SampleMix: A Sample-wise Pre-training Data Mixing Strategy by Coordinating Data Quality and Diversity

Xiangyu Xi[1*], **Deyang Kong** [1,2*], **Jian Yang**[1*], **JiaWei Yang**[1], **Zhengyu Chen**[1], **Wei Wang**[1†], **Jingang Wang**[1], **Xunliang Cai**[1], **Shikun Zhang**[2], **Wei Ye**[2†]
[1] Meituan Group, Beijing, China
[2] National Engineering Research Center for Software Engineering, Peking University, Beijing, China
xixy10@foxmail.com
Project HomePage: https://meituan-group.notion.site/samplemix

Abstract

Existing pretraining data mixing methods for large language models (LLMs) typically follow a domain-wise methodology, a top-down process that first determines domain weights and then performs uniform data sampling across each domain. However, these approaches neglect significant inter-domain overlaps and commonalities, failing to control the global diversity of the constructed training dataset. Further, uniform sampling within domains ignores fine-grained sample-specific features, potentially leading to suboptimal data distribution. To address these shortcomings, we propose a novel sample-wise data mixture approach based on a bottom-up paradigm. This method performs global cross-domain sampling by systematically evaluating the quality and diversity of each sample, thereby dynamically determining the optimal domain distribution. Comprehensive experiments across multiple downstream tasks and perplexity assessments demonstrate that SampleMix surpasses existing domain-based methods. Meanwhile, SampleMix requires 1.4x to 2.1x fewer training steps to achieve the baselines' performance, highlighting the substantial potential of SampleMix to optimize pre-training data.

1 Introduction

The mixture proportions of pretraining data, which greatly affect the language model performance, have received increasing attention from researchers and practitioners. In the early years, heuristic-based methods were widely employed to assign domain weights using manually devised rules, such as upsampling high-quality datasets (e.g., Wikipedia) multiple times (Gao et al., 2020; Laurençon et al., 2022). Afterwards, models like GLaM (Du et al., 2022) and PaLM (Chowdhery et al., 2023) established mixture weights based on the performance

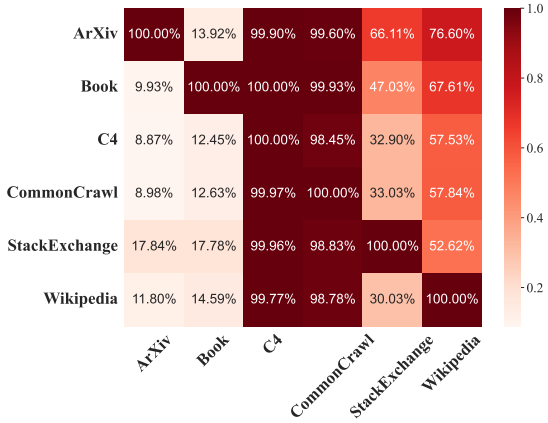


Figure 1: We conduct data clustering analysis using the SlimPajama dataset. For each domain (row), each cell shows the percentage of its clusters that also include samples from other domains (column). E.g., 76.60% of ArXiv's clusters include Wikipedia samples (1st row, 6th column). The results reveal substantial overlap between domains.

metrics of trained smaller models. More recently, learning-based methods have been proposed, involving the training of small proxy models across domains to generate optimal domain weights (Fan et al., 2023; Xie et al., 2024). These existing methods follow a domain-wise methodology, a top-down process that first determines the proportion of each domain and then samples uniformly from the selected domain. Despite achieving advancements, These approaches present two key issues:

(1) Ignoring Inter-domain Overlaps and Commonalities. In current pretraining datasets, "domain" is primarily categorized based on data sources rather than intrinsic textual or semantic properties. An implicit assumption of the domain-wise approaches is that samples are distinct and unrelated across domain boundaries. However, in practice, samples across different domains exhibit significant shared characteristics, both in terms of raw text and high-level semantics. To examine this

*The first three authors contributed equally.
†Corresponding authors.

Findings of the Association for Computational Linguistics: EMNLP 2025, pages 13736–13758
November 4-9, 2025

assumption, we analyzed the SlimPajama dataset (Soboleva et al., 2023), a quality-filtered and deduplicated dataset, focusing on relationships between samples and clusters across its six text domains (excluding GitHub). For each domain, we computed the percentage of its clusters that also included samples from other domains, as Figure 1 shows. Our findings reveal substantial overlap between domains—nearly all clusters contain samples from both CommonCrawl and C4. Furthermore, manual inspection of the clustered samples confirms that data from different domains frequently share similar topics and characteristics. For instance, Figure 6 illustrates that samples from multiple domains discuss Einstein and the Theory of Relativity. By disregarding inter-domain commonalities, domain-wise mixture methods fail to control the global diversity of training data effectively.

(2) Suboptimal Sample Distribution within Domains. A second limitation arises from the uniform sampling within each domain, which can lead to a suboptimal distribution of training samples (Xie et al., 2024; Fan et al., 2023; Ye et al., 2025). Intuitively, samples with higher quality and greater diversity should have a higher probability of being selected (Xie et al., 2023; Abbas et al., 2023). At the same time, lower-quality samples should not be entirely discarded, as they contribute to the model's generalization ability (Sachdeva et al., 2024). Determining an effective sampling strategy within each domain is nontrivial, yet current approaches lack fine-grained control over sample selection.

To address these limitations, we propose a novel sample-wise data mixture approach with a bottom-up paradigm. Instead of defining domain proportions upfront, we first perform global sampling across the dataset based on sample quality and diversity, dynamically determining domain distributions. This allows for more precise control over the overall quality and diversity of the dataset. To implement this, we individually assess the quality and diversity of each sample and assign corresponding sampling weights based on these evaluations. Given a target token budget, we then sample each example according to its weight to construct the optimal training dataset. Also, our approach offers the additional advantage of dynamically adapting to varying token budgets, enabling the determination of optimal data proportions for each specific budget. In contrast, the vast majority of existing works rely on static data proportions, which do not adjust to different token budget constraints. The contributions of this paper are:

1. We study the problem of sample-wise pre-training data mixing, which can alleviate the limitations of overlooking inter-domain overlap and suboptimal sample distribution within domains by existing domain-wise methods.

2. We propose a sample-wise pre-training data mixing strategy that coordinates data quality and diversity on a per-sample basis, effectively capturing commonalities among domains and optimal sample distribution.

3. Extensive experiments on downstream tasks and perplexity evaluations demonstrate the advantages of our method. Notably, it achieves averaged baseline accuracy with 1.9x fewer training steps, highlighting its efficiency.

2 Method

2.1 Problem Formulation

Consider a source dataset D_{src} composed of k distinct domains (e.g., CommonCrawl, Wikipedia, BookCorpus, etc.). For each domain i, let D_i denote the collection of documents within that domain. The entire source dataset is defined as $D_{\text{src}} \triangleq \{D_1, \ldots, D_k\}$, with T_{src} representing the total number of tokens. Our objective is to construct a target training set D_{tgt} for pre-training that adheres to a specific token budget T_{tgt} (e.g., 100B tokens). As illustrated in Figure 2, traditional approaches determine domain weights without explicitly considering the overall token budget, and build D_{tgt} by uniform sampling from each domain based on these weights. In contrast, our proposed method, SampleMix, enhances this process by evaluating both the quality (§ 2.2) and diversity (§ 2.3) of each document. Utilizing these dual criteria, SampleMix assigns unique sampling weights to each document. To ensure compliance with the token budget T_{tgt}, we then construct an optimal training dataset by sampling documents according to their assigned weights (§ 2.4).

2.2 Data Quality Evaluation

The quality of training data is crucial for large language models. However, most existing studies typically rely on simple heuristics (Xie et al., 2023; Li et al., 2023; Sachdeva et al., 2024). Wettig et al. (2024) introduces four metrics and uses pairwise

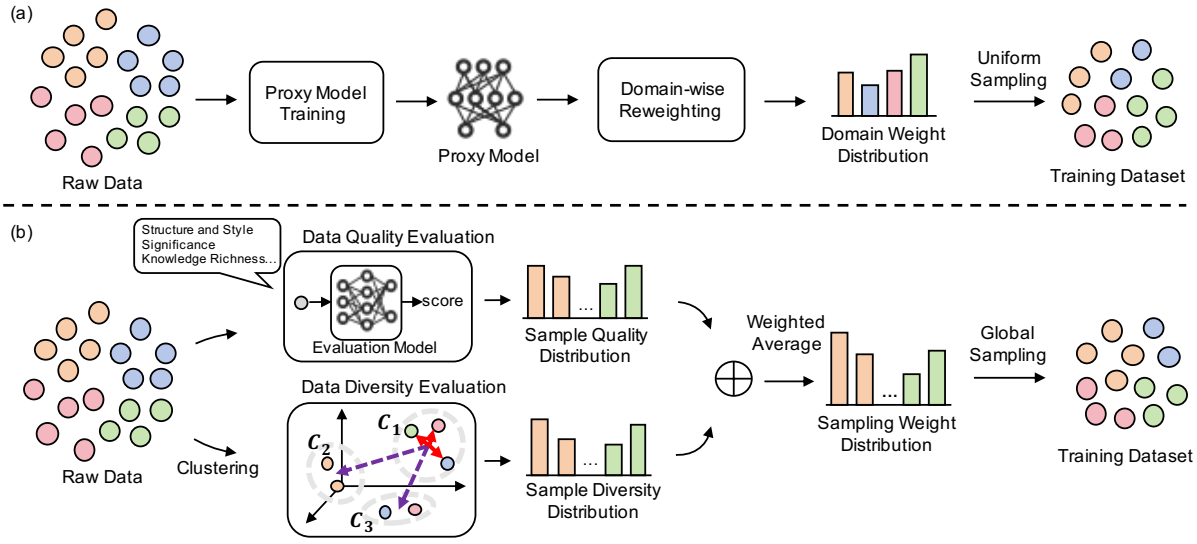


Figure 2: (a) Traditional methods determine domain weights and construct the training dataset by uniformly sampling from each domain. (b) *SampleMix* employs a **sample-wise** mixing strategy by: evaluating sample quality and diversity, assigning appropriate weights, and constructing an optimal dataset based on these weights. Dots of the same color represent data from the same domain.

comparisons to train an evaluator model. However, these metrics are applied separately in data selection, and pairwise training may neglect the objective factors that determine sample quality.

