00

Table 6: Tokenization outputs with different methods (Urdu, Telugu, English)

<table>
<tr><th>Tokenizer</th><th>Sentence and Segmentation</th><th>#Tokens</th></tr>
<tr><td><b>ur</b></td><td>39-year-old SpongeBob was diagnosed with hypertrophic cardiomyopathy in Mumbai.</td><td>–</td></tr>
<tr><td>BPE</td><td>39|Ø|³|Ø§|ÙĦ|Û|ģ| |Ø§|Ø|³|Ù¾|ÙĪ|Ø|¬| Ø|¨|Ø§|Ø|¨| |Ú©|ÙĪ| Ùħ|Ùħ|Ø|¨|Ø|¦|Û|Ĭ| Ùħ|Û|Ĭ|Úº| |Û|ģ|Ø§|Ø|¦|Ù¾|Ø±|Ù¹|Ø±|Ø§|Ù|ģ|Ú©| |Ú©|Ø§|Ø±|ÚĪ|Û|Ĭ|ÙĪ|Ùħ|Û|Ĭ|ÙĪ|Ù¾|Û|Ĭ|Ø|ª|Ú¾|Û|Ĭ| |Ú©|Û|Ĭ| Ø|ª|Ø|´|Ø|®|Û|Ĭ|Ø|µ| |Û|ģ|ÙĪ|Ø|¦|Û|Ĭ|Û|Ķ</td><td>107</td></tr>
<tr><td>Binomial 3×</td><td>۳۹ | سالہ | اسپنج | باب | کو | ممبئی | میں | ھائپرٹر | افک | کارڈیو | میوپیتھی | کی | تشخیص | ھوئی۔</td><td>21</td></tr>
<tr><td>FlexiTokens 3×</td><td>۳۹ | سالہ | اسپنج | باب | کو | ممبئی | میں | ھائپرٹر افک | کارڈیو میوپیتھی | کی تشخیص | ھوئی۔</td><td>17</td></tr>
<tr><td><b>te</b></td><td>He spent the whole night watching Netflix. He fell asleep early.</td><td>–</td></tr>
<tr><td>BPE</td><td>a|ṭa|ḍu| rā|tra|nta| n|eṭ|phi|li|ks| cūstā| gaḍi|pāḍu. aṭaḍu| tva|raga| n|idra|pōyāḍu.</td><td>37</td></tr>
<tr><td>Binomial 3×</td><td>అతడు|రాత్రంతా|నెట్‌ఫ్లిక్స్|చూస్తూ|గడిపాడు.|అతడు|త్వరగా|నిద్రపోయాడు.</td><td>22</td></tr>
<tr><td>FlexiTokens 3×</td><td>అతడు|రాత్రంతా|నెట్‌ఫ్లిక్స్|చూస్తూ|గడిపాడు.|అతడు|త్వరగా|నిద్రపోయాడు.</td><td>17</td></tr>
<tr><td><b>en</b></td><td>Influenza and pneumonia were identified as major causes of mortality in children.</td><td>–</td></tr>
<tr><td>BPE</td><td>In|flu|enza| and| pneu|monia| were| identified| as| major| causes| of| mort|ality| in| children.</td><td>20</td></tr>
<tr><td>Binomial 3×</td><td>In|fl|uenza |an|d |pn|eumon|ia |wer|e |id|ent|ified |as |maj|or| |causes |of |mor|t|ality |in| |chil|dren.</td><td>25</td></tr>
<tr><td>FlexiTokens 3×</td><td>Infl|uenz|a |and| |pneu|m|onia |were |identified |as |m|ajor| |causes |of |m|or|tality |in |childr|en.</td><td>20</td></tr>
</table>

might be because of scale. Recent work has argued that larger models can handle larger vocabularies better (Tao et al., 2024). Its analogue in our case would be to train a larger model with more layers in the tokenization module. Due to computational constraints, we leave that exploration to future work.

5. Related Work

Tokenizer free language modeling Several works have explored the possibilities of training language models without relying on subword tokenization, instead representing text directly as a sequence of bytes (Xue et al., 2022; Al-Rfou et al., 2018; Wang et al., 2024; Limisiewicz et al.,

2024) or pixels (Lotz et al., 2023; Rust et al., 2023; Salesky et al., 2023). To address the efficiency challenges of processing raw characters or byte sequences on tokenizer free LMs, alternative architectures have proposed to either segment byte sequences into fixed-length (Nawrot et al., 2022b; Clark et al., 2022; Godey et al., 2022; Tay et al., 2022; YU et al., 2023) or dynamic segments (Nawrot et al., 2023; Ahia et al., 2024; Pagnoni et al., 2024). However, these models are pretrained with a fixed target compression rate, which limits their ability to adapt to shifts in data distribution.

