How to Synthesize Text Data without Model Collapse?

Xuekai Zhu [1 2] **Daixuan Cheng** [2] **Hengli Li** [2 3] **Kaiyan Zhang** [4] **Ermo Hua** [4] **Xingtai Lv** [4] **Ning Ding** [4]
Zhouhan Lin[† 1 5] **Zilong Zheng**[† 2] **Bowen Zhou**[† 4 5]

Abstract

Model collapse in synthetic data indicates that iterative training on self-generated data leads to a gradual decline in performance. With the proliferation of AI models, synthetic data will fundamentally reshape the web data ecosystem. Future GPT-$\{n\}$ models will inevitably be trained on a blend of synthetic and human-produced data. In this paper, we focus on two questions: what is the impact of synthetic data on language model training, and how to synthesize data without model collapse? We first pre-train language models across different proportions of synthetic data, revealing a negative correlation between the proportion of synthetic data and model performance. We further conduct statistical analysis on synthetic data to uncover distributional shift phenomenon and over-concentration of n-gram features. Inspired by the above findings, we propose token editing on human-produced data to obtain semi-synthetic data. As a proof of concept, we theoretically demonstrate that token-level editing can prevent model collapse, as the test error is constrained by a finite upper bound. We conduct extensive experiments on pre-training from scratch, continual pre-training, and supervised fine-tuning. The results validate our theoretical proof that token-level editing improves model performance.

1. Introduction

As generative artificial intelligence (AI) (Rombach et al., 2021; Achiam et al., 2023) becomes increasingly prevalent in research and industry, synthetic data will proliferate throughout the web data ecosystem. Consequently, future training of GPT-$\{n\}$ on a mixture of synthetic and human-produced data will be inevitable. Thus, model collapse is a critical concern that must be considered when training models on synthetic data.

Model collapse refers to a degenerative process in which the output data of learned generative models contaminates the training sets of subsequent generations. As shown in Figure 1, iterative training coupled with data synthesis induces a progressive accumulation of test errors (Shumailov et al., 2024; Dohmatob et al., 2024a). Consequently, generative models increasingly overfit to synthetic data distributions, failing to capture the complexity in human-produced data. Through successive iterations in Figure 1, these distortions accumulate, finally undermining the model's capacity.

Recent studies focus on two aspects. First, theoretical foundations of model collapse. Shumailov et al. (2024) and Dohmatob et al. (2024a) identify the model collapse phenomenon and formalize a theoretical framework based on linear regression. Gerstgrasser et al. (2024) demonstrate that if synthetic data is accumulated while retaining the initial real data, the test error will be bounded, thus breaking model collapse. Dohmatob et al. (2024c;b) indicate that missing long tails of synthetic data lead to scaling law cutoff. Second, practical implementations on synthetic datasets by diverse prompting. Synthetic datasets (Trinh et al., 2024; Zhang et al., 2024) have been proven to boost the capabilities of language models. Cheng et al. (2024a;b); Maini et al. (2024) rephrase text into more formal styles, thereby improving the data quality. There are still two key questions that require further investigation: **(Q1)** What is the impact of synthetic data on language model training? **(Q2)** How can data be synthesized without causing model collapse?

In this paper, we address the first question by training language models on varying mixtures of synthetic and human-produced data, demonstrating non-iterative model collapse. Unlike the original model collapse setting which iteratively trains on self-generated data, we directly mix synthetic and human-produced data to create training datasets with different mixing ratios. The results show a negative correlation between performance and the proportion of synthetic data.

[1]LUMIA Lab, Shanghai Jiao Tong University [2]State Key Laboratory of General Artificial Intelligence, BIGAI [3]Institute for Artificial Intelligence, Peking University [4]Department of Electronic Engineering, Tsinghua University [5]Shanghai Artificial Intelligence Laboratory. Correspondence to: Zhouhan Lin <lin.zhouhan@gmail.com>, Zilong Zheng <zlzheng@bigai.ai>, Bowen Zhou <zhoubowen@tsinghua.edu.cn>.

Proceedings of the 42^{nd} International Conference on Machine Learning, Vancouver, Canada. PMLR 267, 2025.