2.2.1 Quality Criteria

To comprehensively capture both the fundamental linguistic attributes and the deeper informational and analytical qualities of the text, we assert that high-quality data should adhere to the following principles: linguistic precision and clarity, structural coherence and completeness, content reliability and appropriateness, informational and educational value, as well as significance and originality. To evaluate these aspects effectively, we propose 7 quality dimensions accompanied by corresponding scores based on the aforementioned principles, as outlined in Table 1. Notably, for *Knowledge Richness* and *Logicality and Analytical Depth*, we utilize a larger scoring span to address the wider range and greater complexity inherent in these features. By aggregating all dimension scores, we obtain an overall quality evaluation for each sample, ranging from 0 to 10.

2.2.2 Quality Evaluator

To develop an effective and efficient quality evaluator, we utilize GPT-4o to assess training data based on predefined quality criteria (prompt shown in Fig 8). Specifically, we uniformly sample 420k documents from the SlimPajama dataset, allocating 410k and 10k documents for train and test set respectively [*]. We train the quality evaluator with `gte-en-mlm-base` model (Zhang et al., 2024) as the backbone. Instead of text classification tasks, we employ ordinal regression to leverage the inherent ordering of quality scores. Following Niu et al. (2016), we transform ordinal regression into a series of binary classification problems, each indicating whether the input data exceeds a specific quality threshold. The overall quality score is then derived by subtracting the sequence of binary outputs (code shown in Appendix E).

We evaluate the trained quality evaluator on the test set, as shown in Table 2. Instead of relying solely on Accuracy (ACC), we consider Mean Squared Error (MSE) and Mean Absolute Error

Dimension	Score
Clarity of Expression and Accuracy	{0,1}
Completeness and Coherence	{0,1}
Structure and Style	{0,1}
Content Accuracy and Credibility	{0,1}
Significance	{0,1}
Knowledge Richness	{0,1,2}
Logicality and Analytical Depth	{0,1,2,3}

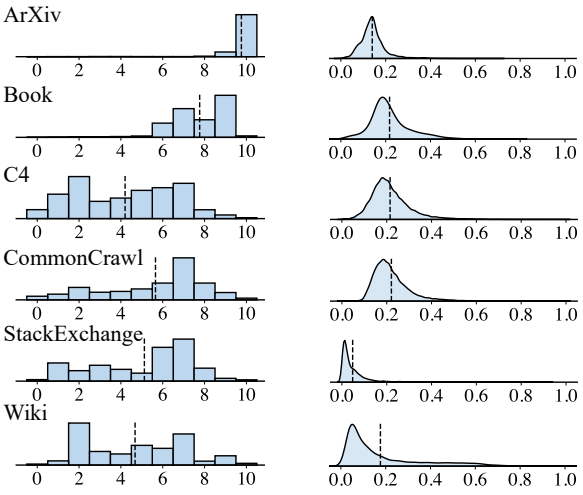
Table 1: Quality dimensions and scores.

[*]The GPT-4o cost is $1873 (see Appendix D for details), aligns with standards seen in related studies (Wettig et al., 2024; Gunasekar et al., 2023) and does not substantially impact the overall cost.

Model	Text Classification	Ordinal Regression
ACC	56.14	55.94
MAE	0.77	0.72
MSE	1.95	1.57
CACC	82.24	83.37

Table 2: Performance comparison between text classification and ordinal regression models on the test set.

(MAE), which more accurately reflect the degree of deviation between the true quality scores and the predicted results. While both the text classification and ordinal regression approaches achieve similar accuracy, the ordinal regression method demonstrates superior performance in terms of MSE and MAE. We noticed that the accuracy is lower than anticipated; detailed analysis shows that most false predictions fall within ±1 of the true quality score. To address this, we introduce Close Accuracy (CACC), a relaxed metric where a prediction is considered correct if it is within ±1 of the true quality score. The CACC results indicate that our model possesses satisfactory discriminatory ability for samples of different qualities.



(a) Quality Distribution (b) Diversity Distribution

Figure 3: Analysis of SlimPajama dataset. Mean values are marked with a dashed line.

2.2.3 Analysis of Quality Distribution

Using the trained quality evaluator, we annotate the SlimPajama dataset and perform extensive case studies, which show that the evaluator can effectively distinguish between high-quality samples (scientific literature, knowledge reports, etc.) and low-quality samples (advertisements, incomplete web pages, etc.), as exampled by Figure 9. The quality distribution is presented in Figure 3a, from which we can find: (1) Arxiv and Book sources exhibit higher quality, as anticipated. (2) Wikipedia is generally considered a high-quality source; however, a substantial portion is of lower quality. Our manual inspection indicates that these low-quality samples typically consist of brief, parsing errors, incomplete content, and other issues. (3) Overall, the CommonCrawl dataset outperforms C4 in terms of quality (average quality score: 5.65 v.s. 4.20).

2.3 Data Diversity Evaluation

Inspired by Shao et al. (2024a) and Abbas et al. (2024), we employ data clustering to capture the text distribution within our training dataset. Through a detailed analysis of the clustered samples, we observe patterns consistent with Abbas et al. (2024)'s work on image data, specifically: (1) Denser clusters exhibit higher similarity among their constituent samples; (2) Clusters that are proximal to others are more likely to contain samples resembling those in neighboring clusters. To quantify data diversity, we estimate a diversity measure for each sample using the Diversity Evaluator.

2.3.1 Diversity Evaluator

Data Clustering We begin by generating embeddings for each sample, which are subsequently organized into clusters via K-Means, effectively structuring the data based on textual similarity. The details of data clustering can be found in Appendix G.

Cluster Compactness We assess the density of a cluster by calculating the average distance of its members from the centroid, referred to as Cluster Compactness. A smaller average distance signifies a more compact cluster, indicating higher similarity among its constituent samples. This metric effectively reveals the dense property of the cluster.

Cluster Separation We evaluate the distinctiveness of each cluster by measuring the distance between its centroid and those of other clusters, termed Cluster Separation. Larger distances imply greater separation, indicating that the cluster is more distinct from others and highlighting its uniqueness on a global scale.

Data Diversity Calculation Finally, the diversity of each sample x_i is estimated by integrating its cluster's separation and compactness as follows:

$$d(x_i) = d_{\text{compactness},j} \times d_{\text{separation},j} \quad (1)$$

where x_i belongs to the j-th cluster, $d_{\text{compactness},j}$ and $d_{\text{separation},j}$ represents the cluster compactness

and cluster separation for the j-th cluster respectively. This composite diversity measure effectively encapsulates both the homogeneity within clusters and the distinctiveness between clusters, providing a comprehensive assessment of data diversity. We discuss the key distinctions of our technical designs from previous works in Appendix H.

2.3.2 Analysis of Diversity Distribution

Extensive case studies demonstrate that our diversity evaluator reliably assigns higher scores to distinctive content, such as academic reports, in-depth analyses, and rare knowledge—examples of which are shown in Figure 10. Conversely, it consistently gives lower scores to content that is overly common or widely circulated online, such as advertisements, duplicate product manuals, and routine sports score updates. To further assess our evaluator, we analyze the diversity distribution within the SlimPajama dataset, as illustrated in Figure 3b. We find that: (1) Within individual domains, samples' diversity can vary significantly. For instance, the diversity distribution of C4 approximates a normal distribution, indicating consistent variability within this domain. (2) Diversity differs markedly across domains in the SlimPajama dataset. Specifically, the C4, CommonCrawl, and Book domains exhibit the highest levels of diversity, as anticipated. In contrast, the StackExchange domain demonstrates the lowest diversity among the examined domains.

2.4 Data Sampling

2.4.1 Sampling Weight Calculation

Given the quality and diversity evaluation for each document, we first min-max normalize the dual measures to ensure they lie within the interval $[0, 1]$ and compute the sampling weight as follows:

$$p(x_i) = \alpha\, d(x_i) + (1 - \alpha)\, q(x_i) \tag{2}$$

where $q(x_i)$ and $d(x_i)$ denote quality and diversity measure of the document x_i, and $\alpha \in [0, 1]$ is the weighting factor that balances the contribution of diversity relative to quality.

2.4.2 Determining Sampling Frequency

Given the source dataset D_{src} containing $|D_{\text{src}}|$ documents with T_{src} tokens, we first estimate the target number of documents for D_{tgt} as follows:

$$|D_{\text{tgt}}| = \frac{T_{\text{tgt}}}{T_{\text{src}}} |D_{\text{src}}| \tag{3}$$

Then we compute each document's sampling frequency $c(x_i)$ using a softmax-based distribution to translate the sampling weights into probabilities:

$$c(x_i) = |D_{\text{tgt}}| \times \frac{\exp\left(p(x_i)/\tau\right)}{\sum_{j \in D_{\text{src}}} \exp\left(p(x_i)/\tau\right)} \tag{4}$$

where τ is the temperature parameter that modulates the softmax distribution, controlling the concentration of the sampling probabilities.

2.4.3 Constructing the Training Dataset

Since $c(x_i)$ typically yields non-integer values, we convert these frequencies into integer counts through the following two-step process:

- Integer Part: Always sample the document $\lfloor c(x_i) \rfloor$ times. For example, if $c(x_i) = 2.3$, the document is sampled 2 times.
- Fractional Part: The remaining fractional part $(c(x_i) - \lfloor c(x_i) \rfloor)$ is used to determine an additional sample probabilistically. Continuing the example, with $c(x_i) = 2.3$, there is a 30% chance that x_i will be sampled a third time, determined by comparing the fractional part to a randomly generated number.