Adapting tokenizers to new distributions There has been little research on adapting tokenizer-free LMs to new data distributions. Mofijul Islam et al. (2022) propose a character-based tokenizer by distilling segmentation information from heuristic-based subword tokenization. In contrast, several studies have explored adaptation strategies for subword tokenizers, both at inference time and during finetuning. For instance, prior work has shown that improved segmentation of large numbers can enhance performance on arithmetic tasks without retraining (Singh & Strouse, 2024; Sathe et al., 2025). In multilingual and domain-specific settings, various approaches have been proposed to adapt subword tokenizers during fine-tuning. These involve refining the tokenizer vocabulary with new tokens from the target distribution and initializing the corresponding embeddings to better capture linguistic and domain-specific characteristics (Park et al., 2021; Alabi et al., 2022; Minixhofer et al., 2022; Sachidananda et al., 2021; Liu et al., 2023). However, our experiments indicate that subword tokenizers often underperform in low-resource and non-Latin script languages due to over-segmentation.

6. Conclusion

We introduced FlexiTokens, a flexible, gradient-based tokenization approach that enables language models to adapt their segmentation patterns during finetuning. Unlike prior methods that enforce static or fixed compression rates, our method promotes dynamic tokenization aligned with the structure of the target distribution. Through multilingual and domain-diverse evaluations, FlexiTokens consistently reduces token over-fragmentation, improves downstream task performance, and achieves higher compression without sacrificing accuracy. Our results highlight the importance of adaptable tokenization strategies for building more efficient and generalizable language models.

References

Adelani, D. I., Liu, H., Shen, X., Vassilyev, N., Alabi, J. O., Mao, Y., Gao, H., and Lee, A. E.-S. Sib-200: A simple, inclusive, and big evaluation dataset for topic classification in 200+ languages and dialects. *arXiv preprint arXiv:2309.07445*, 2023.

Ahia, O., Kumar, S., Gonen, H., Kasai, J., Mortensen, D., Smith, N., and Tsvetkov, Y. Do all languages cost the same? tokenization in the era of commercial language models. In Bouamor, H., Pino, J., and Bali, K. (eds.), *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 9904–9923, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.614. URL https://aclanthology.org/2023.emnlp-main.614/.

Ahia, O., Kumar, S., Gonen, H., Hofmann, V., Limisiewicz, T., Tsvetkov, Y., and Smith, N. A. Magnet: Improving the multilingual fairness of language models with adaptive gradient-based tokenization. *Advances in Neural Information Processing Systems*, 37:47790–47814, 2024.

Al-Rfou, R., Choe, D., Constant, N., Guo, M., and Jones, L. Character-level language modeling with deeper self-attention. In *AAAI Conference on Artificial Intelligence*, 2018. URL https://api.semanticscholar.org/CorpusID:52004855.

Alabi, J. O., Adelani, D. I., Mosbach, M., and Klakow, D. Adapting pre-trained language models to African languages via multilingual adaptive fine-tuning. In Calzolari, N., Huang, C.-R., Kim, H., Pustejovsky, J., Wanner, L., Choi, K.-S., Ryu, P.-M., Chen, H.-H., Donatelli, L., Ji, H., Kurohashi, S., Paggio, P., Xue, N., Kim, S., Hahm, Y., He, Z., Lee, T. K., Santus, E., Bond, F., and Na, S.-H. (eds.), *Proceedings of the 29th International Conference on Computational Linguistics*, pp. 4336–4349, Gyeongju, Republic of Korea, October 2022. International Committee on Computational Linguistics. URL https://aclanthology.org/2022.coling-1.382/.

Ali, M., Fromm, M., Thellmann, K., Rutmann, R., Lübbering, M., Leveling, J., Klug, K., Ebert, J., Doll, N., Buschhoff, J., et al. Tokenizer choice for llm training: Negligible or crucial? In *Findings of the Association for Computational Linguistics: NAACL 2024*, pp. 3907–3924, 2024.

clapAI. Multilingualsentiment: A multilingual sentiment classification dataset, 2024. URL https://huggingface.co/datasets/clapAI/MultiLingualSentiment.

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

10.1162/tacl_a_00448. URL https://aclanthology.org/2022.tacl-1.5/.

Conneau, A., Lample, G., Rinott, R., Williams, A., Bowman, S. R., Schwenk, H., and Stoyanov, V. Xnli: Evaluating cross-lingual sentence representations. *arXiv preprint arXiv:1809.05053*, 2018.

