TREX: Tokenizer Regression for Optimal Data Mixture

Inho Won[1*] **Hangyeol Yoo**[2*]
Minkyung Cho[1] **Jungyeul Park**[1,3] **Hoyun Song**[4†] **KyungTae Lim**[1,4†]
[1]KAIST CT [2]Seoul National University of Science and Technology
[3]Upstage AI [4]KAIST InnoCORE PRISM-AI Center
inho.won@kaist.ac.kr, hgyoo@seoultech.ac.kr,
{minkyung.cho, jungyeul, hysong, ktlim}@kaist.ac.kr

Abstract

Building effective tokenizers for multilingual Large Language Models (LLMs) requires careful control over language-specific data mixtures. While a tokenizer's compression performance critically affects the efficiency of LLM training and inference, existing approaches rely on heuristics or costly large-scale searches to determine optimal language ratios. We introduce **T**okenizer **Re**gression for Optimal Data Mi**X**ture (TREX), a regression-based framework that efficiently predicts the optimal data mixture for tokenizer training. TREX trains small-scale proxy tokenizers on random mixtures, gathers their compression statistics, and learns to predict compression performance from data mixtures. This learned model enables scalable mixture search before large-scale tokenizer training, mitigating the accuracy-cost trade-off in multilingual tokenizer design. Tokenizers trained with TReX's predicted mixtures outperform mixtures based on LLaMA3 and uniform distributions by up to 12% in both in- and out-of-distribution compression efficiency, demonstrating strong scalability, robustness, and practical effectiveness. All experiments are reproducible using the code available at the GitHub repository[1].

1 Introduction

A tokenizer converts raw text into a sequence of tokens, and its performance is often evaluated by its compression capability (Seo et al., 2025; Goldman et al., 2024). This ability to represent the same sentence with fewer tokens is critical for the efficient training and inference of a model (Scao et al., 2022; Stollenwerk, 2023; Ahia et al., 2023).

Achieving optimal tokenizer compression is challenging due to two main constraints. First, the saturation of a tokenizer's performance beyond a certain corpus size makes indiscriminate data scaling ineffective (Goldman et al., 2024; Reddy et al., 2025; Zuo et al., 2025). Second, the presence of significant data imbalance, especially in multilingual settings with underrepresented languages, leads to inconsistent performance across languages and domains (Dagan et al., 2024; Abagyan et al., 2025).

Data mixture optimization has emerged as a promising solution to these challenges, and its importance is particularly pronounced in multilingual settings. In this context, multilingual data distribution is as critical as vocabulary size and total corpus volume (Thakur et al., 2025; Petrov et al., 2023; Ahia et al., 2023), as language ratios directly influence subword segmentation and, consequently, compression efficiency (Wang et al., 2021; Pundalik et al., 2025). Despite its importance, most prior work has relied on empirical or manually tuned mixtures (Zhang et al., 2022), often derived from heuristic methods or costly searches, leaving the interplay between language ratios and compression largely unexplored. Therefore, this work addresses the question: *how can we design an optimal multilingual data mixture that maximizes tokenizer compression efficiency while remaining scalable to large LLM training?*

To this end, we introduce **T**okenizer **Re**gression for Optimal Data Mi**X**ture (TREX), a regression-based framework that efficiently predicts the optimal data mixture for tokenizer training. Instead of using heuristics or expensive large-scale searches, TREX trains a small set of lightweight proxy tokenizers on randomly sampled data mixtures, collects their compression statistics, and fits a regression model to predict compression performance from the data mixture. This learned model enables fast and reliable exploration of the vast mixture space before committing to large-scale tokenizer training. Our study investigates the following research questions:

- **RQ1.** Can TREX effectively approximate an

[*] Equal Contribution
[†] Corresponding Author
[1] https://github.com/HanGyeol-Yoo/TReX

Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics
Volume 1: Long Papers, pages 6353–6370

optimal multilingual data mixture for tokenizer training?

- **RQ2.** Is the relationship between data mixture and compression consistent across different data and vocabulary scales?
- **RQ3.** Can TREX maintain robust compression performance under diverse linguistic and domain-specific settings?

To answer these questions, we (1) propose a regression-based framework for predicting optimal data mixtures for tokenizer training, (2) evaluate tokenizers trained with the predicted mixtures against those based on GPT-4o (Hurst et al., 2024), LLaMA3 (Grattafiori et al., 2024), and uniform distributions, and (3) assess the scalability and generalization of TREX across multilingual and domain-specific datasets. Experiments show that the proposed regression model achieves a mean absolute percentage error of 1.989 and a rank correlation above 0.97, validating its predictive reliability. Tokenizers trained with the predicted optimal mixtures outperform baselines by up to 12% in both in- and out-of-distribution compression efficiency, demonstrating the scalability and practical effectiveness of TREX. The key contributions of this paper are as follows:

- We propose TREX, a regression-based framework that efficiently searches for the optimal data mixture for tokenizer training.
- We conduct a detailed empirical comparison against various data mixture strategies.
- We demonstrate the scalability and generalization of TREX across multilingual and domain-specialized settings.

2 Related Work

Tokenizer and compression A tokenizer's key performance metric is compression, which measures how efficiently a given text can be represented by counting how many tokens are needed to encode it (Scao et al., 2022; Stollenwerk, 2023; Ali et al., 2024; Dagan et al., 2024).

In practice, improving compression is non-trivial, as it is shaped by complex interactions among corpus size, vocabulary size, and data distribution (Whittington et al., 2025; Kim et al., 2025). Prior work has shown that merely increasing the size of the training corpus causes compression performance to saturate beyond a certain point (Reddy et al., 2025; Goldman et al., 2024; Zuo et al., 2025). Similarly, expanding the vocabulary size excessively fails to produce proportional gains in compression efficiency (Gowda and May, 2020; Liu et al., 2025a). Furthermore, when the training data are unevenly distributed, tokenizers tend to overfit to dominant languages or domains, achieving strong performance in some areas but suffering severe degradation in others (Dagan et al., 2024; Abagyan et al., 2025; Petrov et al., 2023). These findings highlight that compression efficiency cannot be improved indefinitely through scale alone. Given fixed corpus and vocabulary sizes, the proportions of languages and domains in the data mixture become the key determinant of tokenizer performance.

Data Mixture Optimization The composition of training data plays a decisive role in model performance, motivating research on balancing data mixtures. Early approaches relied on heuristic or empirical strategies—assigning mixture weights by linguistic families or corpus size (Thakur et al., 2025; Hayase et al., 2024; Karthika et al., 2025) or iteratively refining mixtures through repeated large-scale training (Thakur et al., 2025; Zhang et al., 2022). Despite their practicality, these methods are either manually crafted or prohibitively expensive, lacking a systematic means to predict optimal mixtures. Recent model-based studies such as DoReMi (Xie et al., 2024), DoGE (Fan et al., 2023), and RegMix (Liu et al., 2025b) address this limitation by estimating domain-wise loss via regression or interpolation. These approaches efficiently infer optimal mixture weights and improve pre-training or SFT performance (Li et al., 2025; Ye et al., 2024), but remain tied to loss minimization objectives, limiting their applicability to tokenizer training, where compression provides the most direct and scalable measure of representational efficiency. Building on model-based paradigms such as RegMix, our work extends this regression framework to the tokenizer level, predicting compression outcomes from data mixtures to identify optimal multilingual configurations without exhaustive retraining.

3 Preliminaries

Data Mixture We define the overall corpus as $D = \{D_{\text{train}}, D_{\text{test}}\}$, where D_{train} and D_{test} denote

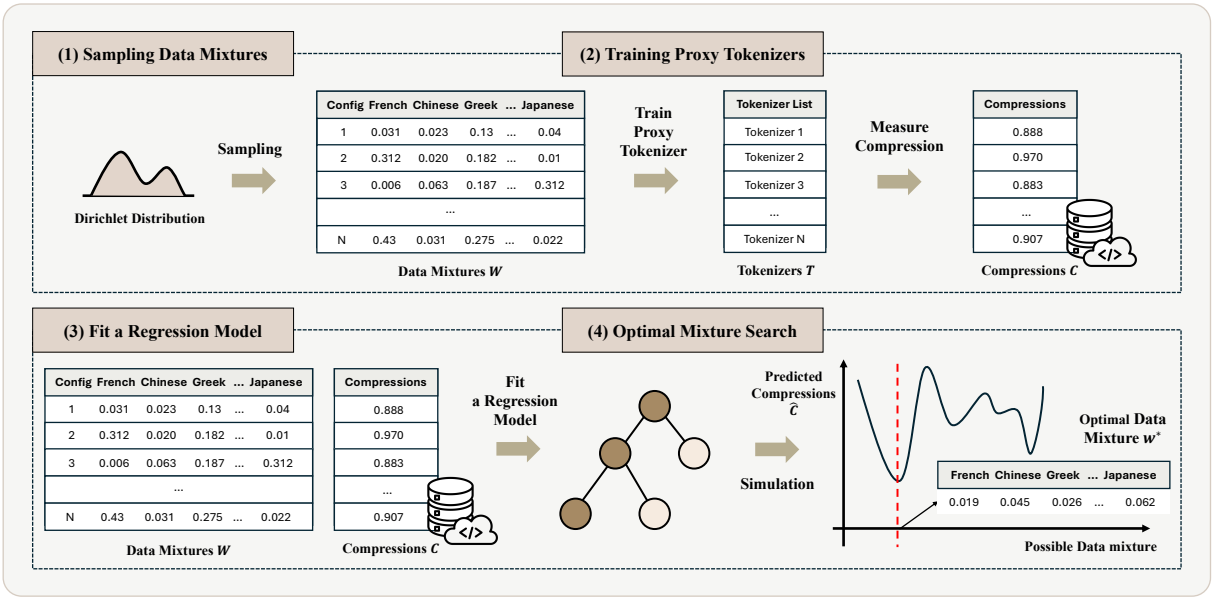

Figure 1: TREX overview. The process consists of four stages: (1) N data mixtures are sampled, (2) proxy tokenizers are trained for each mixture to measure C, (3) a regression model is fitted using $\mathbf{w}$ as input and C as the target, and (4) the model predicts C for candidate mixtures to identify the optimum.