By aggregating the sampled counts for each document x_i, we assemble the final training dataset D_{tgt}, which closely matches the target token budget T_{tgt}. Our method offers key benefits: (1) Prioritization of Quality and Diversity: By incorporating both quality and diversity metrics into the sampling weights, SampleMix ensures that high-quality and diverse documents are preferentially selected, enhancing the overall effectiveness of the training dataset. (2) Adaptive to Training Budget: The sampling mechanism dynamically adjusts to different token budgets T_{tgt}, maintaining an optimal balance between quality and diversity without the need for manual tuning. (3) Flexible Domain Representation: By allowing different sampling rates within the same domain, the method supports a more nuanced representation of various domains.

3 Experimental Setup

3.1 Dataset And Baselines

Following Xie et al. (2024); Ge et al. (2024), we experiment with the SlimPajama dataset (Soboleva et al., 2023), which consists of 7 domains. We compare with the following baselines: (1) **Vanilla**, which denotes the inherent proportions of datasets,

Benchmark	Vanilla	DoReMi	CE	BiMIX-OPT	DoGE	DML	SampleMix
Downstream Tasks Evaluation (Accuracy)							
OpenBookQA	31.40	31.60	31.80	29.80	29.00	30.80	**32.60**
LAMBADA	38.27	40.95	**42.23**	38.02	37.07	35.40	40.69
PiQA	70.40	70.13	69.37	69.64	70.62	65.02	**70.95**
ARC-Easy	47.44	46.65	46.73	45.57	45.74	47.49	**48.73**
ARC-Challenge	28.58	27.30	28.33	28.33	27.65	27.73	**29.86**
WinoGrande	52.33	**54.38**	51.07	52.80	51.14	51.46	53.83
WiC	50.47	48.59	48.28	48.90	50.00	**52.98**	51.72
RTE	50.18	51.62	51.62	47.65	51.26	51.62	**53.79**
Average	46.13	46.40	46.18	45.09	45.31	45.31	**47.77**
Perplexity Evaluation (Perplexity)							
Pile	26.93	26.45	26.20	27.47	29.49	29.76	**25.63**
xP3	47.38	47.08	47.62	48.74	48.38	54.00	**46.38**

Table 3: Comparison of data mixture methods across various downstream tasks and perplexity evaluations. The best performing method for each metric is highlighted in **bold**, while the second-best is underlined.

mirroring the natural distribution patterns (Sobolev et al., 2023). (2) **DoReMi**, which exploits a learning-based solution for multi-round mixture optimization (Xie et al., 2024). (3) **CE**, which uses the Conditional Entropy proxy for data mixture optimization (Ge et al., 2024). (4) **BiMIX-OPT**, which derives the optimized data mixture by the bivariate scaling law (Ge et al., 2024). (5) **DoGE**, which determines the domain weight based on contribution to final generalization objective (Fan et al., 2023). (6) **DML**, which derives the optimized data mixture by the data mixing law (Ye et al., 2025). Note that we focus primarily on text data mixing. Following Liu et al. (2025), we exclude the GitHub domain and apply re-normalization to the baseline weights (the weights are shown in Figure 13). The rationality for re-normalization is detailed in Appendix K. Investigating code data mixing remains an avenue for future research.

3.2 Training Setup

We train 1B-parameters LLaMA models (Dubey et al., 2024) from scratch with 100B tokens. Given that the source dataset (SlimPajama) comprises 503M documents totaling approximately 500B tokens, SampleMix generated the final training dataset consisting of 100M documents, with α and τ set to 0.8 and 0.2 respectively. Detailed hyperparameters, including model architecture, learning rate, etc, are provided in Table 9.

3.3 Evaluation

Downstream Task Accuracy Following Chen et al. (2025), we select 8 extensive downstream tasks, covering commonsense reasoning, language understanding, logical inference and general QA: OpenBookQA (Mihaylov et al., 2018), LAMBADA (Paperno et al., 2016), PiQA (Bisk et al., 2020), ARC-Easy, ARC-Challenge (Clark et al., 2018), WinoGrande (Sakaguchi et al., 2021), and tasks from the SuperGLUE benchmark (Wang et al., 2019).

Validation Set Perplexity Following Ye et al. (2025), we compute perplexity on validation sets from *The Pile* (Gao et al., 2020) to simulate separate collection of training and validation data. This metric measures the model's ability to predict text sequences accurately across various domains, reflecting its general language modeling proficiency.

Instruction Tuning Perplexity Following Tirumala et al. (2023), we evaluate perplexity on the instruction tuning dataset xP3 (Muennighoff et al., 2022) to address the high variance in downstream tasks. This evaluation gauges the model's effectiveness in understanding and following instructions.

4 Results and Analysis

4.1 Main Results

Table 3 presents the performance comparison between the baseline methods and our proposed SampleMix across downstream tasks and perplexity evaluations. We draw the following key observations: (1) SampleMix achieves the highest average accuracy (47.77%) across the eight downstream tasks, outperforming all baseline methods. Specifically, it leads in 5 out of 8 tasks, demonstrating its efficacy in enhancing performance. (2) In perplexity evaluations, SampleMix records the lowest

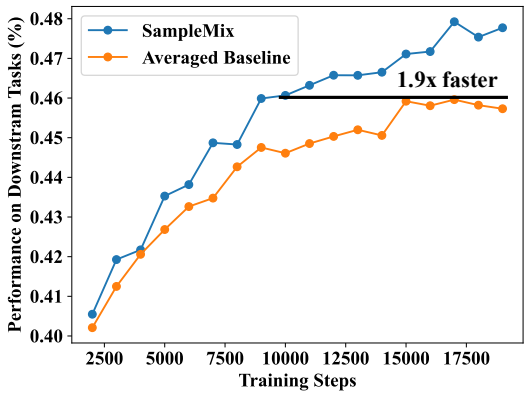


Figure 4: Training efficiency comparison. SampleMix reaches the averaged baseline at 100k training steps.

perplexity scores on both the Pile (25.63) and xP3 (46.38) datasets, underscoring the advantage in language modeling tasks.

Training Efficiency We compare the convergence speed of SampleMix with baselines. SampleMix achieves the baselines' accuracy using 1.4x to 2.1x fewer training steps. As illustrated in Figure 4, it attains the average baseline accuracy within 100k steps—1.9x faster, demonstrating the efficiency gains provided by our approach. The full comparison is shown in Figure 12.

Generalization to larger models Furthermore, to assess the effectiveness on larger models, we trained 8B models using the top 3 performing baselines and SampleMix (training setup detailed in Table 10). As Table 4 shows, SampleMix significantly outperforms the baselines, maintaining consistent advantages observed with 1B models.

Model	Average Performance
Vanilla	53.17
DoReMi	<u>53.58</u>
CE	53.15
SampleMix	**54.86**

Table 4: Performance comparison with 8B models.

Generalization to other datasets. We also compare SampleMix with the strong Vanilla baseline by training with the Chinese Wanjuan dataset (He et al., 2023). The results, detailed in Table 11, demonstrate consistent improvements across popular Chinese benchmarks (50.67 v.s. 43.32), underscoring the robust generalization of SampleMix.

4.2 Effectiveness of Quality and Diversity

To further explore the effectiveness of our quality and diversity evaluation, we conducted a comprehensive analysis by systematically varying the weighting factor α from 0.0 to 1.0. The corresponding model performances on downstream tasks are shown in Figure 5. From the results, we can observe the following two findings: **(1) Importance of Diversity** Setting α to 0.0 directly excludes the diversity measure, relying solely on quality. This configuration yields the lowest accuracy of 45.53%. As α increases from 0.0 to 0.8, there is a steady improvement in accuracy, peaking at 47.77%. This trend highlights the crucial role of diversity in achieving balanced data mixing and comprehensive data coverage. **(2) Necessity of Quality** When α is set to 1.0, diversity is fully weighted, and quality is excluded, leading to a slight decrease in accuracy to 47.58%. This minor drop indicates that while diversity is essential, incorporating the quality measure can further enhance performance.

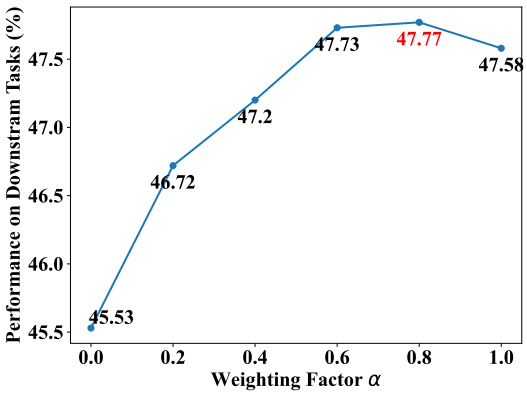


Figure 5: Average performance of downstream tasks with different weighting factor α.

It is important to note that the lowest performance at $\alpha = 0.0$, together with the optimal result at $\alpha = 0.8$ (favoring diversity), might suggest that the quality evaluation metric is less effective. We offer two explanations for this phenomenon: (1) The lowest performance at $\alpha = 0.0$ is primarily due to sampling bias towards certain data types (e.g., ArXiv and Books, as shown in Figure 3), thereby compromising the diversity and coverage of the training set. To directly assess the impact of quality, we conducted an additional experiment: training samples with quality scores below 4 were removed, while preserving the original mixture composition to maintain coverage. As Table 5

shows, this modification led to improved performance compared to the vanilla setting, directly affirming the value of the quality measure. (2) The SlimPajama dataset has already undergone rigorous quality filtering based on RedPajama standards, reducing the necessity for heavy quality weighting within the SampleMix framework. Notably, experiments on datasets with lower inherent quality (e.g., the Chinese WanJuan dataset (He et al., 2023)) reveal that prioritizing quality (e.g., $\alpha = 0.2$) leads to significantly better performance. This further validates both the adaptability of our framework and the importance of quality metrics.