Costa-Jussà, M. R., Cross, J., Çelebi, O., Elbayad, M., Heafield, K., Heffernan, K., Kalbassi, E., Lam, J., Licht, D., Maillard, J., et al. No language left behind: Scaling human-centered machine translation. *arXiv preprint arXiv:2207.04672*, 2022.

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

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)*, pp. 4171–4186, 2019.

Geiping, J., Stein, A., Shu, M., Saifullah, K., Wen, Y., and Goldstein, T. Coercing llms to do and reveal (almost) anything. *arXiv preprint arXiv:2402.14020*, 2024.

Godey, N., Castagné, R., de la Clergerie, É., and Sagot, B. MANTa: Efficient gradient-based tokenization for end-to-end robust language modeling. In Goldberg, Y., Kozareva, Z., and Zhang, Y. (eds.), *Findings of the Association for Computational Linguistics: EMNLP 2022*, pp. 2859–2870, Abu Dhabi, United Arab Emirates, December 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.findings-emnlp.207. URL https://aclanthology.org/2022.findings-emnlp.207/.

Graves, A. Generating sequences with recurrent neural networks. *arXiv preprint arXiv:1308.0850*, 2013.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Land, S. and Bartolo, M. Fishing for magikarp: Automatically detecting under-trained tokens in large language models. *arXiv preprint arXiv:2405.05417*, 2024.

Li, H., Koto, F., Wu, M., Aji, A. F., and Baldwin, T. Bactrian-x: Multilingual replicable instruction-following models with low-rank adaptation. *arXiv preprint arXiv:2305.15011*, 2023.

Limisiewicz, T., Blevins, T., Gonen, H., Ahia, O., and Zettlemoyer, L. MYTE: Morphology-driven byte encoding for better and fairer multilingual language modeling. In Ku, L.-W., Martins, A., and Srikumar, V. (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 15059–15076, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.804. URL https://aclanthology.org/2024.acl-long.804/.

Liu, S., Deng, N., Sabour, S., Jia, Y., Huang, M., and Mihalcea, R. Task-adaptive tokenization: Enhancing long-form text generation efficacy in mental health and beyond. In Bouamor, H., Pino, J., and Bali, K. (eds.), *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 15264–15281, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.944. URL https://aclanthology.org/2023.emnlp-main.944/.

Lotz, J., Salesky, E., Rust, P., and Elliott, D. Text rendering strategies for pixel language models. In Bouamor, H., Pino, J., and Bali, K. (eds.), *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 10155–10172, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.628. URL https://aclanthology.org/2023.emnlp-main.628/.

Minixhofer, B., Paischer, F., and Rekabsaz, N. WECHSEL: Effective initialization of subword embeddings for cross-lingual transfer of monolingual language models. In Carpuat, M., de Marneffe, M.-C., and Meza Ruiz, I. V. (eds.), *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 3992–4006, Seattle, United States, July 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.naacl-main.293. URL https://aclanthology.org/2022.naacl-main.293/.

Minixhofer, B., Ponti, E. M., and Vulić, I. Zero-shot tokenizer transfer. *arXiv preprint arXiv:2405.07883*, 2024.

Mofijul Islam, M., Aguilar, G., Ponnusamy, P., Solomon Mathialagan, C., Ma, C., and Guo, C. A vocabulary-free multilingual neural tokenizer for end-to-end task learning. In Gella, S., He, H., Majumder, B. P., Can, B., Giunchiglia, E., Cahyawijaya, S., Min, S., Mozes, M., Li, X. L., Augenstein, I., Rogers, A., Cho, K., Grefenstette, E., Rimell, L., and Dyer, C. (eds.), *Proceedings of the 7th Workshop on Representation Learning for NLP*, pp. 91–99, Dublin, Ireland, May 2022. Association for Computational Linguistics.

doi: 10.18653/v1/2022.repl4nlp-1.10. URL https://aclanthology.org/2022.repl4nlp-1.10/.

Nawrot, P., Tworkowski, S., Tyrolski, M., Kaiser, L., Wu, Y., Szegedy, C., and Michalewski, H. Hierarchical transformers are more efficient language models. In Carpuat, M., de Marneffe, M.-C., and Meza Ruiz, I. V. (eds.), *Findings of the Association for Computational Linguistics: NAACL 2022*, pp. 1559–1571, Seattle, United States, July 2022a. Association for Computational Linguistics. doi: 10.18653/v1/2022.findings-naacl.117. URL https://aclanthology.org/2022.findings-naacl.117/.