the training and test corpora, respectively. Each tokenizer is trained under a configuration characterized by two parameters: the total training corpus size S and the vocabulary size V.

The training corpus D_{train} consists of k language-specific corpora, $D_{\text{train}} = \{d_1, d_2, \ldots, d_k\}$. A *data mixture* $\mathbf{w}$ specifies the relative contribution of each language corpus:

$$\mathbf{w} = (w_1, \ldots, w_k), \ \sum_{i=1}^{k} w_i = 1, \ w_i \geq 0 \quad (1)$$

The set of all valid mixtures defines the *mixture space* $\mathcal{W}$:

$$\mathcal{W} = \Big\{ \mathbf{w} \mid \sum_{i=1}^{k} w_i = 1, \ w_i \geq 0 \text{ for all } i \Big\}. \quad (2)$$

By fixing (S, V) and varying $\mathbf{w} \in \mathcal{W}$, we can train a family of tokenizers, denoted $T_{\mathbf{w}}$, each reflecting a different multilingual data mixture.

Tokenizer Compression We evaluate tokenizer performance using the *Normalized Sequence Length (NSL)* (Dagan et al., 2024), which measures the compression C_{tar} of a target tokenizer T_{tar} relative to a reference tokenizer T_{ref} on a test corpus D_{test}:

$$C_{\text{tar}} = \frac{\sum_{i=1}^{N} \text{Len}(T_{\text{tar}}(D^i_{\text{test}}))}{\sum_{i=1}^{N} \text{Len}(T_{\text{ref}}(D^i_{\text{test}}))}. \quad (3)$$

Here, $T(\cdot)$ denotes the tokenization function and $\text{Len}(\cdot)$ the number of tokens in the i-th sample. A smaller C_{tar} indicates better compression; values below 1.0 imply that the target tokenizer achieves more compact representations than the reference.

Problem Statement Our objective is to find the optimal data mixture $\mathbf{w}^*$ that minimizes the compression score for a given corpus size S and vocabulary size V:

$$C_{\mathbf{w}}(S, V) = \frac{\sum_{i=1}^{N} \text{Len}(T_{\mathbf{w}}(D^i_{\text{test}}))}{\sum_{i=1}^{N} \text{Len}(T_{\text{ref}}(D^i_{\text{test}}))}. \quad (4)$$

$$\mathbf{w}^* = \arg\min_{\mathbf{w} \in \mathcal{W}} C_{\mathbf{w}}(S, V). \quad (5)$$

Here, $T_{\mathbf{w}}$ denotes the tokenizer trained with mixture $\mathbf{w}$. This formulation captures how the tokenizer's compression varies with the data mixture and formalizes the goal of discovering the optimal mixture $\mathbf{w}^*$.

4 Method

We propose TREX (**T**okenizer **Re**gression for e**X**ploration), a framework that predicts the optimal data mixture for large-scale tokenizer training by leveraging numerous small-scale proxy tokenizers. TREX consists of four steps as detailed in Figure 1.

4.1 Sampling Data Mixtures

First, we sample N data mixtures, $\mathbf{W} = \{\mathbf{w}_1, \cdots, \mathbf{w}_n\}$, from the mixture space $\mathcal{W}$ using

a Dirichlet distribution based on the data size of each language. The Dirichlet distribution is ideal for this task as it generates mixture ratios on a probability simplex (i.e., vectors whose elements sum to 1) while reflecting the actual data distribution across languages (Lin, 2016; Tsun, 2020). This ensures a diverse yet feasible set of configurations for training the regression model.

4.2 Training Proxy Tokenizers

Each sampled mixture $\mathbf{w_i} \in \mathbf{W}$ is used to train a corresponding proxy tokenizer $T_{\mathbf{w_i}}$ in a small-scale setting (S_s, V_s), where the subscript s denotes small-scale configurations with reduced corpus size S_s and vocabulary size V_s. We then use each proxy tokenizer to measure its compression score $C_{\mathbf{w_i}}$ on a target test corpus D_{test}. The resulting set $\mathbf{C} = \{C_{\mathbf{w_1}}, \cdots, C_{\mathbf{w_n}}\}$, forms the target values for training the regression model.

4.3 Fitting a Regression Model

We fit a regression model f on the collected training set $\{(\mathbf{w_i}, C_{\mathbf{w_i}})\}_{\mathbf{i}=1}^{N}$, where each $\mathbf{w_i}$ denotes a sampled data mixture and $C_{\mathbf{w}_i}$ is its measured compression score in the small-scale setting. The model learns the mapping

$$f : \mathbf{w} \mapsto C_{\mathbf{w}}, \tag{6}$$

which approximates the relationship between data mixture and compression performance. Once trained, f can predict the expected compression score for any new mixture $\mathbf{w}'$ without requiring additional tokenizer training, enabling efficient exploration of the mixture space.

4.4 Optimal Mixture Search

The trained model f enables an efficient, large-scale search across the entire mixture space. For example, it can estimate compression scores for over 50M data mixtures, and this process can be completed within a few seconds using minimal computation. We identify the optimal data mixture $\mathbf{w}^*$ by finding the minimum of the learned function:

$$\mathbf{w}^* = \arg\min_{\mathbf{w}} f(\mathbf{w}) \tag{7}$$

Finally, we use this optimal mixture $\mathbf{w}^*$ to train a single, large-scale tokenizer.

5 Experiments and Results

In this section, we evaluate the effectiveness of TREX by addressing the following three research questions introduced earlier: (RQ1) Can TREX effectively approximate an optimal multilingual data mixture for tokenizer training? (RQ2) Is the relationship between data mixture ratios and compression performance consistent across different corpus scales and vocabulary sizes? (RQ3) Is efficient and scalable in real-world multilingual environment?

5.1 Experimental Setup

The experiments were organized from three perspectives: (1) dataset configuration, (2) regression model design, and (3) evaluation methodology.

Datasets For multilingual tokenizer training, we used FineWeb2-HQ (Messmer et al., 2025), one of the most widely used large-scale multilingual corpora. It comprises 19 languages and provides a diverse, publicly available dataset suitable for studying data mixture effects. We held out 0.1% of the corpus as the In-Distribution (ID) test set to avoid data leakage, and used FLORES (Team et al., 2022) as the Out-of-Distribution (OOD) test set. A detailed description of the corpus composition is provided in Appendix A.

Regression Model Following prior work on data mixture optimization (Liu et al., 2025b), we employed LightGBM (Ke et al., 2017) for the regression task. LightGBM, a gradient boosting based ensemble of decision trees, efficiently captures nonlinear relationships between language mixture ratios and compression performance, making it well-suited for this experiment.

Training Details As described in Section 4 and illustrated in Figure 1, TREX was trained through four main stages. First, we sampled N=512 data mixtures $\mathbf{W} = \{\mathbf{w_1}, \mathbf{w_2}, \ldots, \mathbf{w_{512}}\}$ from a Dirichlet distribution. Second, for each sampled mixture $\mathbf{w_i}$, we trained a proxy tokenizer $T_{\mathbf{w_i}}$ and measured its corresponding compression $C_{\mathbf{w_i}}$, forming the set $\mathbf{C} = \{C_{\mathbf{w_1}}, \ldots, C_{\mathbf{w_{512}}}\}$. To train lightweight proxy tokenizers, we randomly extracted S=1 GB of data from the D_{train} and set the vocabulary size to V=64K. Each subset was automatically partitioned according to the language ratios specified by $\mathbf{w_i}$. This process yielded 512 pairs of language mixtures and their corresponding compression $(\mathbf{w_i}, C_{\mathbf{w_i}})$, of which 480 were used for training and 32 for evaluation. For large-scale experiments, following prior studies (Bi et al., 2024; Liu et al., 2025a), we increased the configuration to S=30 GB and V=200K. Compression

Test Setting		Correlation (ρ)	MAPE
Data Size	Vocabulary	↑	↓
1GB	64k	0.979	1.989

Table 1: Prediction performance of the regression model under (S = 1GB, V = 64k). The model accurately predicts actual compression for unseen data mixtures, achieving a high rank correlation.

was measured relative to the GPT-4o tokenizer, which serves as the reference tokenizer T_{ref}, as recent analyses have shown that it provides more balanced multilingual coverage than other widely used tokenizers, such as LLaMA3 (Hayase et al., 2024).