Model	Average Performance
Vanilla	46.13
-w/o low-quality samples	47.27

Table 5: Performance comparison of removing low-quality samples.

User Recommendation The optimal value of α can vary based on characteristics of a dataset. Generally, for datasets of lower inherent quality, prioritizing quality by selecting a smaller α often leads to improved performance. We recommend users determine the optimal α for their particular dataset by following these steps: (1) assess the overall quality of your dataset; (2) select an appropriate range of hyperparameters based on this assessment; and (3) conduct grid search experiments using smaller models to efficiently identify the optimal value.

4.3 Adaptation to Varying Token Budget

Model development typically involves multiple training stages—such as pretraining, annealing, and continual pretraining—each requiring different token budgets. However, most existing methods present fixed data proportions, which limits their ability to accommodate varying token budget constraints effectively. To evaluate the benefits of dynamically adapting to different token budgets, we scale the SlimPajama dataset to $\frac{1}{5}$ of its original size, resulting in a smaller source dataset ($\approx$ 100B tokens). With the reduced source dataset, we adjusted the token budget proportion from $T_{\text{tgt}} = \frac{1}{5}T_{\text{src}}$ to $T_{\text{tgt}} = T_{\text{src}}$ (while maintaining $T_{\text{tgt}} = 100B$). We then conduct experiments under this adjusted token budget using the same setup. As Table 6 shows, we can observe that: (1) Baseline methods exhibit inconsistent performance when the token budget changes. For instance, DoReMi, the best-performing baseline in previous experiments, underperforms Vanilla and CE. This inconsistency indicates that baseline methods struggle to adapt effectively to different token budgets. (2) SampleMix achieves the highest average accuracy (47.46%), demonstrating SampleMix's ability to effectively adapt to varying token budgets. Detailed analysis in Appendix P shows that SampleMix can effectively utilize the sampling space and construct optimal training data.

Model	Average Performance
Vanilla	46.65
DoReMi	46.25
CE	46.40
BiMiX-OPT	45.54
DoGE	45.01
DML	44.96
SampleMix	**47.46**

Table 6: Performance comparison of different data mixture methods with 100B data as candidate pool.

4.4 Analysis of Computational Cost

We conducted a thorough examination of the cost breakdown for SampleMix and its baselines, as detailed in Appendix R. SampleMix incurs computational costs from three main components: quality evaluation, diversity evaluation, and hyperparameter tuning. These contribute to a total computational cost of $\mathbf{1.29 \times 10^{20}}$ FLOPs. SampleMix proves to be **more efficient than most existing data mixing methods**, accounting for only 2.68% of the cost required to train an 8B-parameter model on 100B tokens. Additionally, we introduce **two cost optimization strategies** for quality and diversity evaluation that we found beneficial in our practice, as described in Appendix S.

5 Related Work

We have covered research on data mixture in § 1, related work related to our technical designs is mainly introduced in the following.

Data Quality Heuristic rules, such as thresholds on word repetitions and perplexity, are commonly used to filter out low-quality data (Yuan et al., 2021; Dodge et al., 2021; Laurençon et al., 2022) . Earlier model-based methods employ binary classifiers to distinguish high-quality from low-quality data (Brown et al., 2020). Recent approaches incorporated more sophisticated models. Wettig et al. (2024) investigated four qualities-writing style, re-

quired expertise, facts & trivia, and educational value respectively. However, most methods rely on relatively coarse criteria and do not fully leverage the multi-dimensional property of data quality.

Diversity Traditional deduplication methods struggle to capture more complex semantic similarities (Wenzek et al., 2020; Soldaini et al., 2024). To better handle semantic redundancy, Abbas et al. (2023) applies K-Means clustering in the embedding space to identify and remove redundant data. Tirumala et al. (2023) builds on this approach by using SemDeDup as a preprocessing step before applying SSL Prototypes (Sorscher et al., 2022). Shao et al. (2024b) balances common and rare samples and ensures diversity by data clustering.

6 Conclusion

We have presented SampleMix, a sample-wise pre-training data mixing strategy by coordinating data quality and diversity. Extensive experiments demonstrate that SampleMix outperforms existing domain-wise methods, achieving comparable accuracy with 1.9x fewer training steps. In the future, we are interested in incorporating automatic evaluation metrics derived from the model's perspective to complement the current manually designed measures, and exploring code data mixing.

7 Limitations

In this study, we conducted experiments mainly using the SlimPajama dataset and identified the optimal hyperparameters specific to this dataset. While SampleMix is designed as a universal method applicable to various datasets, we acknowledge that optimal hyperparameters may vary across different datasets, which is consistent with existing works that require dataset-specific parameter tuning (Fan et al., 2023; Xie et al., 2024; Liu et al., 2025). Users aiming to apply our methodology to their own datasets will need to perform hyperparameter tuning to achieve optimal performance. We provide clear usage recommendations of three steps for hyperparameter tuning in § 4.2 (see **User Recommendation**). Specifically, we suggest assigning a smaller α to prioritize data quality in lower-quality datasets, thereby minimizing the influence of subpar data. Conversely, for higher-quality datasets, a larger α is recommended to ensure comprehensive data coverage through increased diversity.

References

Amro Abbas, Evgenia Rusak, Kushal Tirumala, Wieland Brendel, Kamalika Chaudhuri, and Ari S Morcos. 2024. Effective pruning of web-scale datasets based on complexity of concept clusters. *arXiv preprint arXiv:2401.04578*.

Amro Abbas, Kushal Tirumala, Dániel Simig, Surya Ganguli, and Ari S Morcos. 2023. Semdedup: Data-efficient learning at web-scale through semantic deduplication. *arXiv preprint arXiv:2303.09540*.

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*.

Yonatan Bisk, Rowan Zellers, Jianfeng Gao, Yejin Choi, et al. 2020. Piqa: Reasoning about physical commonsense in natural language. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 7432–7439.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Mayee F Chen, Michael Y. Hu, Nicholas Lourie, Kyunghyun Cho, and Christopher Re. 2025. Aioli: A unified optimization framework for language model data mixing. In *The Thirteenth International Conference on Learning Representations*.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. 2023. Palm: Scaling language modeling with pathways. *Journal of Machine Learning Research*, 24(240):1–113.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, et al. 2024. Scaling instruction-finetuned language models. *Journal of Machine Learning Research*, 25(70):1–53.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*.

Jesse Dodge, Maarten Sap, Ana Marasović, William Agnew, Gabriel Ilharco, Dirk Groeneveld, Margaret Mitchell, and Matt Gardner. 2021. Documenting large webtext corpora: A case study on the colossal clean crawled corpus. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 1286–1305.

Nan Du, Yanping Huang, Andrew M Dai, Simon Tong, Dmitry Lepikhin, Yuanzhong Xu, Maxim Krikun, Yanqi Zhou, Adams Wei Yu, Orhan Firat, et al. 2022. Glam: Efficient scaling of language models with mixture-of-experts. In *International Conference on Machine Learning*, pages 5547–5569. PMLR.

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

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

Leo Gao, Stella Biderman, Sid Black, Laurence Golding, Travis Hoppe, Charles Foster, Jason Phang, Horace He, Anish Thite, Noa Nabeshima, et al. 2020. The pile: An 800gb dataset of diverse text for language modeling. *arXiv preprint arXiv:2101.00027*.

Ce Ge, Zhijian Ma, Daoyuan Chen, Yaliang Li, and Bolin Ding. 2024. Data mixing made efficient: A bivariate scaling law for language model pretraining. *arXiv preprint arXiv:2405.14908*.

Sachin Goyal, Pratyush Maini, Zachary C Lipton, Aditi Raghunathan, and J Zico Kolter. 2024. Scaling laws for data filtering–data curation cannot be compute agnostic. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 22702–22711.

Suriya Gunasekar, Yi Zhang, Jyoti Aneja, Caio César Teodoro Mendes, Allie Del Giorno, Sivakanth Gopi, Mojan Javaheripi, Piero Kauffmann, Gustavo de Rosa, Olli Saarikivi, et al. 2023. Textbooks are all you need. *arXiv preprint arXiv:2306.11644*.

Conghui He, Zhenjiang Jin, Chao Xu, Jiantao Qiu, Bin Wang, Wei Li, Hang Yan, Jiaqi Wang, and Dahua Lin. 2023. Wanjuan: A comprehensive multimodal dataset for advancing english and chinese large models. *arXiv preprint arXiv:2308.10755*.

Binyuan Hui, Jian Yang, Zeyu Cui, Jiaxi Yang, Dayiheng Liu, Lei Zhang, Tianyu Liu, Jiajun Zhang, Bowen Yu, Keming Lu, et al. 2024. Qwen2. 5-coder technical report. *arXiv preprint arXiv:2409.12186*.

Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2019. Billion-scale similarity search with gpus. *IEEE Transactions on Big Data*, 7(3):535–547.

Hugo Laurençon, Lucile Saulnier, Thomas Wang, Christopher Akiki, Albert Villanova del Moral, Teven Le Scao, Leandro Von Werra, Chenghao Mou, Eduardo González Ponferrada, Huu Nguyen, et al. 2022. The bigscience roots corpus: A 1.6 tb composite multilingual dataset. *Advances in Neural Information Processing Systems*, 35:31809–31826.

Yuanzhi Li, Sébastien Bubeck, Ronen Eldan, Allie Del Giorno, Suriya Gunasekar, and Yin Tat Lee. 2023. Textbooks are all you need ii: phi-1.5 technical report. *arXiv preprint arXiv:2309.05463*.

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

Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. 2018. Can a suit of armor conduct electricity? a new dataset for open book question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2381–2391.