Nawrot, P., Tworkowski, S., Tyrolski, M., Kaiser, L., Wu, Y., Szegedy, C., and Michalewski, H. Hierarchical transformers are more efficient language models. In Carpuat, M., de Marneffe, M.-C., and Meza Ruiz, I. V. (eds.), *Findings of the Association for Computational Linguistics: NAACL 2022*, pp. 1559–1571, Seattle, United States, July 2022b. Association for Computational Linguistics. doi: 10.18653/v1/2022.findings-naacl.117. URL https://aclanthology.org/2022.findings-naacl.117/.

Nawrot, P., Chorowski, J., Lancucki, A., and Ponti, E. M. Efficient transformers with dynamic token pooling. In Rogers, A., Boyd-Graber, J., and Okazaki, N. (eds.), *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 6403–6417, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.acl-long.353. URL https://aclanthology.org/2023.acl-long.353/.

Pagnoni, A., Pasunuru, R., Rodriguez, P., Nguyen, J., Muller, B., Li, M., Zhou, C., Yu, L., Weston, J., Zettlemoyer, L., Ghosh, G., Lewis, M., Holtzman, A., and Iyer, S. Byte latent transformer: Patches scale better than tokens, 2024. URL https://arxiv.org/abs/2412.09871.

Pan, X., Zhang, B., May, J., Nothman, J., Knight, K., and Ji, H. Cross-lingual name tagging and linking for 282 languages. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1946–1958, Vancouver, Canada, July 2017. Association for Computational Linguistics. doi: 10.18653/v1/P17-1178. URL https://www.aclweb.org/anthology/P17-1178.

Park, C., Eo, S., Moon, H., and Lim, H. Should we find another model?: Improving neural machine translation performance with ONE-piece tokenization method without model modification. In Kim, Y.-b., Li, Y., and Rambow, O. (eds.), *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies: Industry Papers*, pp. 97–104, Online, June 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.naacl-industry.13. URL https://aclanthology.org/2021.naacl-industry.13/.

Penedo, G., Kydlíček, H., Lozhkov, A., Mitchell, M., Raffel, C. A., Von Werra, L., Wolf, T., et al. The fineweb datasets: Decanting the web for the finest text data at scale. *Advances in Neural Information Processing Systems*, 37:30811–30849, 2024a.

Penedo, G., Kydlíček, H., Sabolčec, V., Messmer, B., Foroutan, N., Jaggi, M., von Werra, L., and Wolf, T. Fineweb2: A sparkling update with 1000s of languages, December 2024b. URL https://huggingface.co/datasets/HuggingFaceFW/fineweb-2.

Petrov, A., La Malfa, E., Torr, P., and Bibi, A. Language model tokenizers introduce unfairness between languages. *Advances in neural information processing systems*, 36:36963–36990, 2023.

Rohanian, O., Taslimipoor, S., Evans, R., and Mitkov, R. Wlv at semeval-2018 task 3: Dissecting tweets in search of irony. In *Proceedings of The 12th International Workshop on Semantic Evaluation*, pp. 553–559, 2018.

Rust, P., Lotz, J. F., Bugliarello, E., Salesky, E., de Lhoneux, M., and Elliott, D. Language modelling with pixels. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=FkSp8VW8RjH.

Sachidananda, V., Kessler, J., and Lai, Y.-A. Efficient domain adaptation of language models via adaptive tokenization. In Moosavi, N. S., Gurevych, I., Fan, A., Wolf, T., Hou, Y., Marasović, A., and Ravi, S. (eds.), *Proceedings of the Second Workshop on Simple and Efficient Natural Language Processing*, pp. 155–165, Virtual, November 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.sustainlp-1.16. URL https://aclanthology.org/2021.sustainlp-1.16/.

Salesky, E., Verma, N., Koehn, P., and Post, M. Multilingual pixel representations for translation and effective cross-lingual transfer. In Bouamor, H., Pino, J., and Bali, K. (eds.), *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 13845–13861, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.854. URL https://aclanthology.org/2023.emnlp-main.854/.

Sathe, A., Aggarwal, D., and Sitaram, S. Improving consistency in LLM inference using probabilistic tokenization. In Chiruzzo, L., Ritter, A., and Wang, L. (eds.), *Findings of the Association for Computational Linguistics: NAACL 2025*, pp. 4766–4778, Albuquerque, New Mexico, April 2025. Association for Computational Linguistics. ISBN 979-8-89176-195-7. URL https://aclanthology.org/2025.findings-naacl.268/.

Schopf, T., Braun, D., and Matthes, F. Evaluating unsupervised text classification: zero-shot and similarity-based approaches. In *Proceedings of the 2022 6th International Conference on Natural Language Processing and Information Retrieval*, pp. 6–15, 2022.