Evaluation metrics We evaluated the predictive performance of the regression model using the Mean Absolute Percentage Error (MAPE) and the Spearman rank correlation (ρ). MAPE quantifies the absolute deviation between the predicted and actual compression values, while ρ measures the consistency of their relative rankings. MAPE is used to validate RQ1, which evaluates the model's ability to approximate optimal compression of data mixture. Meanwhile, ρ is used to validate RQ2 by examining rank stability across different corpus and vocabulary scales (Liu et al., 2025b). Together, these metrics provide a comprehensive view of the predictive accuracy and rank consistency of TREX.

5.2 Regression Model Performance

Within-Scale Prediction Table 1 presents the performance of the TREX regression model under the S=1GB, V=64K setting. Across 32 test samples, TREX achieved a MAPE of 1.989, indicating that it predicts compression score with less than 2% average error. Furthermore, the model also achieved a Spearman rank correlation of ρ=0.979, showing that it preserves the relative ranking among data mixtures with nearly 98% consistency. These results show that TREX can accurately model the relationship between data mixtures and compression even in a small proxy environment, providing strong empirical support for RQ1.

Cross-Scale Generalization Previously, we showed that TREX accurately predicts the compression of various data mixtures under a fixed configuration (S=1GB, V=64K). However, in practical tokenizer design, both corpus size (S) and vocabulary size (V) can vary substantially. A key question, therefore, is whether the relative ranking

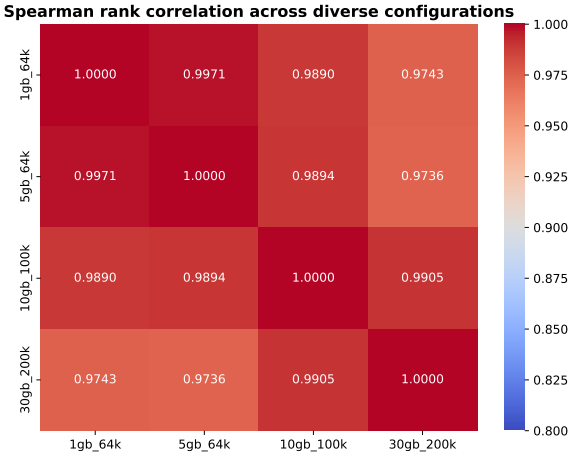

Figure 2: Spearman rank correlation heatmap across diverse configurations (S, V).

of mixtures in terms of compression remains consistent across different scales. If this stability holds, TREX can serve as a scalable approach, enabling small proxy tokenizers to predict performance in large-scale settings. This assumption corresponds to the principle of *Rank Invariance* proposed by Liu et al. (2025b).

Rank Invariance assumes that although absolute compression values may change as environmental variables (e.g., corpus size or vocabulary size) vary, the relative ordering of mixtures remains stable. To test this assumption, we trained tokenizers under multiple scale settings with varying corpus and vocabulary sizes (1GB–64K, 5GB–64K, 10GB–100K, 30GB–200K) and computed the Spearman rank correlation of mixture efficiencies across these configurations. As shown in Figure 2, correlation between all setting pairs were consistently high ($\rho \geq 0.96$). For example, the correlation between the smallest (1GB–64K) and largest (30GB–200K) configurations reached ρ=0.974, indicating that compression rankings remain nearly unchanged even when corpus size and vocabulary scale differ by several orders of magnitude. This finding demonstrates that mixture efficiency is largely invariant to training scale–i.e., Rank Invariance holds.

These findings confirm that TREX can reliably generalize language importance patterns observed in small proxy tokenizers to large-scale environments. By maintaining consistent mixture prediction performance independent of scale variations, TREX provides strong evidence for the "scale-invariant mixture consistency" proposed in RQ2. Further experiments supporting Rank Invariance are provided in Appendix C.

LANG	CHAR	$\mathbf{w}^{uni}$	$\mathbf{w}^{LB}$	$\mathbf{w}^{gpt}$	$\mathbf{w}^{llama}$	$\mathbf{w}^{\text{TREX}}$
In-Distribution (over FineWeb2-HQ) (↓)						
CMN	Hani	0.827	1.097	0.841	0.962	0.831
ELL	Grek	0.721	0.847	0.763	0.715	0.778
FAS	Arab	0.842	1.021	0.900	0.825	0.906
JPN	Jpan	0.682	0.997	0.748	0.621	0.663
RUS	Cyrl	0.982	1.121	0.868	0.901	0.878
DEU	Latn	0.971	0.932	0.928	0.953	0.933
FRA	Latn	0.995	0.962	0.939	0.991	0.933
ITA	Latn	0.886	0.868	0.912	0.912	0.862
POR	Latn	0.981	0.964	0.938	0.988	0.962
SPA	Latn	0.992	0.971	0.943	0.985	0.978
CES	Latn	0.746	0.715	0.838	0.713	0.813
DAN	Latn	0.827	0.812	0.872	0.896	0.859
HUN	Latn	0.693	0.667	1.204	1.204	0.918
IND	Latn	0.842	0.821	0.891	0.957	0.890
NLD	Latn	0.970	0.952	0.930	1.006	0.970
POL	Latn	0.758	0.743	0.809	0.762	0.778
SWE	Latn	0.841	0.827	0.906	0.915	0.864
TUR	Latn	0.782	0.765	0.815	0.737	0.894
VIE	Latn	0.894	0.890	0.905	0.893	0.896
All Languages		0.888	0.970	0.883	0.907	**0.871**

LANG	CHAR	$\mathbf{w}^{uni}$	$\mathbf{w}^{LB}$	$\mathbf{w}^{gpt}$	$\mathbf{w}^{llama}$	$\mathbf{w}^{\text{TREX}}$
Out-Of-Distribution (over FLORES-200) (↓)						
CMN	Hani	0.838	1.134	0.858	0.951	0.838
ELL	Grek	0.707	0.839	0.752	0.701	0.768
FAS	Arab	0.940	1.127	1.003	0.923	1.009
JPN	Jpan	0.672	1.004	0.746	0.608	0.652
RUS	Cyrl	0.993	1.141	0.870	0.905	0.879
DEU	Latn	0.981	0.937	0.933	0.960	0.937
FRA	Latn	1.007	0.972	0.946	1.002	0.939
ITA	Latn	0.883	0.864	0.915	0.913	0.859
POR	Latn	0.992	0.973	0.945	1.000	0.970
SPA	Latn	1.005	0.985	0.954	0.999	0.992
CES	Latn	0.752	0.720	0.852	0.719	0.824
DAN	Latn	0.871	0.856	0.919	0.944	0.902
HUN	Latn	0.696	0.669	1.223	1.225	0.929
IND	Latn	0.854	0.832	0.907	0.980	0.905
NLD	Latn	0.985	0.968	0.943	1.027	0.987
POL	Latn	0.757	0.743	0.812	0.763	0.779
SWE	Latn	0.858	0.843	0.930	0.938	0.884
TUR	Latn	0.794	0.779	0.827	0.748	0.908
VIE	Latn	0.916	0.912	0.926	0.916	0.916
All Languages		0.904	0.997	0.907	0.906	**0.877**

Table 2: A comparison of C (lower is better) for various data mixtures on the In-Distribution (FineWeb2-HQ) dataset and the Out-of-Distribution (FLORES) dataset. Each row represents a language (Lang) and its corresponding character system (Char). The proposed optimal mixture, $\mathbf{w}^{\text{TREX}}$, achieves the best performance across both settings.

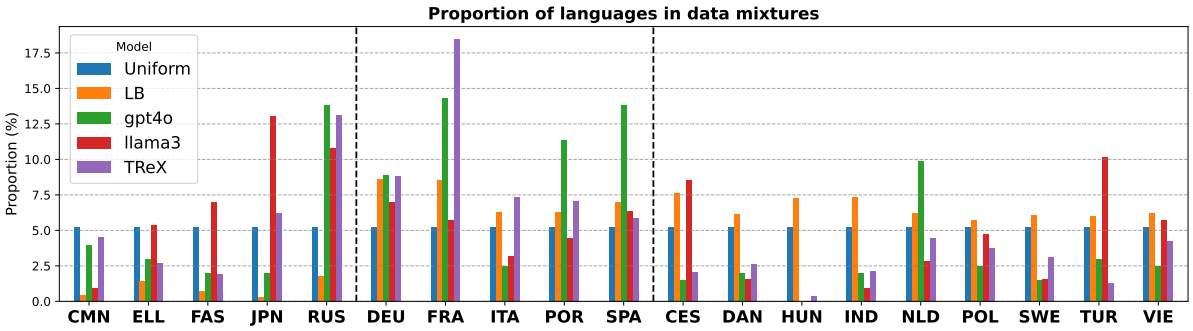

Figure 3: Proportions of languages in the data mixtures.