Niklas Muennighoff, Thomas Wang, Lintang Sutawika, Adam Roberts, Stella Biderman, Teven Le Scao, M Saiful Bari, Sheng Shen, Zheng-Xin Yong, Hailey Schoelkopf, et al. 2022. Crosslingual generalization through multitask finetuning. *arXiv preprint arXiv:2211.01786*.

Zhenxing Niu, Mo Zhou, Le Wang, Xinbo Gao, and Gang Hua. 2016. Ordinal regression with multiple output cnn for age estimation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4920–4928.

Denis Paperno, Germán Kruszewski, Angeliki Lazaridou, Ngoc-Quan Pham, Raffaella Bernardi, Sandro Pezzelle, Marco Baroni, Gemma Boleda, and Raquel Fernández. 2016. The lambada dataset: Word prediction requiring a broad discourse context. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1525–1534.

Noveen Sachdeva, Benjamin Coleman, Wang-Cheng Kang, Jianmo Ni, Lichan Hong, Ed H Chi, James Caverlee, Julian McAuley, and Derek Zhiyuan Cheng. 2024. How to train data-efficient llms. *arXiv preprint arXiv:2402.09668*.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2021. Winogrande: An adversarial winograd schema challenge at scale. *Communications of the ACM*, 64(9):99–106.

Yunfan Shao, Linyang Li, Zhaoye Fei, Hang Yan, Dahua Lin, and Xipeng Qiu. 2024a. Balanced data sampling for language model training with clustering. *arXiv preprint arXiv:2402.14526*.

Yunfan Shao, Linyang Li, Zhaoye Fei, Hang Yan, Dahua Lin, and Xipeng Qiu. 2024b. Balanced data sampling for language model training with clustering. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 14012–14023, Bangkok, Thailand. Association for Computational Linguistics.

Daria Soboleva, Faisal Al-Khateeb, Robert Myers, Jacob R Steeves, Joel Hestness, and Nolan Dey. 2023.

SlimPajama: A 627B token cleaned and deduplicated version of RedPajama.

Luca Soldaini, Rodney Kinney, Akshita Bhagia, Dustin Schwenk, David Atkinson, Russell Authur, Ben Bogin, Khyathi Chandu, Jennifer Dumas, Yanai Elazar, Valentin Hofmann, Ananya Jha, Sachin Kumar, Li Lucy, Xinxi Lyu, Nathan Lambert, Ian Magnusson, Jacob Morrison, Niklas Muennighoff, Aakanksha Naik, Crystal Nam, Matthew Peters, Abhilasha Ravichander, Kyle Richardson, Zejiang Shen, Emma Strubell, Nishant Subramani, Oyvind Tafjord, Evan Walsh, Luke Zettlemoyer, Noah Smith, Hannaneh Hajishirzi, Iz Beltagy, Dirk Groeneveld, Jesse Dodge, and Kyle Lo. 2024. Dolma: an open corpus of three trillion tokens for language model pretraining research. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15725–15788, Bangkok, Thailand. Association for Computational Linguistics.

Ben Sorscher, Robert Geirhos, Shashank Shekhar, Surya Ganguli, and Ari Morcos. 2022. Beyond neural scaling laws: beating power law scaling via data pruning. *Advances in Neural Information Processing Systems*, 35:19523–19536.

Kushal Tirumala, Daniel Simig, Armen Aghajanyan, and Ari Morcos. 2023. D4: Improving llm pretraining via document de-duplication and diversification. *Advances in Neural Information Processing Systems*, 36:53983–53995.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2019. Superglue: A stickier benchmark for general-purpose language understanding systems. *Advances in neural information processing systems*, 32.

Guillaume Wenzek, Marie-Anne Lachaux, Alexis Conneau, Vishrav Chaudhary, Francisco Guzmán, Armand Joulin, and Édouard Grave. 2020. Ccnet: Extracting high quality monolingual datasets from web crawl data. In *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 4003–4012.

Alexander Wettig, Aatmik Gupta, Saumya Malik, and Danqi Chen. 2024. Qurating: Selecting high-quality data for training language models. *arXiv preprint arXiv:2402.09739*.

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.

Sang Michael Xie, Shibani Santurkar, Tengyu Ma, and Percy S Liang. 2023. Data selection for language models via importance resampling. *Advances in Neural Information Processing Systems*, 36:34201–34227.

Jiasheng Ye, Peiju Liu, Tianxiang Sun, Jun Zhan, Yunhua Zhou, and Xipeng Qiu. 2025. Data mixing laws: Optimizing data mixtures by predicting language modeling performance. In *The Thirteenth International Conference on Learning Representations*.

Sha Yuan, Hanyu Zhao, Zhengxiao Du, Ming Ding, Xiao Liu, Yukuo Cen, Xu Zou, Zhilin Yang, and Jie Tang. 2021. Wudaocorpora: A super large-scale chinese corpora for pre-training language models. *AI Open*, 2:65–68.

Xin Zhang, Yanzhao Zhang, Dingkun Long, Wen Xie, Ziqi Dai, Jialong Tang, Huan Lin, Baosong Yang, Pengjun Xie, Fei Huang, et al. 2024. Mgte: generalized long-context text representation and reranking models for multilingual text retrieval. *arXiv preprint arXiv:2407.19669*.

Qihao Zhu, Daya Guo, Zhihong Shao, Dejian Yang, Peiyi Wang, Runxin Xu, Y Wu, Yukun Li, Huazuo Gao, Shirong Ma, et al. 2024. Deepseek-coder-v2: Breaking the barrier of closed-source models in code intelligence. *arXiv preprint arXiv:2406.11931*.

A Domain Overlaps

We manually check the samples within the same cluster but from different domains. Such samples are usually topic-relevant and similar in terms of structure, semantics, and context. As Figure 6 shows, the samples all discuss topics about Einstein and the Theory of Relativity.

B Samples from Slimpajama CommonCarwl

We manually check the low-quality and high-quality samples from Slimpajama CommonCarwl. As Figure 7 shows, the data quality of CommonCrawl varies significantly. The low-quality sample is characterized by fragmented and disorganized information, primarily consisting of sporadic headlines and links related to sports news. On the other hand, the high-quality sample provides a coherent and informative excerpt about astrophysical research, demonstrating a clear and structured narrative.

C Quality Evaluation Prompt

The prompt for GPT-4o to assess training data quality is given in Figure 8.

D Cost of GPT-4o

Our usage of GPT-4o aligns with standards seen in related studies (Wettig et al., 2024). For instance, the average input length is 2214 tokens, accounting for a cost calculation of 2214 tokens × 420k documents × 1.25/M = $1162. The average output length is 339 tokens and the cost for output is 339 * 420 * 5/1M=$711. The aggregate cost sums to $1873, notably less than the $2820 reported by QuRating (Wettig et al., 2024). Using AWS p4d as a reference, the cost of training an 8-billion-parameter model with 500 billion tokens is approximately 250k dollars to 300k dollars, taking into account GPU, storage, and other expenses. Thus, the expenses associated with GPT-4o are minimal in comparison to the costs of training large language models and do not substantially impact the overall cost of SampleMix. Furthermore, to support ongoing research efforts, we will make both the GPT-4o-generated training data and the annotated SlimPajama dataset publicly available.

E Code of Quality Evaluator

Table 7 shows the Python code for implementing the ordinal regression model aimed at quality scoring tasks, including model definition, loss function computation, and inference process. The full code can be found in the supplementary materials.

The `OrdinalRegressionModel` class initializes the pre-trained base model and a series of ordinal layers. Each ordinal layer outputs the probability that the quality score is greater than a specific threshold. For instance, the first ordinal layer (index 0) computes the probability that the quality score is greater than 0, i.e., the probability that the score is at least 1. Similarly, the second ordinal layer (index 1) calculates the probability that the quality score is greater than 1, meaning the probability that the score is at least 2, and so on. The last ordinal layer (index 9) computes the probability that the score is greater than 9, which is equivalent to the probability that the score is exactly 10. Therefore, the model has 10 ordinal layers in total, each corresponding to one of these thresholds.

The `loss` function calculates the ordinal loss by summing the binary cross-entropy loss between the predicted probabilities and the target values. For each ordinal layer, a binary target is created, indicating whether the true score is greater than the threshold corresponding to that layer. Specifically, the larger the deviation between the predicted score and the true score, the higher the loss, which helps the model focus on reducing these deviations during training.

The `predict` function implements inference using the trained ordinal regression model. It first computes the predicted probabilities for each class, and then calculates the final predicted score by selecting the class with the maximum probability. The function also calculates the probability distribution across all possible scores, which provides a measure of confidence for the predicted score.

F Cases of High/Low Quality

Figure 9 shows cases of high/low quality. Our quality evaluator can effectively distinguish between high-quality samples (scientific literature, knowledge reports, etc.) and low-quality samples (advertisements, incomplete web pages, etc.)

Samples from Different Data Sources with Similar Topics

Arxiv
General relativity suffers from a number of problems regarding its local conservation laws for energy and momentum. This was the subject of a crucial discussion between Hilbert, Klein, Noether, and Einstein between 1915 and 1918....

C4
The term mc^2 had already made an appearance in his paper of 26 September, which introduced special relativity. The paper of 21 November showed that $E = mc^2$ applies to bodies at rest. [Physics Today]...

CommonCrawl
The General Theory of Relativity (GRT) was born among other things from the demand to be able to use arbitrary coordinate systems for the description of the laws of nature. According to the covariance principle, the form of the laws of nature should not depend decisively on the choice of the special coordinate system...

StackExchange
No one but Einstein can be sure of exactly how he arrived at GR. From reading various histories of the time it seems to me that once Einstein had come up with the equivalence principle he started looking around for theories that embodied it...

Wikipedia
The Meaning of Relativity: Four Lectures Delivered at Princeton University, May 1921 is a book published by Princeton University Press in 1922 that compiled the 1921 Stafford Little Lectures at Princeton University, given by Albert Einstein...