Sennrich, R., Haddow, B., and Birch, A. Neural machine translation of rare words with subword units. In Erk, K. and Smith, N. A. (eds.), *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1715–1725, Berlin, Germany, August 2016. Association for Computational Linguistics. doi: 10.18653/v1/P16-1162. URL https://aclanthology.org/P16-1162/.

Singh, A. K. and Strouse, D. Tokenization counts: the impact of tokenization on arithmetic in frontier llms, 2024. URL https://arxiv.org/abs/2402.14903.

Tao, C., Liu, Q., Dou, L., Muennighoff, N., Wan, Z., Luo, P., Lin, M., and Wong, N. Scaling laws with vocabulary: Larger models deserve larger vocabularies, 2024. URL https://arxiv.org/abs/2407.13623.

Tay, Y., Tran, V. Q., Ruder, S., Gupta, J., Chung, H. W., Bahri, D., Qin, Z., Baumgartner, S., Yu, C., and Metzler, D. Charformer: Fast character transformers via gradient-based subword tokenization. *arXiv preprint arXiv:2106.12672*, 2021.

Tay, Y., Tran, V. Q., Ruder, S., Gupta, J., Chung, H. W., Bahri, D., Qin, Z., Baumgartner, S., Yu, C., and Metzler, D. Charformer: Fast character transformers via gradient-based subword tokenization. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=JtBRnrlOEFN.

Wang, J., Gangavarapu, T., Yan, J. N., and Rush, A. M. Mambabyte: Token-free selective state space model. In *First Conference on Language Modeling*, 2024. URL https://openreview.net/forum?id=X1xNsuKssb.

Xue, L., Barua, A., Constant, N., Al-Rfou, R., Narang, S., Kale, M., Roberts, A., and Raffel, C. ByT5: Towards a token-free future with pre-trained byte-to-byte models. *Transactions of the Association for Computational Linguistics*, 10:291–306, 2022. doi: 10.1162/tacl_a_00461. URL https://aclanthology.org/2022.tacl-1.17/.

YU, L., Simig, D., Flaherty, C., Aghajanyan, A., Zettlemoyer, L., and Lewis, M. MEGABYTE: Predicting million-byte sequences with multiscale transformers. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=JTmO2V9Xpz.

Zampieri, M., Nakov, P., Ljubešić, N., Tiedemann, J., Malmasi, S., and Ali, A. (eds.). *Proceedings of the Fifth Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial 2018)*, Santa Fe, New Mexico, USA, August 2018. Association for Computational Linguistics. URL https://aclanthology.org/W18-3900/.

Appendix

A. Limitations

Our limited computational budget prevents us from training larger models with more language on larger datasets. We anticipate the results will improve with scaling potentially providing even higher compression. We leave this exploration to future work. While we aimed for diversity of languages and scripts in our experiments, we acknowledge we do not cover a vast majority of linguistic diversity. But our methods are general and we believe our results should translate to more languages. We also acknowledge a tradeoff between the performance and compression rate of the languages with higher compression leading to slight decline in performance with some languages being more sensitive than others. FlexiTokens shares limitations of other segmentation methods in that it may not be suitable for languages where morphemes are discontinuous and vowels are interspersed between consonant roots for inflection or sometimes omitted such as Semitic languages or other languages with Templatic morphologies.

B. Broader Impacts Statement

Through this work, we demonstrate that tokenization can be performed in a non-rigid but adaptive manner that is more equitable, efficient, and performant across multiple domains. This flexibility opens new opportunities for incorporating low-resource and out-of-distribution (OOD) languages into state-of-the-art multilingual language models, particularly those being developed at industrial scale. FlexiTokens enables easier adaptation of models to new domains, even in data-scarce settings, creating pathways for easier and more targeted model adaptation. We also acknowledge a limitation in scaling the α, and we encourage the research community to further explore strategies for tuning this parameter that best suits their target domains and languages. We include our code in this submission and upon acceptance, we will release our code and training recipes to support reproducibility and foster adoption of FlexiTokens in future research.

C. Proof for optimizing the Binomial PMF

We begin by revisiting the boundary regularization term based on the Binomial distribution. Rather than minimizing the negative log-likelihood (NLL) of the Binomial, we simplify the form as follows:

$$\log P(k \mid N, \alpha) = k \log \alpha + (N-k) \log(1-\alpha) \tag{4}$$

Here, k is the number of predicted boundaries, N is the sequence length, and α is the boundary prior. Taking the derivative with respect to α:

$$\frac{d}{d\alpha} \log P(k \mid N, \alpha) = \frac{k}{\alpha} - \frac{N-k}{1-\alpha} \tag{5}$$

Setting this gradient to zero yields the maximum likelihood estimate (MLE):

$$\frac{k}{\alpha} = \frac{N-k}{1-\alpha} \quad \Rightarrow \quad k(1-\alpha) = (N-k)\alpha \quad \Rightarrow \quad \alpha = \frac{k}{N} \tag{6}$$

This shows that the optimal α aligns with the empirical boundary rate $\frac{k}{N}$. Therefore, instead of explicitly computing the Binomial loss, we may directly regularize the deviation between the predicted and expected boundary rates.