5.3 Evaluating TREX on Large-Scale Tokenizer Training

In this section, we compare how the optimal data mixture $\mathbf{w}^{\text{TREX}}$ derived from the TREX performs in large-scale tokenizer training (S=30GB, V=200K) relative to other baseline mixtures in terms of the compression performance C. To this end, we established the following four mixtures as baselines:

- $\mathbf{w}^{uni}$: A data mixture that assigns equal weights to all languages, i.e. $\mathbf{w}^{uni} = (\frac{1}{19}, \frac{1}{19}, \ldots, \frac{1}{19})$
- $\mathbf{w}^{LB}$: A data mixture reflecting the language proportions proposed by Abagyan et al.. It specifically applies a strategy where languages are grouped into language buckets based on family and script to determine the optimal mixture ratios for training.
- $\mathbf{w}^{gpt}$: A data mixture reflecting the proportions of 19 languages in the vocabulary of the GPT-4o tokenizer
- $\mathbf{w}^{llama}$: A data mixture reflecting the proportions of 19 languages in the vocabulary of the LLaMA3 tokenizer

We trained large-scale tokenizers using these four mixtures, and the exact language ratios for each model are presented in Appendix H. The average compression of each tokenizer was evaluated as a weighted mean, where the weights correspond to the language proportions in the test corpus D_{test} and each term represents the compression performance for that language.

Overall Table 2 compares the compression performance of the tokenizer trained with the optimal data mixture $\mathbf{w}^{\text{TREX}}$ derived from TREX's regression model against four baseline methods. In the In-Distribution setting, $\mathbf{w}^{\text{TREX}}$ achieved a score of 0.871, slightly outperforming $\mathbf{w}^{gpt}$, which recorded 0.883, and showing a 10.21% improvement over $\mathbf{w}^{LB}$. In the Out-of-Distribution setting,

$\mathbf{w}^{\text{TREX}}$ also achieved the best performance with a score of 0.877, surpassing all baseline methods in compression efficiency.

These results demonstrate that $\mathbf{w}^{\text{TREX}}$ yields a more efficient tokenizer in terms of compression compared to both previously proposed data mixture strategies ($\mathbf{w}^{uni}$, $\mathbf{w}^{LB}$) and widely used practical tokenizers ($\mathbf{w}^{gpt}$, $\mathbf{w}^{llama}$).

Out-of-Distribution Results and Generalization As shown in Table 2, the most significant strength of $\mathbf{w}^{\text{TREX}}$ lies in its robustness on Out-of-Distribution (OOD) data. Despite being optimized under the In-Distribution setting, $\mathbf{w}^{\text{TREX}}$ achieved the lowest average compression score of 0.877 on OOD corpus that deviate from the training distribution. This value is substantially lower than that of the second-best method, $\mathbf{w}^{uni}$ (0.904), and shows a remarkable performance gap compared to $\mathbf{w}^{LB}$ (0.997). These results suggest that the data mixture derived through $\mathbf{w}^{\text{TREX}}$ does not overfit to the specific training data but instead captures the underlying relationship between languages and their compression, enabling the design of mixtures that generalize well. Ultimately, tokenizers trained with $\mathbf{w}^{\text{TREX}}$ can deliver the most stable and efficient compression performance across the diverse and unpredictable data distributions encountered in real-world deployment scenarios.

Script-Level Analysis: Pronounced Gains in Non-Latin Languages Notably, when focusing on non-Latin character languages (Hani, Grek, Arab, Jpan, and Cyrl), the efficiency of TREX becomes even more pronounced. When comparing the average compression scores of these five languages, $\mathbf{w}^{\text{TREX}}$ achieved 0.814 in the Non-Latin group, lower than all baselines ($\mathbf{w}^{uni}$=0.848, $\mathbf{w}^{llama}$=0.863, etc.). These results represent an improvement of approximately 3.4 percentage points over the uniform distribution and 4.9 percentage points over the $\mathbf{w}^{llama}$, indicating that $\mathbf{w}^{\text{TREX}}$ consistently preserves compression efficiency even for non-Latin scripts. As shown in Figure 3, the $\mathbf{w}^{llama}$ allocates 36.8% of the total data to non-Latin character languages, whereas $\mathbf{w}^{\text{TREX}}$ assigns only 28.2%, which is about 8.6 percentage points lower. Nevertheless, the average compression score of the non-Latin group in $\mathbf{w}^{llama}$ (0.863) remains higher than that of $\mathbf{w}^{\text{TREX}}$ (0.814), further demonstrating TREX's superior efficiency.

This finding suggests that simply increasing the proportion of certain languages is insufficient; rather, efficient weighting that accounts for the segmentation structure and statistical redundancy of each script is the key. Consequently, $\mathbf{w}^{\text{TREX}}$ achieves higher efficiency with fewer non-Latin tokens, empirically demonstrating the importance of character-aware mixture optimization in multilingual tokenizer design.

Impact of Language Distribution on Compression Performance Figure 3 presents the language proportions of five related languages (from DEU to SPA) across different models, and Table 2 shows their corresponding compression efficiencies. When considered together, these results reveal a nonlinear relationship between the mixture entropy, representing the diversity of language ratios, and the average compression efficiency.

Mixtures with high entropy such as $\mathbf{w}^{uni}$ are generally stable, but they fail to sufficiently remove token redundancy within specific language families, resulting in a limited average compression score of 0.904. In contrast, $\mathbf{w}^{\text{TREX}}$ achieved the lowest average compression score (0.877) despite having a relatively lower entropy (a more biased distribution). This suggests that a mixture does not necessarily need to be uniform; rather, efficient bias can actually enhance tokenizer performance.

This observation contrasts with the results of $\mathbf{w}^{llama}$, whose distribution is closer to $\mathbf{w}^{uni}$, yet exhibits higher compression scores (lower efficiency) in major Romance languages such as Italian (ITA), Portuguese (POR), and Spanish (SPA). In other words, a biased mixture that accounts for structural redundancy among languages can yield a more efficient tokenizer than a simple uniform distribution.

As further shown in Appendix Figure 13 and Table 8, $\mathbf{w}^{\text{TREX}}$ achieves the highest efficiency at a moderate level of entropy, supporting the notion that efficient bias, rather than uniformity, leads to the optimal design of multilingual mixtures.

6 Analysis in Real-World Scenarios

In the previous section, we demonstrated that TREX achieves superior compression compared to existing tokenizers. In this section, we examine whether this advantage translates into consistent efficiency during large-scale language model (LLM) training. In addition, we investigate its robustness in domain-specialized environments, as formulated in RQ3.

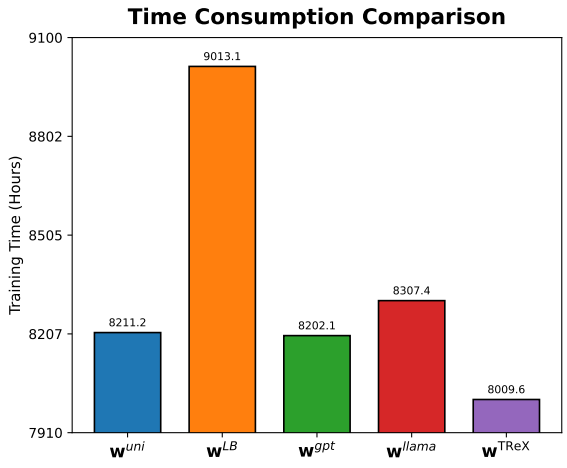

Figure 4: Model Training Time Consumption Comparison

Does TREX Improve Efficiency in LLM Training? To quantify the impact of tokenizer choice on LLM training costs, we follow the experimental setup of prior work (OLMo et al., 2024), and consider a 13-billion-parameter model trained on a 3-trillion-token corpus, with token counts computed under the GPT-4o tokenizer. Under this setting, a tokenizer with higher compression performance can encode the same raw corpus into fewer tokens, thereby proportionally reducing the total number of training FLOPs.

Figure 4 compares the estimated total training time for models trained with different tokenizer mixtures. The tokenizer trained with $\mathbf{w}^{\text{TREX}}$ achieves the shortest training time of 8,009.6 hours, outperforming all baselines. Relative to the least efficient mixture ($\mathbf{w}^{LB}$), TREX reduces total training time by more than 1,000 hours, and still saves roughly 200 hours compared to the next-best configuration ($\mathbf{w}^{gpt}$). These results highlight that the data mixture predicted by TREX not only improves compression efficiency but also yields substantial computational savings during large-scale language model training.