Figure 6: Samples from different domains, all describing information related to Einstein and Theory of Relativity.

G K-means Clustering Details

For the data clustering in § 2.3, we generate 768-dimensional embeddings for each sample [†]. Further, we normalize the embeddings to have L2-norm of 1.0, and use faiss (Johnson et al., 2019) to perform K-means clustering. Following Tirumala et al. (2023); Abbas et al. (2024), we set the number of clusters to be the square root of the number of total points being clustered. The core code of data clustering is presented in Table 8. The full code can be found in the supplementary materials.

H Key Distinctions of Diversity Evaluation

Technical designs of the diversity evaluator are inspired by (Abbas et al., 2024; Shao et al., 2024a). However, we point out the key distinctions as follows:

- Shao et al. (2024a) employs data clustering but selects data uniformly from clusters, overlooking distinct cluster characteristics. We also perform clusters but mainly to extract diversity features for further sampling.

- Abbas et al. (2024) focuses on data pruning, proposing a three-stage pipeline that includes deduplication, CLIP-score filtering, and density-based pruning, and is validated on a dataset without domain boundaries. Conversely, we address the challenges associated with pretraining data mixtures across multiple domains, incorporating diversity as one feature within our framework. Further, our analysis reveals that the average diversity of text data is typically lower than that of image data, as illustrated in Figure 3 (b). To effectively model diversity within text data, we use a smaller number of clusters, specifically $\sqrt{N}$, where N is the number of documents, along with a higher number of nearest neighbors (1% of cluster numbers).

I Cases of High/Low Diversity

As Figure 10 shows, our diversity evaluation assigns greater weight to distinctive content, such as academic reports, in-depth analyses, and rare knowledge. In contrast, it gives lower weight to content that is overly common or widely circulated online, including advertisements, duplicate product manuals, and routine sports score updates. This approach ensures that unique and valuable information is prioritized, while redundant or ubiquitous content is given less emphasis.

J Statistics of Diversity Evaluation

We present the distribution of cluster size (Figure 11a), $d_{\text{compactness}}$ (Figure 11b), $d_{\text{separation}}$ (Figure 11c).

[†]https://huggingface.co/princeton-nlp/unsup-simcse-bert-base-uncased

Low-Quality and High-Quality Samples

Low-Quality Sample
New posts Featured Search forums
Sports Briefing (New York Times)
Thread starter articlebot
Cycling News Headlines
articlebot
auto racing.
http://us.rd.yahoo.com/dailynews/rss/search/cycling+racing/SIG=120pnaegk/*http
Cycling News Headlines Jul 31, 2007
Cycling News Headlines Jun 2, 2007
Cycling News Headlines May 11, 2007
Cycling News Headlines Mar 17, 2007
Cycling News Headlines Dec 20, 2006
Sports Briefing: Basketball, Cycling, Auto Racing, Hockey, Golf, Football and Soccer (New York Times
Cycling News Headlines Nov 26, 2006
Cycling News Headlines Oct 17, 2006
Cycling News Headlines Sep 21, 2006
Sports Briefing: Track and Field, Marathon, Auto Racing, College Football and Cycling (New York Time
Cycling News Headlines Aug 28, 2006
Sports Briefing: Baseball, Golf, Horse Racing and Cycling (New York Times)
Cycling News Headlines Jun 26, 2006

High-Quality Sample
Decades of studies show that most massive galaxies harbor a supermassive black hole at their center, with the mass of the black hole being one tenth of the total mass of the surrounding spheroid of stars. Two astrophysicists from the Center for Astrophysics | Harvard and the Smithsonian have proposed a method to observe what could be the second-closest supermassive black hole to Earth.

Figure 7: Quality of CommonCrawl Samples may vary significantly.

K Rationality For Re-Normalization

Our re-normalization approach follows the methodology established by RegMix (Liu et al., 2025), which similarly rescaled weights from other methods when applied to the Pile dataset. The rationality for this re-normalization stems from two key observations:

- The GitHub domain exhibits fundamentally different characteristics from textual domains, showing minimal mutual influence in terms of both content and structure (Ye et al., 2025).
- This aligns with established practices in the field, where code is typically treated as a distinct language modality (Dubey et al., 2024; Zhu et al., 2024; Hui et al., 2024)

Given these considerations, we (1) directly adopted the baseline weights from the original papers; (2) applied re-normalization to account for the GitHub domain exclusion. This approach maintains methodological consistency while appropriately addressing the unique nature of code as a data domain.

L Hyper-Parameters of Training Models

The experiments for both 1B and 8B parameter models follow standard transformer architecture with carefully optimized hyper-parameters. Table 9 and Table 10 introduce the architectural configurations and training specifications for both model scales respectively.

M Coverage Speed of All Methods

Figure 12 shows the full comparison of SampleMix and all baselines. SampleMix achieves the baselines' accuracy using 1.4x to 2.1x fewer training steps.

N Domain Weights of Different Methods

Figure 13 shows the domain weights of different methods.

O Generalization to other datasets

To evaluate the generalization ability of SampleMix, we conducted an additional experiment using the Chinese Wanjuan dataset (He et al., 2023). We compared SampleMix to the vanilla mixture method. The results, detailed in Table 11, demonstrate consistent improvements across popular Chinese benchmarks (50.67 v.s. 43.32), underscoring the robust generalization capability of SampleMix.

P Analysis of Varying Token Budgets

To further investigate how SampleMix adapts to varying token budgets, we analyze the sampling counts under different scenarios. Figure 14a illustrates the proportion of various sampling counts, while Figure 14b presents the average sampling weights $p(x)$ associated with these counts. We can observe that: For $T_{\text{tgt}} = \frac{1}{5}T_{\text{src}}$, the source dataset is sufficiently large, allowing top-tier data to meet the token budget. SampleMix precisely selects high-weight samples to fulfill the budget requirements, minimizing the need for extensive upsampling (i.e., sampling count > 1 is rare) and ensuring that all valuable data is included. For $T_{\text{tgt}} = T_{\text{src}}$, the source dataset is relatively smaller, and high-weight samples alone are insufficient to meet the token budget. To satisfy the budget, SampleMix incorporates lower-weight samples. Despite this inclusion, the method effectively identifies and discards the least valuable data, which accounts for 18.245% of the dataset due to their low sampling weights (average weight = 0.166). Data with higher sampling weights are upsampled more frequently, thereby enhancing their representation within the constrained budget. Additionally, for $T_{\text{tgt}} = \frac{1}{5}T_{\text{src}}$, the average sampling weight is larger (0.312 v.s. 0.289 when $T_{\text{tgt}} = T_{\text{src}}$), further verifying SampleMix's ability to effectively utilize the sampling space and adapt to varying token budgets.

Q Analysis of Sampling Count Distribution

Figure 15a presents the distribution of sampling counts for each domain. Although our target training budget T_{tgt} is approximately equal to the size of the candidate pool T_{src}, our method strategically discards documents with the lowest quality and diversity by assigning them a sampling count of zero. This approach contrasts with traditional methods that utilize uniform sampling across all documents. In Figure 15b, we display the sampling weights corresponding to the sampling counts. The results demonstrate that our method allocates higher sampling counts to samplers with larger sampling weights, aligning with our expectations. Additionally, the distribution of sampling counts exhibits significant variation across different domains. This variability underscores our method's effectiveness in capturing both fine-grained variations and commonalities among diverse domains, ensuring a more nuanced and efficient sampling process.

R Analysis of Computatioanl Cost

R.1 Cost Breakdown of SampleMix

Our method incurs computational costs from 3 main components:

(1) Quality Evaluation Our approach involves training a model with 137 million parameters on 410,000 data points and annotating a training set of 100 billion with the trained quality evaluator. The computational costs are detailed as follows: (1) Training evaluator: The cost is calculated as $6 \times 137M \times 410k \times 1024 = 3.45 \times 10^{17}$ FLOPs; (2) Dataset annotation: Annotating the dataset requires $2 \times 137M \times 100B = 2.74 \times 10^{19}$ FLOPs. (3) Subtotal for quality evaluation: The total cost amounts to 2.77×10^{19} FLOPs.

(2) Diversity Evaluation For diversity evaluation, we extract embeddings from 10^8 documents using a model with 108M parameters and an input length of 512. This process incurs a computational cost of $2 \times 108M \times 10^8 \times 512 = 1.12 \times 10^{19}$ FLOPs.

(3) Hyperparameter Tuning We provide clear usage recommendations in 4.2 to guide users through an efficient hyperparameter tuning process:

- Dataset Quality Evaluation: Assess the quality of your dataset before tuning.
- Hyperparameter Range Selection: Choose appropriate hyperparameter ranges based on the quality evaluation.

- Grid Search with Small Models: Perform initial tuning experiments using smaller models.

While our original experiments involved six grid searches across the full range [0,1], we recommend three targeted grid searches within a refined range (determined by quality evaluation). Following the common practice of using data experiments from smaller models as a foundation for larger models (Goyal et al., 2024; Chung et al., 2024; Bi et al., 2024), we recommend a practical scenario widely used in our practice (100M-parameter models trained on 50B tokens with 3 tuning experiments), the cost is: $6 \times 100\text{M} \times 50\text{B} \times 3 = 9 \times 10^{19}$ FLOPs.

Summing up all components, the total computational cost of SampleMix is $\mathbf{1.29 \times 10^{20}}$ **FLOPs**.