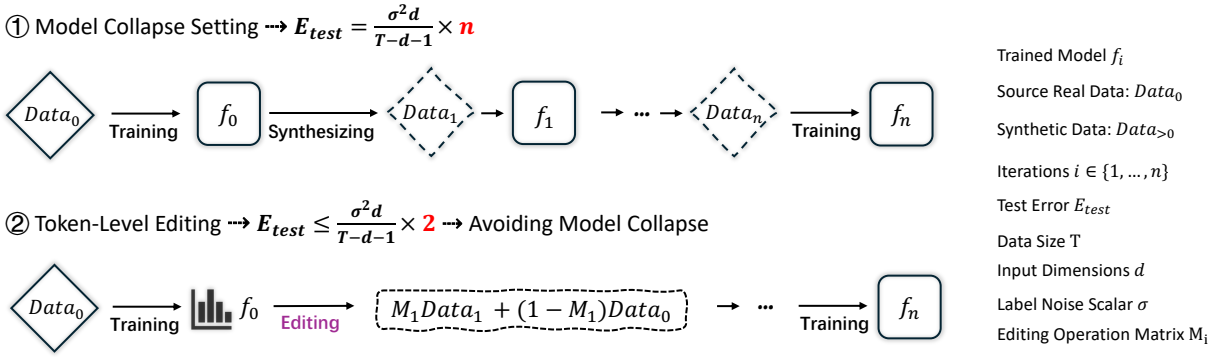


Figure 1. Model collapse of synthetic data. ① The model continuously trains on its previously generated data, leading to a gradual decline in model performance, i.e., model collapse. Starting from real data $Data_0$, the test error E_{test} increases as f_0 undergoes iterative training on synthetic data $Data_{>0}$. ② ToEdit (ours), we use a trained model for token-level editing rather than purely synthesizing data. Leveraging f_0 and an operation matrix M_i to edit the data, the test error is constrained within a fixed upper bound. Therefore, we can preserve the distribution coverage to avoid model collapse.

Subsequent statistical analysis on distributions and features indicates coverage narrowing—synthetic data covers only a small portion of the human-produced data distribution—and over-concentration of synthetic n-gram features. Based on the above findings, we address the second question by proposing **to**ken-level **edit**ing (ToEdit), resamples and replaces data points with relatively high model confidence. As illustrated in Figure 1, ToEdit preserves distribution coverage and theoretically constrains test error within a fixed upper bound. Extensive experiments across pre-training from scratch, continual pre-training, and supervised fine-tuning confirm its positive impact on model performance.[1]

Contributions. We summarize the key contributions of this work as follows[2]:

- We demonstrate non-iterative model collapse by pre-training language models on a mixture of synthetic and human-produced data (§ 3.1): directly mixing pure synthetic data, without iterative training, leads to performance degradation. Furthermore, we perform a distributional statistical analysis, revealing that synthetic data leads to coverage narrowing and over-concentration of n-gram features. Even subsequent data selection struggles to correct the distribution (§ 3.2).
- We propose token-level editing with a theoretical proof to prevent model collapse (§ 4) and validate its effectiveness through experiments spanning pre-training from scratch, continual pre-training, and supervised fine-tuning of language models (§ 5).

2. Background

Shumailov et al. (2024); Dohmatob et al. (2024a;c) demonstrate AI models trained recursively on data generated by earlier versions of themselves can result in performance degradation, ultimately rendering the AI model completely useless. This process can be formulated as follows:

$$E_{test}(\hat{w}_{n+1}) = \frac{\sigma^2 d}{T - d - 1} \times n.$$

This indicates that the error will continuously increase with the number of iterations n. The detailed theoretical notation is provided in § 4.2. Dohmatob et al. (2024c) further point out that synthetic data also contribute to a truncation of the scaling law. Gerstgrasser et al. (2024); Seddik et al. (2024) further adjust the data iteration setting to data accumulation or real data mixing. They demonstrate that data accumulation can prevent model collapse. Inspired by the above work, we further explore the impact of synthetic data in pre-training and analyze its differences from real data. Building on our findings, we propose token editing as a method to prevent model collapse during data synthesis. Further comparisons are in Appendix B and D.