Cost Efficiency of TREX in Finding the Optimal Data Mixture. A potential concern is whether TREX's regression-based approach offers tokenizer training efficiency. Despite the initial cost of training 480 proxy tokenizers, TREX determines the optimal mixture in a single step, while AdaptMix (Thakur et al., 2025) requires about 20 large-scale iterations (S=30GB, V=200K) to converge. As shown in Appendix F, this leads to a 52.2% reduction in total training time, saving roughly 20 hours overall. These results demonstrate that TREX achieves significantly higher time efficiency with minimal computational overhead. Further implementation details are provided in Appendix F.

Test Setting		Correlation (ρ)	MAPE
Data Size	Vocabulary	↑	↓
1GB	64k	0.981	0.921

Table 3: Performance of the regression model in the medical domain.

Is TREX Effective in Domain-Specific Scenarios? In practice, it is important to train tokenizers that are effective in both multilingual and domain-specific environments (Dagan et al., 2024; Abagyan et al., 2025). We therefore examine whether the proposed TREX remains effective when the training is focused on a specific target domain. To this end, we used the `Pile` dataset with domain labels and trained the regression model to optimize compression performance on medical domain text. As shown in Table 3, the regression model of TREX achieved a Spearman rank correlation above 0.965 and a MAPE of 0.921. This indicates that TREX can accurately predict the compression behavior of data mixtures even within a specific domain. Additional supporting experiments and detailed analyses of domain-specific performance are provided in Appendix D.

7 Conclusion

TREX effectively addresses the fundamental challenge of identifying the optimal data mixture for tokenizer training. By leveraging a regression model trained on small-scale proxy tokenizers, it can accurately predict the compression performance of various data mixtures without the need for repeated full-scale training, thereby significantly reducing computational costs. The regression model of TREX demonstrated high reliability, achieving a MAPE of less than 2% and a Spearman rank correlation exceeding 0.97 when predicting tokenizer compression performance. Owing to this precise predictive capability, the tokenizer trained with the optimal mixture derived from TREX achieved up to a 12% improvement in compression efficiency compared to heuristic approaches and data mixture used in tokenizers such as LLaMA3 and GPT-4o. This improvement was consistently observed across both in-distribution and out-of-distribution data.

Limitations

TREX has a clear objective: to improve the compression performance of tokenizers. Better compression provides the practical benefit of reducing the total token count, thereby enhancing the training and inference speed of LLMs. However, compression performance is not always proportional to the resulting language model's downstream performance on tasks such as translation, summarization, or reasoning. This study does not examine how a tokenization method optimized solely for compression may affect a model's semantic understanding or complex reasoning capability. Investigating the relationship between compression efficiency and the overall model quality remains an important direction for future research. Furthermore, while our experimental results (from 1GB/64k to 30GB/200k) strongly support the underlying assumption of rank invariance, further validation is required. It is necessary to determine if this assumption holds at more extreme scales (e.g., training data exceeding several hundred gigabytes, vocabularies larger than 500,000) or with entirely different sets of languages. Finally, this research was conducted using a dataset composed of 19 languages. Although these languages span various families and writing systems, they do not represent the full linguistic diversity of the world. The relationship between data mixture and compression ratio could become more complex, particularly if a large number of morphologically rich languages (such as agglutinative or polysynthetic ones) or low-resource languages are included.

Acknowledgement

This research was supported by the Inno-CORE program of the Ministry of Science and ICT(N10250154) and the Top-Tier AI Global HRD invitation program (RS-2025-25461932) supervised by the IITP(Institute for Information & Communications Technology Planning & Evaluation)

References

Diana Abagyan, Alejandro R Salamanca, Andres Felipe Cruz-Salinas, Kris Cao, Hangyu Lin, Acyr Locatelli, Marzieh Fadaee, Ahmet Üstün, and Sara Hooker. 2025. One tokenizer to rule them all: Emergent language plasticity via multilingual tokenizers. *arXiv preprint arXiv:2506.10766*.

Orevaoghene Ahia, Sachin Kumar, Hila Gonen, Jungo Kasai, David R Mortensen, Noah A Smith, and Yulia Tsvetkov. 2023. Do all languages cost the same? tokenization in the era of commercial language models. *arXiv preprint arXiv:2305.13707*.

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

Xiao Bi, Deli Chen, Guanting Chen, Shanhuang Chen, Damai Dai, Chengqi Deng, Honghui Ding, Kai Dong, Qiushi Du, Zhe Fu, et al. 2024. Deepseek llm: Scaling open-source language models with longtermism. *arXiv preprint arXiv:2401.02954*.

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

Simin Fan, Matteo Pagliardini, and Martin Jaggi. 2023. Doge: Domain reweighting with generalization estimation. *arXiv preprint arXiv:2310.15393*.

Omer Goldman, Avi Caciularu, Matan Eyal, Kris Cao, Idan Szpektor, and Reut Tsarfaty. 2024. Unpacking tokenization: Evaluating text compression and its correlation with model performance. *arXiv preprint arXiv:2403.06265*.

Thamme Gowda and Jonathan May. 2020. Finding the optimal vocabulary size for neural machine translation. *arXiv preprint arXiv:2004.02334*.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Jonathan Hayase, Alisa Liu, Yejin Choi, Sewoong Oh, and Noah A Smith. 2024. Data mixture inference attack: Bpe tokenizers reveal training data compositions. *Advances in Neural Information Processing Systems*, 37:8956–8983.

Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, et al. 2024. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*.

N J Karthika, Maharaj Brahma, Rohit Saluja, Ganesh Ramakrishnan, and Maunendra Sankar Desarkar. 2025. Multilingual tokenization through the lens of indian languages: Challenges and insights.

Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. 2017. Lightgbm: A highly efficient gradient boosting decision tree. *Advances in neural information processing systems*, 30.

Minjun Kim, Hyeonseok Lim, Hangyeol Yoo, Inho Won, Seungwoo Song, Minkyung Cho, Junhun Yuk, Changsu Choi, Dongjae Shin, Huije Lee, Hoyun Song, Alice Oh, and Kyungtae Lim. 2025. Kormo: Korean open reasoning model for everyone.

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

Yuan Li, Zhengzhong Liu, and Eric P. Xing. 2025. Data mixing optimization for supervised fine-tuning of large language models. In *Forty-second International Conference on Machine Learning*.

Jiayu Lin. 2016. On the dirichlet distribution. *Department of Mathematics and Statistics, Queens University*, 40.

Alisa Liu, Jonathan Hayase, Valentin Hofmann, Sewoong Oh, Noah A. Smith, and Yejin Choi. 2025a. SuperBPE: Space travel for language models. In *Second Conference on Language Modeling*.

Qian Liu, Xiaosen Zheng, Niklas Muennighoff, Guangtao Zeng, Longxu Dou, Tianyu Pang, Jing Jiang, and Min Lin. 2025b. Regmix: Data mixture as regression for language model pre-training. In *International Conference on Learning Representations (ICLR)*.

Bettina Messmer, Vinko Sabolčec, and Martin Jaggi. 2025. Enhancing multilingual llm pretraining with model-based data selection. *arXiv*.

Team OLMo, Pete Walsh, Luca Soldaini, Dirk Groeneveld, Kyle Lo, Shane Arora, Akshita Bhagia, Yuling Gu, Shengyi Huang, Matt Jordan, et al. 2024. 2 olmo 2 furious. *arXiv preprint arXiv:2501.00656*.

Aleksandar Petrov, Emanuele La Malfa, Philip H. S. Torr, and Adel Bibi. 2023. Language model tokenizers introduce unfairness between languages. In *Advances in Neural Information Processing Systems 36 (NeurIPS 2023)*.

Kundeshwar Pundalik, Piyush Sawarkar, Nihar Sahoo, Abhishek Shinde, Prateek Chanda, Vedant Goswami, Ajay Nagpal, Atul Singh, Viraj Thakur, Vijay Dewane, Aamod Thakur, Bhargav Patel, Smita Gautam, Bhagwan Panditi, Shyam Pawar, Madhav Kotcha, Suraj Racha, Saral Sureka, Pankaj Singh, Rishi Bal, Rohit Saluja, and Ganesh Ramakrishnan. 2025. Param-1 bharatgen 2.9b model.

Varshini Reddy, Craig W Schmidt, Yuval Pinter, and Chris Tanner. 2025. How much is enough? the diminishing returns of tokenization training data. *arXiv preprint arXiv:2502.20273*.

Teven Le Scao, Angela Fan, Christopher Akiki, Ellie Pavlick, Suzana Ilić, Daniel Hesslow, Roman Castagné, Alexandra Luccioni, et al. 2022. Bloom: A 176b-parameter open-access multilingual language model. *CoRR*, abs/2211.05100.

Jean Seo, Jaeyoon Kim, SungJoo Byun, and Hyopil Shin. 2025. How does a language-specific tokenizer affect llms? *arXiv preprint arXiv:2502.12560*.

Marco Siino, Ilenia Tinnirello, and Marco La Cascia. 2024. Is text preprocessing still worth the time? A comparative survey on the influence of popular preprocessing methods on Transformers and traditional classifiers. *Information Systems*, 121(102342):1–19.