R.2 Cost Comparison with Baselines

We compare the cost with the baselines as follows:

- DoreMi, trains 280M proxy and reference model with 104B tokens, resulting in $6 \times 280M \times 104B \times 2 = 3.49 \times 10^{20}$ FLOPS.
- Doge, trains 82M proxy model with 104B, resulting in $6 \times 82M \times 104B = 5.12 \times 10^{19}$ FLOPS.
- Bimix, trains 280M proxy-model with 100B, the computational cost is $6 \times 280M \times 100B = 1.68 \times 10^{20}$FLOPS
- DML, trains a series of 70M, 160M, 305M and 410M proxy models on 30B tokens, resulting in $6 \times (70M + 160M + 305M + 410M) \times 30B = 1.7 \times 10^{20}$ FLOPS

SampleMix is more efficient than most of the existing data mixing methods and only represents 2.68% of the cost to train an 8B-parameter model on 100B tokens (4.8×10^{21} FLOPs). Even if this additional compute were allocated to extended training, SampleMix still achieves superior performance.

S Cost Optimization Strategies

In our application of SampleMix to training LLMs (over 1T parameters models and 15T tokens), we explored methods to reduce computational overhead in quality evaluation and data clustering. Two key optimizations include:

- Diversity evaluation. Compute centroids using a document subset, then assign remaining documents to these centroids.
- Quality evaluation. Employ lightweight evaluators (e.g., smaller encoder models or FastText) for efficiency.

By integrating SampleMix with these strategies, our approach achieves high scalability for real-world applications, effectively balancing computational cost with model performance.

Quality Evaluation Template

Annotator Task: Text Data Quality Evaluation
Role: You are a Language Model Training Data Annotator. Your job is to evaluate the quality of text documents.
Objective: Assess each document using the seven evaluation dimensions below. For each dimension, assign a score based on the provided criteria to determine the document's quality.
Evaluation Dimensions:
1. Clarity of Expression and Accuracy (0-1 points)
- Evaluate: How clearly ideas are expressed and the correctness of language (grammar, syntax, punctuation).
- Score:
 - 0: Numerous grammatical, spelling, or punctuation errors that significantly hinder comprehension.
 - 1: Few grammatical or punctuation errors that do not impede understanding; ideas are clearly and smoothly expressed.
2. Completeness and Coherence (0-1 points)
- Evaluate: Whether paragraphs are fully developed, relevant to the main theme, and logically connected.
- Score:
 - 0: Underdeveloped or off-topic paragraphs; lack of logical flow causing confusion.
 - 1: Well-developed, relevant paragraphs that are logically connected and contribute to a unified theme.
3. Structure and Style (0-1 points)
- Evaluate: The overall logical flow of the document and the clarity of the author's presentation.
- Score:
 - 0: Unclear structure and inconsistent or unengaging style.
 - 1: Clear and logical structure with a consistent and appropriate style that facilitates understanding.
4. Content Accuracy and Credibility (0-1 points)
- Evaluate: Appropriateness of content (free from pornography, drugs, violence) and the accuracy and reliability of facts and sources.
- Score:
 - 0: Inappropriate material or contains factual inaccuracies and unreliable sources.
 - 1: Appropriate content, free from prohibited material, with accurate and credible information.
5. Significance (0-1 points)
- Evaluate: The importance, originality, and broader impact of the document compared to others in the field. Verify that the document is not machine-generated.
- Score:
 - 0: Lacks importance and originality. It does not provide unique insights or contribute meaningfully beyond its immediate purpose. It is not recognized as historically significant or exhibits characteristics of being machine-generated.
 - 1: Demonstrates originality and important or impactful. It demonstrates originality and offers unique insights or contributions. It may also hold historical significance or be recognized as influential.
6. Knowledge Richness (0-2 points)
- Evaluate: The depth and breadth of information, including comprehensive insights and detailed explanations that enhance the reader's understanding. Ensure that any concepts or jargon used are well-explained.
- Score:
 - 0: Minimal information with little to no depth or insights.
 - 1: Adequate information with some insightful explanations; concepts or jargon introduced but not thoroughly explained.
 - 2: Comprehensive and detailed information with deep insights; all concepts and jargon are clearly explained and accessible, offering strong educational value.
7. Logicality and Analytical Depth (0-3 points)
- Evaluate: The text's ability to present profound insights or viewpoints, supported by in-depth analysis and reasoning.
- Score:
 - 0: Contains only simple statements and basic facts without deeper exploration.
 - 1: Describes or analyzes straightforward issues or processes with limited depth.
 - 2: Offers detailed analysis or solutions, addressing complex professional issues with substantial depth.
 - 3: Building on the 2-point criteria, if the text involves STEM fields (Science, Technology, Engineering, Mathematics), such as astronomy, medicine, mathematics, physics, chemistry, biology, etc., an additional point is awarded for a total of 3 points, acknowledging the specialized complexity and depth required in these areas.
Requirements: Based on the above dimensions, score the text content, first stating the evaluation reasons, then providing the quality assessment score. The final score is the sum of all dimensions, ranging from 0-10 points. Output format is JSON:
{"Evaluation Reasons": "Clarity of Expression": "...", "Completeness and Coherence": "...", "Structure and Style": "...", "Appropriate Content and Credibility": "...", "Significance": "...", "Knowledge Richness": "...", "Logicality and Analytical Depth": "...", "Clarity of Expression": X, "Completeness and Coherence": X, "Structure and Style": X, "Appropriate Content and Credibility": X, "Significance": X, "Knowledge Richness and Educational Value": X, "Logicality and Analytical Depth": X, "Final Score": X}
Evaluate all the text as a whole:
«<Document»>

Figure 8: Prompt for GPT-4o to assess training data quality.

```python
# Define the ordinal regression model class
class OrdinalRegressionModel(torch.nn.Module):
    def __init__(self, pretrained_path, num_classes=10):
        super(OrdinalRegressionModel, self).__init__()
        self.base_model = AutoModel.from_pretrained(pretrained_path)
        self.ordinal_layers = torch.nn.ModuleList([torch.nn.Linear(
                              self.base_model.config.hidden_size, 1)
                              for _ in range(num_classes)])

    def forward(self, input_ids, attention_mask=None, token_type_ids=None):
        outputs = self.base_model(input_ids=input_ids,
                                  attention_mask=attention_mask,
                                  token_type_ids=token_type_ids)
        last_hidden_state = outputs.last_hidden_state
        cls_representation = last_hidden_state[:, 0, :]

        # Compute the output for each ordinal layer
        ordinal_outputs = [torch.sigmoid(layer(cls_representation))
                           for layer in self.ordinal_layers]
        ordinal_outputs = torch.cat(ordinal_outputs, dim=1)
        return ordinal_outputs

# Calculate the ordinal loss
def loss(outputs, targets):
    loss = 0.0
    for i in range(outputs.size(1)):
        binary_targets = (targets > i).float()
        loss += nn.functional.binary_cross_entropy(outputs[:, i], binary_targets)
    return loss

# Inference function
def predict(text):
    with torch.no_grad():
        inputs = tokenizer(
            text,
            truncation=True,
            padding=True,
            max_length=4096,
            return_tensors="pt"
        )
        # Get model outputs
        outputs = model(input_ids=inputs['input_ids'],
                        attention_mask=inputs['attention_mask'])

        # Initialize probability array
        probabilities = torch.zeros(outputs.size(0), outputs.size(1) + 1)
        # Calculate probability for the first class
        probabilities[:, 0] = 1 - outputs[:, 0]
        if outputs.size(1) > 1:
            # Calculate probabilities for the middle classes
            probabilities[:, 1:-1] = outputs[:, :-1] - outputs[:, 1:]
        # Calculate probability for the last class
        probabilities[:, -1] = outputs[:, -1]

        # Calculate scores by finding the index of the maximum probability
        scores = torch.argmax(probabilities, dim=1)
    return scores, probabilities
```

Table 7: Python Code for implementing the ordinal regression model.

High-Quality and Low-Quality Samples

High-Quality Sample (quality score $q(x) = 10$)
In floricultural crops, flower morphology, such as large petals and double flower formation, and flower longevity are important factors that influence their quality. Petunia has been proved to be an excellent model plant for the study of flower development and senescence. However, even in petunia, there are a lot of genes whose function in flower development and senescence have not yet been characterized. Recently, techniques using virus induced gene silencing (VIGS) have been developed as efficient reverse genetics tools to test gene function. In this study, VIGS system that visualizes silencing induced-flower was established in petunia. Using this system, functional characterization of petunia candidate genes involved in flower morphogenesis and senescence was conducted. In parallel, identification and expression analysis of flower development related-genes that had not yet been identified in petunia was performed. Disadvantage of VIGS is that silencing is induced in a chimeric manner and it is sometimes difficult to identify flowers on which silencing is induced.

Low-Diversity Sample (quality score $q(x) = 1$)
Tonight off Witney show Register Login Contact Us Billings Montana married women looking for men I Search Cock I Searching Sexual Encounters Waiting flr this asap. Do you love the feel of a tongue on your nips, if you like them caressed, played with, licked and sucked, i am waiting for your email. No games or pornography. Have best dayAmerican, asianmiddle Eastern, Persian. I was the boy with a shaved head and glboobieses; if you're ever feeling adventurous, hit this ad up with what auto part you were replacing in the subject

Figure 9: Cases of high/low quality.

```
# Calculate the number of clusters
n_centroids = int(math.sqrt(all_embeddings.shape[0]))
# define the parameters
kmeans = faiss.Kmeans(
    d = 768,
    k = n_centroids,
    niter=50, # 50 iterations
    gpu = True,
    seed = 1024,
    spherical = True,
    min_points_per_centroid=1,
    max_points_per_centroid=all_embeddings.shape[0]
)
# perform data clustering
kmeans.train(all_embeddings)
```

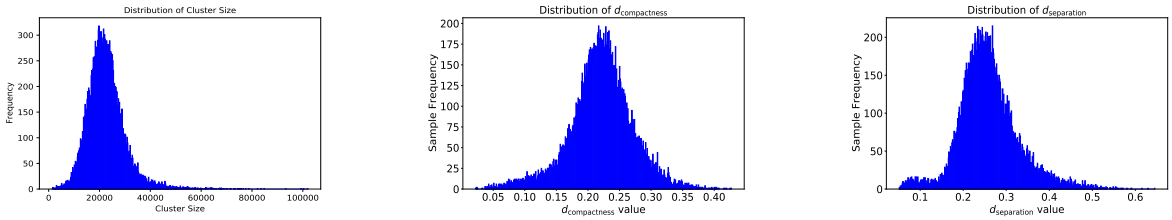
Table 8: Python Code for implementing K-Means clustering.