To encourage compression and avoid over-segmentation, we introduce a one-sided penalty:

$$\max\left(\frac{k}{N} - \alpha, 0\right) \tag{7}$$

This penalizes only when the boundary rate exceeds the prior α, allowing lower rates without penalty. However, to prevent trivial collapse (i.e., $\frac{k}{N} \to 0$), we relax this constraint by defining a soft upper bound:

$$\beta = \alpha - \lambda\sigma \tag{8}$$

where σ is the standard deviation of boundary rates over multiple samples and λ is a tunable margin. This leads to the final loss term:

$$\mathcal{L}_{\text{boundary}} = \max\left(\frac{k}{N} - \beta, 0\right) \tag{9}$$

This is the expression used in FlexiTokens from Equation 3. It replaces the rigid binomial constraint with a margin-aware compression regularizer that adapts across languages, scripts, and domains during training.

D. Hyperparameters

We extend our hyperparameter section (§3.2) and present the exact batch size used for finetuning all the models used in our experiments on a downstream task (see Table 2). In Figure 5, we also show a distribution of the training dataset size we used for each language in our experiment's training corpus. In addition to English, we keep the number of samples for all other languages the same to avoid any bias that could be caused by data imbalance in our models.

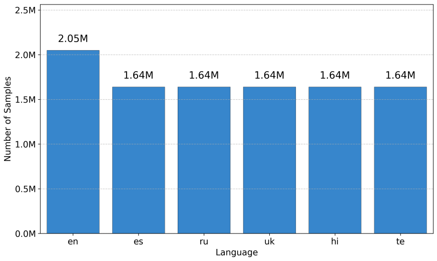

Figure 5: Number of training documents sampled by language

E. Results and Analyses

In this section, we present the full results discussed in §4 across all our selected downstream tasks as seen in Table 8, 9, 10, and 4. We also present the full results for our multilingual sentiment analysis evaluation (Table 7). All Results in this section contain values for performance metrics like accuracy and F1 score, compression rates and standard deviation of the compression rates.

F. Full Ablation Results

We present the full ablation results as discussed in §4 in Table 5. All results in this section (12, 13, 14, **??**, and 16) contain values for performance metrics like accuracy and F1 score, compression rates and standard deviation of the compression rates.

Table 7: Multilingual Sentiment Accuracy and Compression Results for 3x Configurations

Model	**es**	**ru**	**hi**	**Avg**
Accuracy				
BPE	–	–	–	–
Binomial 3x	**77.89**	87.20	**53.63**	**72.91**
FlexiTokens $\lambda 1$	77.75	**87.33**	53.42	72.83
FlexiTokens $\lambda 2$	77.77	**87.33**	53.12	72.74
FlexiTokens $\lambda 3$	77.63	87.13	53.01	72.59
Compression Rate ± Std				
Binomial	3.61 ± 0.48	5.97 ± 0.98	7.98 ± 1.90	5.85 ± 1.27
FlexiTokens $\lambda 1$	3.78 ± 0.27	6.22 ± 0.53	8.26 ± 1.82	6.09 ± 1.11
FlexiTokens $\lambda 2$	3.90 ± 0.28	6.44 ± 0.61	8.16 ± 1.65	6.17 ± 1.03
FlexiTokens $\lambda 3$	4.04 ± 0.37	6.67 ± 0.75	8.83 ± 1.84	6.51 ± 1.17