Shivalika Singh, 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.

Felix Stollenwerk. 2023. Training and evaluation of a multilingual tokenizer for gpt-sw3. *CoRR*, abs/2304.14780.

NLLB Team, Marta R. Costa-jussà, James Cross, Onur Çelebi, Maha Elbayad, Kenneth Heafield, Kevin Heffernan, Elahe Kalbassi, Janice Lam, Daniel Licht, Jean Maillard, Anna Sun, Skyler Wang, Guillaume Wenzek, Al Youngblood, Bapi Akula, Loic Barrault, Gabriel Mejia Gonzalez, Prangthip Hansanti, John Hoffman, Semarley Jarrett, Kaushik Ram Sadagopan, Dirk Rowe, Shannon Spruit, Chau Tran, Pierre Andrews, Necip Fazil Ayan, Shruti Bhosale, Sergey Edunov, Angela Fan, Cynthia Gao, Vedanuj Goswami, Francisco Guzmán, Philipp Koehn, Alexandre Mourachko, Christophe Ropers, Safiyyah Saleem, Holger Schwenk, and Jeff Wang. 2022. No language left behind: Scaling human-centered machine translation.

Aamod Thakur, Ajay Nagpal, Atharva Savarkar, Kundeshwar Pundalik, Siddhesh Dosi, Piyush Sawarkar, Viraj Thakur, Rohit Saluja, Maunendra Sankar Desarkar, and Ganesh Ramakrishnan. 2025. The art of breaking words: Rethinking multilingual tokenizer design. *arXiv preprint arXiv:2508.06533*.

A Tsun. 2020. Probability and statistics with applications to computing.

Xinyi Wang, Sebastian Ruder, and Graham Neubig. 2021. Multi-view subword regularization. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational*

Linguistics: Human Language Technologies, pages 473–482, Online. Association for Computational Linguistics.

Philip Whittington, Gregor Bachmann, and Tiago Pimentel. 2025. Tokenisation is NP-Complete. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 28133–28153, Vienna, Austria. Association for Computational Linguistics.

Sang Michael Xie, Hieu Pham, Xuanyi Dong, Nan Du, Hanxiao Liu, Yifeng Lu, Percy S Liang, Quoc V Le, Tengyu Ma, and Adams Wei Yu. 2024. Doremi: Optimizing data mixtures speeds up language model pretraining. *Advances in Neural Information Processing Systems*, 36:69798–69818.

Jiasheng Ye, Peiju Liu, Tianxiang Sun, Jun Zhan, Yunhua Zhou, and Xipeng Qiu. 2024. Data mixing laws: Optimizing data mixtures by predicting language modeling performance. *arXiv preprint arXiv:2403.16952*.

Shiyue Zhang, Vishrav Chaudhary, Naman Goyal, James Cross, Guillaume Wenzek, Mohit Bansal, and Francisco Guzman. 2022. How robust is neural machine translation to language imbalance in multilingual tokenizer training? In *Proceedings of the 15th biennial conference of the Association for Machine Translation in the Americas (Volume 1: Research Track)*, pages 97–116, Orlando, USA. Association for Machine Translation in the Americas.

Jingwei Zuo, Maksim Velikanov, Ilyas Chahed, Younes Belkada, Dhia Eddine Rhayem, Guillaume Kunsch, Hakim Hacid, Hamza Yous, Brahim Farhat, Ibrahim Khadraoui, et al. 2025. Falcon-h1: A family of hybrid-head language models redefining efficiency and performance. *arXiv preprint arXiv:2507.22448*.

A Detailed Description of Datasets

Subset name	Language name	Disk size
rus_Cyrl	Russian	1.2T
cmn_Hani	Chinese	784G
deu_Latn	German	618G
spa_Latn	Spanish	515G
jpn_Jpan	Japanese	393G
fra_Latn	French	483G
ita_Latn	Italian	269G
por_Latn	Portuguese	222G
pol_Latn	Polish	168G
nld_Latn	Dutch	160G
ind_Latn	Indonesian	125G
tur_Latn	Turkish	100G
ces_Latn	Czech	104G
fas_Arab	Persian	69G
hun_Latn	Hungarian	79G
swe_Latn	Swedish	61G
ell_Grek	Greek	84G
dan_Latn	Danish	61G
vie_Latn	Vietnamese	59G

Table 4: Language Composition of the FineWeb2-HQ Dataset

FineWeb2-HQ Dataset FineWeb2-HQ is a high-quality, model-filtered pretraining dataset designed for multilingual Large Language Models (LLMs). It is a subset of the FineWeb2 corpus, spanning 19 languages, and was created by selecting the top 10% of documents in each language. The selection was based on scores from a deep learning classifier, which used XLM-RoBERTa embeddings to identify structured and knowledge-rich samples. The language composition of the dataset is detailed in Table 4.

FLORES Dataset The FLORES (Facebook Low-Resource Translation Evaluation) dataset is a multilingual benchmark designed to evaluate machine translation quality between English and low-resource languages. FLORES contains translations of 3,001 sentences sourced from 842 distinct web articles, with each sentence averaging about 21 words.

Pile Dataset The Pile is an open-source dataset consisting of approximately 800 GB of diverse text designed for large-scale language model pretraining. It was curated to provide a high-quality, representative mixture of domains including academic papers (e.g., arXiv), web text (e.g., Wikipedia, StackExchange, HackerNews), code (e.g., GitHub), legal and medical documents, and more. The dataset comprises 22 component corpora, each selected to balance domain diversity and textual quality, enabling models trained on The Pile to develop broad generalization and reasoning abilities across multiple knowledge domains. Due to copyright concerns, we utilize the 17 subsets that do not violate copyright issues.

B Compression Rate Prediction Result for the 1GB–64k Tokenizer

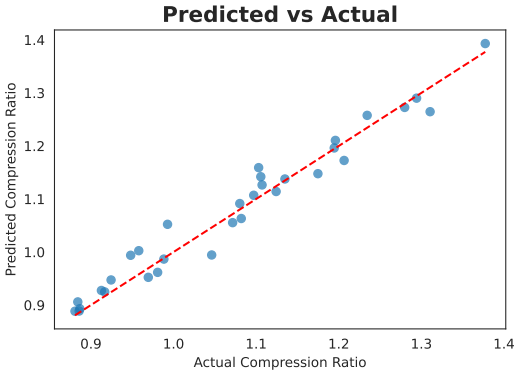

Figure 5: Compression rate prediction results for the 1GB–64k tokenizer using TREX

Figure 5 illustrates the relationship between the actual compression rates and those predicted by the TREX regression model for the 1GB–64k tokenizer. The red dashed line denotes the ideal prediction line, and each blue dot represents the predicted and actual compression rate for a specific data mixture. Most points are closely aligned with the dashed line, indicating that the regression model accurately captures the real compression performance in this small-scale setting. This result confirms that TREX can effectively predict tokenizer compression behavior even with limited-scale training data.

C Rank Invariance

To achieve higher compression performance in tokenizer training, it is essential to find the optimal data mixture $\mathbf{w}$. However, in a full-scale setting, exploring this requires repeatedly training tokenizers on various data mixtures, which demands enormous computational cost and time. To effectively address the significant cost and time challenges, we propose rank invariance in tokenizer space as our core hypothesis. Rank invariance refers to the property that the performance ranking of tokenizers according to data mixture strategies $\mathbf{w}$ remains unchanged even when the training scale differs. For example, given two arbitrary data mixtures $\mathbf{w}_i$ and $\mathbf{w}_j$, if their compression performance rank-

ing in small-scale settings (S_s, V_s) is preserved in full-scale settings (S_f, V_f), this relationship can be expressed as follows:

$$C_{\mathbf{w}_i}(S_s, V_s) > C_{\mathbf{w}_j}(S_s, V_s) \iff C_{\mathbf{w}_i}(S_f, V_f) > C_{\mathbf{w}_j}(S_f, V_f) \quad (8)$$

To verify whether our proposed hypothesis holds in practice, we sampled 32 data mixtures across 19 languages and trained 32 tokenizers at various scales, ranging from $S = 1\text{GB}, V = 64\text{k}$ to $S = 30\text{GB}, V = 200\text{k}$. We then measured the compression performance of each trained tokenizer. To quantitatively assess how consistent the performance rankings are across different scale settings, we employed the Spearman Rank Correlation. Unlike metrics that focus on absolute performance differences, the Spearman Rank Correlation measures the correlation between rankings, making it the most suitable choice for numerically validating our core hypothesis of rank invariance. Figure 2 presents the resulting heatmap. Across all setting pairs, the correlation coefficients were above 0.97, indicating strong consistency, even between the smallest and largest scales. This supports the Rank Invariance hypothesis and suggests that the optimal mixture in large-scale settings can be reliably predicted using only small-scale experiments.