High-Diversity and Low-Diversity Samples

High-Diversity Sample (diversity score $d(x) = 0.7302$)
Spartan boys began their military training at age 7, and men served in the army until age 60. Loki's father was Fárbauti and his mother was Laufey. On the other hand, the goal of education in Athens, a democratic city-state, was to produce citizens trained in the arts of both peace and war. In Athens, boys received a well-rounded education, but girls were only taught household chores. In Sparta, both boys and girls received physical training to stay fit. Spartan boys received a military education and training for many years. The ultimate goal of the agoge, or the Spartan education system, was to raise male soldiers who would be effective in the Spartan army. Training began at the age of seven and all male citizens, except the firstborn male of the household, was required to attend this training. IT'S FUNNING: Best answer: What dangerous animals lived in ancient Greece? They learned basic things like reading, writing and math.

Low-Diversity Sample (diversity score $d(x) = 0.0685$)
External component identification Finding your hardware and software information Locating hardware Locating software Buttons and speakers Illustrated parts catalog Computer major components Display assembly subcomponents Mass storage devices Sequential part number listing Removal and replacement procedures preliminary requirements Service considerations Drive handling Grounding guidelines Electrostatic discharge damage Packaging and transporting guidelines Workstation guidelines Removal and replacement procedures for Customer Self-Repair parts Component replacement procedures Removal and replacement procedures for Authorized Service Provider parts Base enclosure WLAN module TouchPad button board Battery Board (select models only)

Figure 10: Cases of high/low diversity.



(a) Distribution of Cluster Size (b) Distribution of $d_{\text{compactness}}$ (c) Distribution of $d_{\text{separation}}$

Figure 11: Statistics of Diversity Evaluation.

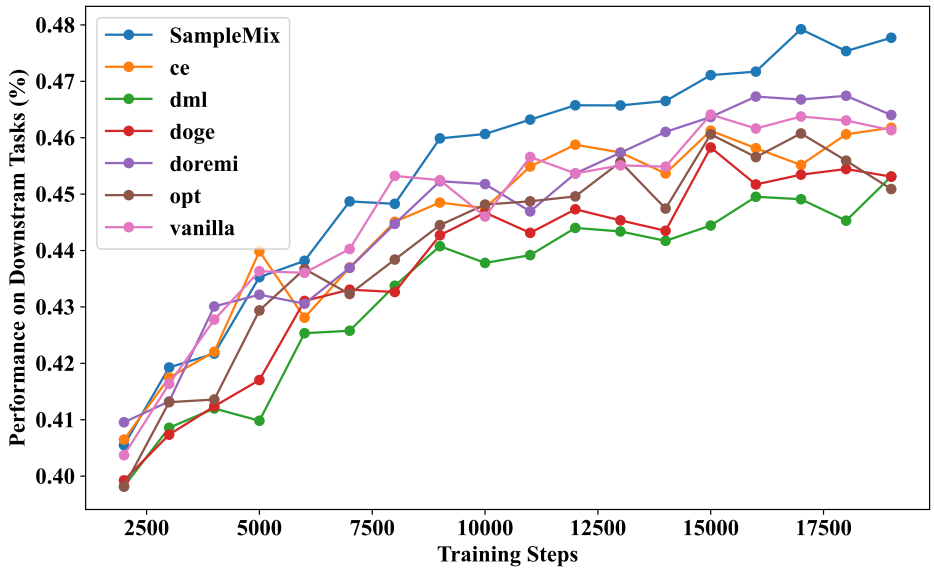


Figure 12: Coverage speed of all baselines and SampleMix. SampleMix achieves the best training efficiency.

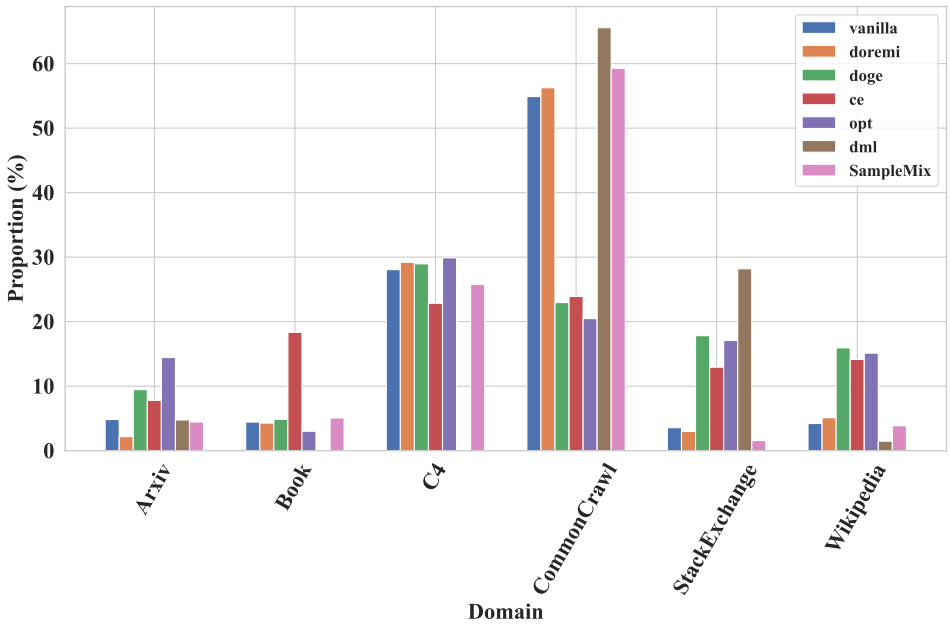


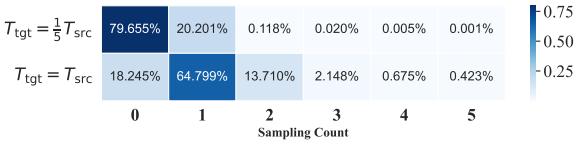
Figure 13: Domain weights of different methods.

Hyper-parameter	Value
layer num	28
attention head num	13
attention head dim	128
model dim	1664
ffn intermediate dim	4480
global batch size	1280
sequence length	4096
learning rate	$2e^{-4}$
learning rate scheduler	cosine scheduler
learning rate warmup tokens	525M

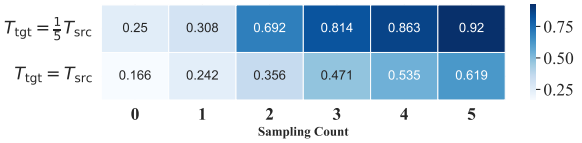
Table 9: Hyper-parameters of 1B models used in the experiment.

Hyper-parameter	Value
layer num	32
attention head num	32
attention head dim	128
model dim	4096
ffn intermediate dim	14336
global batch size	1280
sequence length	4096
learning rate	$2e^{-4}$
learning rate scheduler	cosine scheduler
learning rate warmup tokens	525M

Table 10: Hyper-parameters of 8B models used in the experiment.



(a) Proportion of different sampling counts.

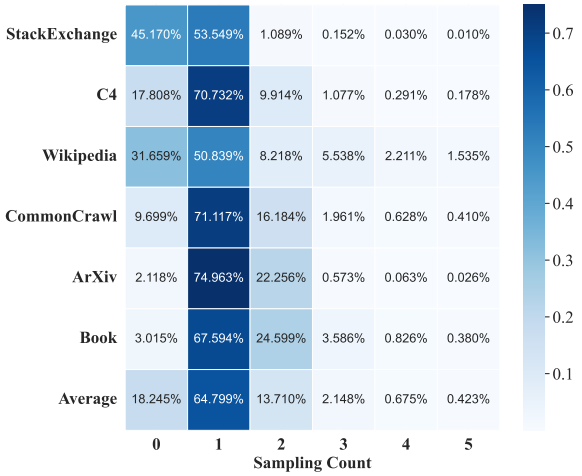


(b) Sampling weight (i.e., $p(x)$) of different sampling counts.

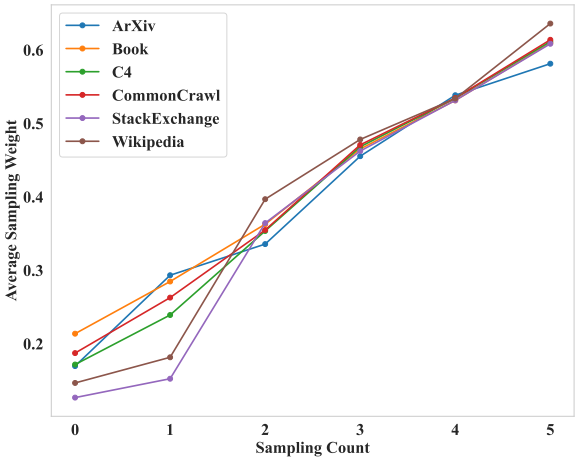
Figure 14: Analysis of different sampling counts.

Dataset	**Vanilla**	**SampleMix**
CMMLU	35.35	41.89
CEval	35.35	41.93
CSL	31.50	47.50
DRCD	62.83	72.86
Classical_Chinese_Translate	52.08	52.89
Idiom_Antonym	60.63	63.13
Logiqa_MRC	25.50	34.50
Average	43.32	50.67

Table 11: Performance comparison with Chinese Wanjuan dataset.



(a) Proportion of different sampling count for $T_{\text{tgt}} = T_{\text{src}}$



(b) Sampling weight (i.e., $p(x)$) of different sampling counts for $T_{\text{tgt}} = T_{\text{src}}$

Figure 15: Analysis of sampling counts.