Table 8: WikiANN NER F1 Score and Compression Results for 3x Configurations

Model	**en**	**es**	**ru**	**uk**	**hi**	**te**	**Avg**
F1 Score							
BPE	52.30	67.7 0	64.94	74.99	60.23	48.18	61.39
Binomial	63.80	75.06	67.59	**78.06**	61.21	48.31	65.67
FlexiTokens $\lambda 1$	63.07	76.12	**68.30**	77.94	**62.26**	**51.74**	**66.57**
FlexiTokens $\lambda 2$	**63.96**	**76.23**	67.55	77.99	62.24	48.13	66.02
FlexiTokens $\lambda 3$	63.73	75.45	68.25	78.01	61.97	50.88	66.38
Compression Rate ± Std							
Binomial 3x	3.05 ± 0.47	3.88 ± 0.76	6.37 ± 1.67	5.75 ± 1.11	8.74 ± 3.27	8.56 ± 2.29	6.06 ± 1.86
FlexiTokens $\lambda 1$	3.18 ± 0.43	3.84 ± 0.54	6.31 ± 1.15	5.92 ± 0.90	8.42 ± 1.68	8.64 ± 1.55	6.05 ± 1.14
FlexiTokens $\lambda 2$	3.27 ± 0.44	3.93 ± 0.58	6.58 ± 1.38	6.12 ± 1.00	8.52 ± 1.49	9.15 ± 2.21	5.66 ± 1.33
FlexiTokens $\lambda 3$	3.42 ± 0.53	4.18 ± 0.66	6.64 ± 1.29	6.30 ± 1.07	8.76 ± 1.77	8.99 ± 2.07	6.38 ± 1.35

Table 9: SIB-200 Accuracy and Compression Results for with 3x Configurations

Model	**en**	**es**	**ru**	**uk**	**hi**	**te**	**Avg**
Accuracy							
BPE	**80.88**	**81.37**	**81.37**	**76.96**	60.78	**72.55**	**75.65**
Binomial	79.41	74.02	71.08	68.63	64.71	69.61	71.24
FlexiTokens $\lambda 1$	78.92	72.55	75.49	69.61	61.27	66.18	70.67
FlexiTokens $\lambda 2$	77.94	75.98	74.51	71.57	69.12	66.18	72.55
FlexiTokens $\lambda 3$	**80.88**	77.45	73.04	72.55	71.08	**71.08**	74.35
Compression Rate ± Std							
Binomial	3.04 ± 0.27	3.70 ± 0.34	5.97 ± 0.64	6.26 ± 0.70	6.59 ± 0.48	10.16 ± 1.34	5.95 ± 0.72
FlexiTokens $\lambda 1$	3.13 ± 0.25	3.81 ± 0.29	6.35 ± 0.64	6.40 ± 0.64	8.46 ± 0.82	8.44 ± 0.61	6.10 ± 0.58
FlexiTokens $\lambda 2$	3.32 ± 0.27	3.92 ± 0.31	6.49 ± 0.56	6.06 ± 0.54	8.35 ± 0.54	9.00 ± 0.79	6.19 ± 0.53
FlexiTokens $\lambda 3$	3.34 ± 0.35	4.19 ± 0.38	6.55 ± 0.75	6.36 ± 0.81	8.36 ± 0.59	9.65 ± 1.28	6.41 ± 0.76

Table 10: XNLI Accuracy and Compression Results for 3x Configurations

Model	**en**	**es**	**ru**	**hi**	**te**	**ur** (OOD)	**Avg**
				Accuracy			
BPE	73.09	69.9	65.95	61.48	68	54.11	65.42
Binomial	72.87	70.28	65.93	62.26	66.11	54.79	65.37
FlexiTokens λ1	**73.51**	70.22	66.47	**62.42**	67.11	56.99	66.12
FlexiTokens λ2	73.21	**70.84**	**66.97**	62.16	66.71	**57.58**	**66.25**
FlexiTokens λ3	73.35	70.22	66.75	62.36	**67.82**	57.33	66.31
				Compression Rate ± Std			
Binomial	3.13 ± 0.30	3.79 ± 0.48	6.10 ± 0.74	9.85 ± 1.28	8.37 ± 1.21	8.58 ± 0.82	6.64 ± 0.88
FlexiTokens λ1	3.17 ± 0.19	3.89 ± 0.26	6.47 ± 0.53	7.99 ± 0.75	8.39 ± 0.58	8.52 ± 0.71	6.40 ± 0.55
FlexiTokens λ2	3.36 ± 0.26	4.10 ± 0.30	6.98 ± 0.60	9.18 ± 0.85	8.62 ± 0.65	8.73 ± 0.73	6.83 ± 0.60
FlexiTokens λ3	3.56 ± 0.31	4.32 ± 0.34	7.45 ± 0.72	10.06 ± 1.17	8.95 ± 0.74	9.07 ± 0.80	7.24 ± 0.74

Table 11: ILI, Medical Abstracts, and Irony (for 3× Configuration)