D Evaluation of Regression Model in Large-Scale Settings

In the previous section, we confirmed that the proposed Rank Invariance hypothesis holds in actual tokenizer training. A natural follow-up question is whether this invariance also extends to the relationship between the regression model's predicted compression and the actual compression scores. To examine this, we conducted an experiment using a regression model trained at the proxy level—that is, under a small-scale configuration—to evaluate whether the rank correlation observed at small scales persists in real tokenizer performance.

Table 5 presents the results of regression models evaluated under both multilingual and medical-specific settings. Across diverse scale configurations, the predicted compression scores from the regression model exhibited a consistently high rank correlation ρ with the actual tokenizer compression results. These findings indicate that the Rank Invariance—previously observed in real tokenizer training—also holds within the TREX's regression model, reinforcing its validity as a scalable and reliable predictor of tokenizer performance.

Test Setting		Correlation (ρ)	MAPE
Data Size	Vocabulary	↑	↓
Multilingual Domain			
1GB	64k	0.979	1.989
5GB	64k	0.970	-
10GB	100k	0.967	-
30GB	200k	0.960	-
Medical Domain			
1GB	64k	0.981	0.921
5GB	64k	0.968	-
10GB	100k	0.970	-
30GB	200k	0.967	-

Table 5: Performance of the TREX regression model across different corpus and vocabulary scales under multilingual and medical-domain settings. The model consistently exhibits high Spearman rank correlation ($\rho \geq 0.96$) between predicted and actual compression values, demonstrating that the Rank Invariance holds across both domains.

E Tokenizer Compression Score in Medical Domain

In Section 6, we further applied TREX to the medical domain to examine whether the optimal data mixture predicted by TREX leads to improved tokenizer performance. Table 6 presents the experimental results, showing that the tokenizer trained with the TREX-predicted mixture achieved a slight but consistent improvement in compression performance compared to existing baselines. These findings demonstrate that TREX is effective not only in multilingual settings but also in domain-specific scenarios such as the medical domain.

	Baseline	TREX
Compression	0.911	**0.904**

Table 6: Tokenizer compression in medical domain

F Cost Efficiency of TREX in Finding Optimal Data Mixture

A potential concern regarding TREX is the initial overhead of building 480 proxy tokenizers to train its regression model. To address the question of its time cost-efficiency, we compared the total time required to obtain a final tokenizer with that of a recent iterative method, AdaptMix (Thakur et al., 2025), which requires approximately 20 iterations on the full-scale settings (S=30GB, V=200k).

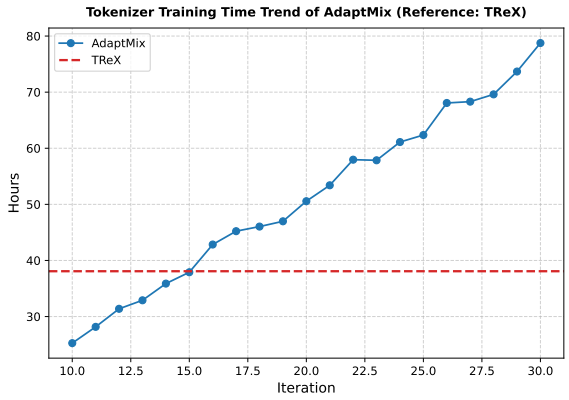

Figure 6: Time Consumption Comparison between AdaptMix and TREX

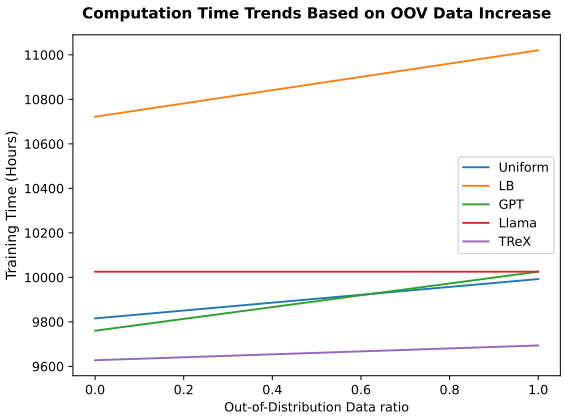

Figure 7: The x-axis represents the proportion of Out-of-Distribution (OOD) data within the pretraining corpus, and the y-axis denotes the corresponding training time. As shown, the tokenizer trained with TREX exhibits a slower increase in training time as the OOD ratio grows, compared to other tokenizers.

As illustrated in Figure 6, our approach reduces the total training time by approximately 41 hours compared to AdaptMix.

Moreover, the advantage of TREX goes beyond mere time efficiency. Once trained, the regression model can rapidly simulate the performance of diverse data mixtures, enabling broad exploration of mixture ratios to identify globally optimal tokenizer configurations at negligible additional cost, a capability fundamentally distinct from iterative approaches that incur new costs for every trial.

G Cost Efficiency of TREX in Language Model Training

As established in the previous section, an improved tokenization compression rate directly leads to higher efficiency in language model training. To quantify this effect, we designed an experiment to further illustrate the following point: a tokenizer with an improved compression encodes the same amount of raw text into fewer tokens, thereby reducing the total training FLOPs and, consequently, the overall training time. We estimated this impact by first calculating a baseline — the total time required to train a model on 3 trillion (3T) tokens using a Uniform tokenizer on a cluster of 32 H100 GPUs.

Figure 7 illustrates the projected time savings for various tokenizers based on their respective compression rates relative to this baseline. These results offer a concrete view of how tokenization compression impacts real-world training efficiency. Although the numerical differences in compression rates may appear small, the figure clearly shows that they yield substantial reductions in total training time, underscoring the significant efficiency gains achieved by the tokenizer optimized through TREX.

H Language Composition of Data Mixtures

This section provides a visual analysis of the composition of each mixture used in Table 2. $\mathbf{w}^{LB}$, $\mathbf{w}^{gpt}$ and $\mathbf{w}^{llama}$ cover a wide range of languages, but in this study, we focus only on the languages included in the previously described dataset. Figures 8, 9, 10, 11, and 12 show pie charts visualizing the language proportions of each mixture.

Correlation Between Language Distribution and Compression As summarized in Figure 3 and Table 2, the five mixture models exhibit markedly different language distributions. $\mathbf{w}^{uni}$ allocates roughly equal proportions (5.2%) to all languages, whereas $\mathbf{w}^{LB}$ assigns greater weight to mid-frequency languages (e.g., DEU, FRA, NLD) while underrepresenting low-resource ones. $\mathbf{w}^{gpt}$ concentrates heavily on Latin-based languages such as French, Portuguese, and Spanish, while $\mathbf{w}^{llama}$ is biased towards non-Latin scripts (e.g., Japanese, Persian, Turkish).

In contrast, $\mathbf{w}^{\text{TREX}}$ maintains a balanced distribution between Latin and non-Latin language groups, while strategically increasing the presence of languages with lower domain redundancy (e.g., French, Russian, Portuguese). The resulting language distributions exhibit a statistically significant negative correlation with actual compression

LANG	CHAR	$\mathbf{w}^{uni}$	$\mathbf{w}^{LB}$	$\mathbf{w}^{gpt}$	$\mathbf{w}^{llama}$	$\mathbf{w}^{*}$
CES	Latn	0.052	0.076	0.014	0.085	0.020
CMN	Hani	0.052	0.004	0.039	0.009	0.045
DAN	Latn	0.052	0.061	0.019	0.015	0.026
DEU	Latn	0.052	0.086	0.088	0.069	0.087
ELL	Grek	0.052	0.014	0.029	0.053	0.026
FAS	Arab	0.052	0.007	0.019	0.069	0.019
FRA	Latn	0.052	0.085	0.142	0.057	0.184
HUN	Latn	0.052	0.072	0	0	0.003
IND	Latn	0.052	0.073	0.019	0.009	0.021
ITA	Latn	0.052	0.062	0.024	0.031	0.073
JPN	Jpan	0.052	0.002	0.019	0.130	0.062
NLD	Latn	0.052	0.062	0.098	0.028	0.044
POL	Latn	0.052	0.057	0.024	0.047	0.037
POR	Latn	0.052	0.063	0.113	0.044	0.070
RUS	Cyrl	0.052	0.017	0.137	0.107	0.130
SPA	Latn	0.052	0.070	0.137	0.063	0.058
SWE	Latn	0.052	0.060	0.014	0.015	0.030
TUR	Latn	0.052	0.059	0.029	0.101	0.013
VIE	Latn	0.052	0.062	0.024	0.057	0.042

Table 7: Overview of data mixtures used in our experiments. Each mixture defines a unique weighting configuration across multiple language corpora.

efficiency ($r = -0.47$, $p < 0.05$). This implies that moderately increasing the proportion of certain languages tends to improve their compression performance (i.e., token length efficiency). Such results suggest that when a tokenizer sufficiently learns the segment redundancy of each language, subword segmentation becomes more stable.