Model	**ILI (hi)**	**Med. Abs. (en)**	**Irony (en)**
		Accuracy	
BPE	89.06	57.68	67.86
binomial	89.47	62.81	67.60
FlexiTokens λ1	89.58	62.92	68.37
FlexiTokens λ2	**90.33**	62.74	68.75
FlexiTokens λ3	89.55	**63.19**	**69.26**
		Compression Rate ± Std	
Binomial 3x	8.02 ± 1.38	3.01 ± 0.13	3.05 ± 0.14
FlexiTokens λ1	8.04 ± 0.89	3.11 ± 0.13	3.09 ± 0.08
FlexiTokens λ2	8.35 ± 0.87	3.21 ± 0.15	3.22 ± 0.31
FlexiTokens λ3	**8.77** ± 1.21	**3.43** ± 0.18	**3.36** ± 0.13

Table 12: SIB-200 α Ablation: Accuracy and Compression Results

Model	**en**	**es**	**ru**	**uk**	**hi**	**te**	**Avg**
				Accuracy			
FlexiTokens 10x	57.35	59.80	55.88	50.98	47.06	51.47	53.76
FlexiTokens 5x	**78.92**	**78.92**	74.51	**73.04**	62.75	58.82	71.16
FlexiTokens 3x	77.94	75.98	**74.51**	71.57	**69.12**	**66.18**	**72.55**
				Compression Rate ± Std			
FlexiTokens 10x	19.37 ± 8.23	16.23 ± 4.45	24.57 ± 6.82	28.69 ± 8.88	40.06 ± 14.68	44.43 ± 17.47	28.89 ± 11.06
FlexiTokens 5x	5.75 ± 0.65	6.78 ± 0.71	12.58 ± 1.91	10.62 ± 1.70	13.42 ± 1.63	15.17 ± 2.04	10.72 ± 1.54
FlexiTokens 3x	3.32 ± 0.27	3.92 ± 0.31	6.49 ± 0.56	6.06 ± 0.54	8.35 ± 0.54	9.00 ± 0.79	6.19 ± 0.53

Table 13: WikiANN α Ablation: F1 Score and Compression Results

Model	en	es	ru	uk	hi	te	Avg
F1 Score							
FlexiTokens 10x	61.81	75.48	66.90	76.90	59.88	45.15	64.35
FlexiTokens 5x	62.84	75.81	67.48	77.68	60.02	45.66	64.92
FlexiTokens 3x	**63.96**	**76.23**	**67.55**	**77.99**	**62.24**	**48.13**	**66.02**
Compression Rate ± Std							
FlexiTokens 10x	14.15 ± 6.07	16.87 ± 6.39	40.03 ± 19.10	27.91 ± 11.95	42.52 ± 21.82	26.55 ± 11.73	28.01 ± 14.14
FlexiTokens 5x	5.83 ± 1.23	7.26 ± 2.01	15.30 ± 5.90	11.93 ± 3.59	15.92 ± 4.68	10.80 ± 2.57	11.17 ± 3.69
FlexiTokens 3x	3.27 ± 0.44	3.93 ± 0.58	8.52 ± 1.49	6.58 ± 1.38	9.15 ± 2.21	6.12 ± 1.00	6.26 ± 1.33

Table 14: XNLI α Ablation: Accuracy and Compression Results

Model	en	es	ru	hi	te	ur	Avg
Accuracy							
FlexiTokens 10x	71.42	68.60	65.59	62.22	66.05	57.52	65.23
FlexiTokens 5x	72.97	70.38	65.47	61.88	65.49	56.71	65.48
FlexiTokens 3x	**73.21**	**70.84**	**66.97**	**62.16**	**66.71**	**57.58**	**66.25**
Compression Rate ± Std							
FlexiTokens 10x	13.41 ± 2.88	15.88 ± 3.12	25.20 ± 6.07	41.81 ± 12.06	37.23 ± 8.77	40.84 ± 12.71	29.06 ± 8.55
FlexiTokens 5x	6.06 ± 0.72	7.59 ± 0.88	13.02 ± 2.08	15.44 ± 2.16	15.10 ± 1.60	15.67 ± 2.40	12.15 ± 1.76
FlexiTokens 3x	3.36 ± 0.26	4.10 ± 0.30	6.98 ± 0.60	9.18 ± 0.85	8.62 ± 0.65	8.73 ± 0.73	6.83 ± 0.60

Table 16: ILI (hi) and Medical Abstract (en) λ Ablation: Accuracy and Compression Results

Model	ILI (hi)	Med. Abstract (en)
Accuracy		
FlexiTokens 10x	89.07	62.95
FlexiTokens 5x	89.28	**63.47**
FlexiTokens 3x	**90.33**	62.74
Compression Rate ± Std		
FlexiTokens 10x	38.80 ± 16.75	13.22 ± 2.15
FlexiTokens 5x	14.82 ± 3.00	5.63 ± 0.33
FlexiTokens 3x	8.35 ± 0.87	3.21 ± 0.15