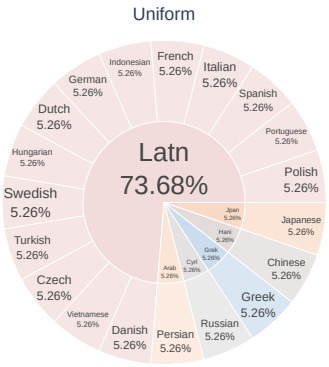

Figure 8: Data mixture of Uniform

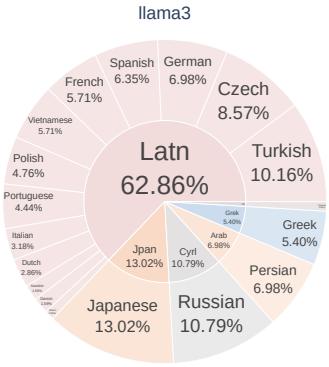

Figure 11: Data mixture of llama3 (entries below 1.7% are omitted)

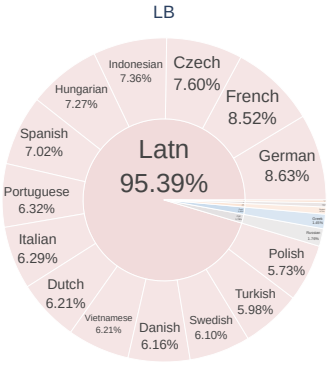

Figure 9: Data mixture of LB (entries below 1.7% are omitted)

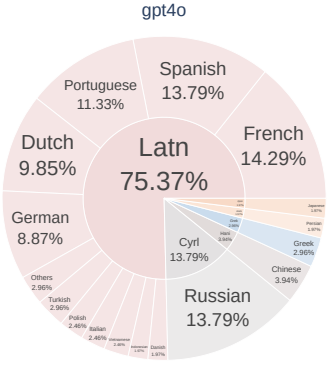

Figure 10: Data mixture of gpt-4o (entries below 1.7% are omitted)

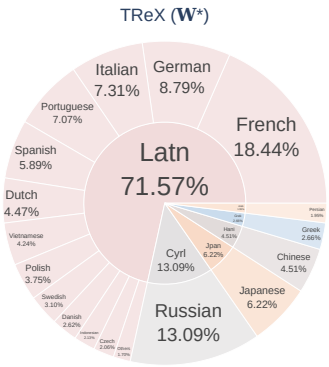

Figure 12: Data mixture of TREX (entries below 1.7% are omitted)

I Mixture Diversity Analysis

To understand how language mixture diversity affects tokenizer efficiency, we analyze the statistical relationship between the entropy of each mixture distribution and its compression performance. Figure 13 illustrates that mixtures with moderate entropy neither fully uniform nor overly skewed achieve the most efficient tokenization. This trend suggests that balanced yet biased allocation of languages improves subword segmentation across multilingual corpora.

Observation Uniform and LLaMA3 exhibit high mixture entropy (0.91–1.00), but yield only moderate compression efficiency (0.888–0.907). TREX, however, maintains moderate entropy (0.90) while achieving the best average compression ratio (0.871) and the lowest non-Latin subset average (0.814). This demonstrates that TREX learns an *efficient bias* rather than relying on uniformity or hand-crafted heuristics.

Implication These results empirically validate that mixture optimization benefits from data-driven modeling of the non-linear relationship between language proportion and compression efficiency. In particular, the optimal point lies between complete uniformity and high skewness, highlighting the role of predictive mixture learning in multilingual tokenizer design.

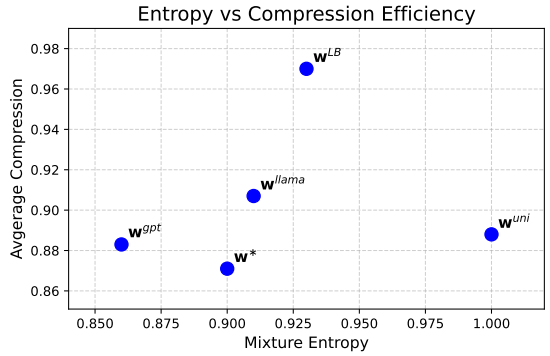

Figure 13: Relationship between mixture entropy (x-axis) and average compression ratio (y-axis) across five tokenizer configurations. A moderate level of entropy yields the most efficient tokenization, indicating that uniform distribution is not necessarily optimal for multilingual mixtures.

I.1 Impact of tokenization and preprocessing on model behavior

Recent work has shown that tokenization and text preprocessing play a far more active role in shaping language model behavior than previously assumed. Lesci et al. (2025) identify and quantify tokenization bias, showing that models trained with different vocabularies assign markedly different probabilities to the same character sequences. Framing this discrepancy as a causal effect, they estimate that the inclusion or exclusion of a single subword in a tokenizer's vocabulary can alter a model's predicted probability for its corresponding string by up to seventeenfold, revealing that tokenization is not a neutral preprocessing choice but a key determinant of model output. Whittington et al. (2025) complement this empirical perspective with a formal analysis, demonstrating that the problem of optimal tokenization—whether defined in terms of vocabulary composition or merge sequence—is NP-complete. Their results explain why heuristic approaches such as byte-pair encoding and UnigramLM dominate in practice and highlight the inherent computational difficulty of designing universally optimal tokenizers. At a more applied level, Siino et al. (2024) show that even seemingly minor preprocessing operations, including normalization, noise reduction, and punctuation handling, can substantially affect downstream performance, sometimes yielding differences exceeding 25% in classification accuracy. Their findings reaffirm that preprocessing decisions remain critical in neural pipelines, affecting not only model robustness but also interpretability and computational efficiency. Taken together, these studies situate tokenization and preprocessing at the intersection of theory, practice, and linguistic representation, demonstrating that choices made at the input level fundamentally influence both the statistical and structural behavior of language models.

J Evaluation of Downstream Performance

To address concerns regarding the correlation between tokenizer compression and downstream model performance, we conducted additional experiments to verify whether the increased compression efficiency of TREX affects the representative capabilities of Large Language Models.

Due to computational resource constraints, we performed evaluations under a budget-limited setting. We trained a 200M-parameter Transformer model from scratch using two different tokenizers: the standard LLaMA tokenizer (as a baseline) and the TREX-optimized tokenizer mixture.

Mixture Model	Entropy (↑)	Avg. Compression (↓)	Non-Latin Avg. (↓)	Δ vs. Uniform (↓)
$\mathbf{w}^{uni}$	1.00	0.888	0.848	–
$\mathbf{w}^{LB}$	0.93	0.970	1.076	+0.082
$\mathbf{w}^{gpt}$	0.86	0.883	0.831	-0.005
$\mathbf{w}^{llama}$	0.91	0.907	0.863	+0.019
$\mathbf{w}^{\text{TREX}}$	0.90	**0.871**	**0.814**	**-0.017**

Table 8: Statistical summary of mixture entropy and compression performance across five tokenizer configurations. Entropy measures the diversity of language proportions within each mixture. TREX achieves the best overall and non-Latin compression efficiency while maintaining moderate entropy, demonstrating that effective bias rather than uniformity leads to superior multilingual tokenization.

To ensure a fair comparison of training volume, both models were trained on a fixed subset of the FineWeb2-HQ dataset, totaling 50 billion tokens as measured by the LLaMA tokenizer. This ensures that both models were exposed to the same amount of raw text data. For downstream evaluation, we utilized the Global MMLU (Singh et al., 2025) benchmark across 16 different languages to assess the model's cross-lingual reasoning and factual knowledge.

The results, summarized in Table 9, demonstrate that the model trained with the TREX tokenizer consistently achieves comparable or superior performance to the baseline across all tested languages. While the absolute performance gains are modest due to the restricted model size, the consistent trend suggests that TREX's compression efficiency does not degrade—and in many cases enhances—the downstream utility of the language model.

Language	$\mathbf{w}^{llama}$	$\mathbf{w}^{\text{TREX}}$	Diff.
Russian (RUS)	22.95	**23.46**	+0.51
Polish (POL)	23.10	**23.75**	+0.65
Chinese (CMN)	22.91	**23.09**	+0.18
Dutch (NLD)	**24.16**	23.69	-0.47
German (DEU)	22.89	**24.17**	+1.28
Indonesian (IND)	22.95	**26.75**	+3.80
Japanese (JPN)	22.92	**22.95**	+0.03
Turkish (TUR)	22.97	**24.12**	+1.15
French (FRA)	23.31	**23.99**	+0.68
Czech (CES)	23.05	**23.17**	+0.12
Italian (ITA)	22.90	**23.56**	+0.66
Persian (FAS)	**22.97**	22.93	-0.04
Portuguese (POR)	**22.97**	22.95	-0.02
Swedish (SWE)	24.09	**24.51**	+0.42
Greek (ELL)	22.92	**23.01**	+0.09
Vietnamese (VIE)	24.72	**25.14**	+0.42
Average	23.24	**23.83**	**+0.59**

Table 9: Global MMLU performance of 200M-parameter models. Both models were trained on 50 billion tokens, with the dataset size determined based on the LLaMA tokenizer to ensure exposure to the same amount of raw text.