3. Non-Iterative Model Collapse

Prior studies (Shumailov et al., 2024; Dohmatob et al., 2024a) investigate the curse of recursion, where iterative training on self-generated data leads to a degenerative process known as iterative model collapse. However, we often face direct data mixing of human-produced and synthetic data, and pre-training from scratch. We attempt to analyze model collapse in this more general scenario, called non-iterative model collapse. Specifically, we conduct pre-training on synthetic data mixtures and explore the reasons behind non-iterative model collapse through data distribution and characteristics. A more detailed comparison of iterative and non-iterative settings is in Appendix G.1.

[1]Work done during internship at BIGAI.

[2]Code repository available at https://github.com/Xuekai-Zhu/toedit.

Table 1. Subdomain PPL evaluation results for GPT-2 Small (124M) pre-trained on data mixture. The PPL increases as the proportion of synthetic data grows, providing further confirmation of Figure 2.

	ArXiv	Books2	Books3	Math	Enron	EuroParl	FreeLaw	GitHub	PG-19	HackerNews	NIH	Avg
Human data	22.26	25.39	22.87	10.84	23.50	30.73	12.04	4.15	16.88	32.54	23.53	20.99
25% Synthetic Data	21.86	26.32	23.87	11.05	24.85	35.02	12.84	4.35	17.99	33.80	23.76	22.06
50% Synthetic Data	22.50	28.01	25.75	10.84	26.56	41.99	14.02	4.67	19.70	36.12	24.61	23.48
75% Synthetic Data	24.35	31.19	28.98	11.81	30.30	56.32	16.03	5.30	22.75	40.44	26.19	27.60
Synthetic Data	35.60	43.72	47.72	17.25	66.97	129.75	29.62	12.00	50.14	87.95	39.48	51.93
	OpenSubts	**OWT2**	**Phil**	**Pile-CC**	**PubMed-A**	**PubMed-C**	**StackEx**	**Ubuntu**	**USPTO**	**Wikipedia**	**Youtube**	**Avg**
Human data	28.08	25.77	33.56	26.78	18.97	15.49	10.81	20.86	19.32	24.31	21.54	22.59
25% Synthetic Data	29.25	26.94	34.63	27.83	19.55	15.38	11.03	22.32	19.58	25.88	22.63	23.91
50% Synthetic Data	31.00	28.76	37.48	29.36	20.51	15.89	11.54	23.53	20.51	27.57	24.91	25.09
75% Synthetic Data	34.18	32.04	42.39	32.17	22.33	16.92	12.55	26.54	22.21	30.68	28.98	28.64
Synthetic Data	57.83	53.94	78.18	54.69	34.82	23.87	20.47	51.78	37.24	46.12	65.49	47.87

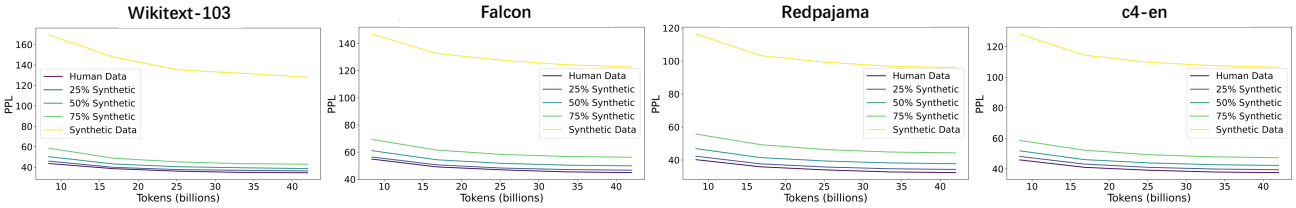


Figure 2. Non-iterative model collapse. Training language models from scratch on AI-synthesized data or a mixture of human and synthetic data leads to performance degradation. This degradation is negatively correlated with the proportion of synthetic data used in training. *Setting:* We pre-train GPT-2 Small (124M) on human data (Dolma (Soldaini et al., 2024)) and synthetic data (Cosmopedia (Ben Allal et al., 2024)) and evaluate the PPL on the Paloma benchmark (Magnusson et al., 2023). Training loss in Figure 7. Further validations on 22 subdomains and general downstream tasks are presented in Table 1 and Table 9, respectively.

3.1. Pre-training on Data Mixture

In this section, we investigate the impact of synthetic data on pre-training. Compared with studies on SFT and RLHF, we examine synthetic data integration in a more fundamental stage of the language model.

Setup We pre-train GPT-2 (Radford et al., 2019) and OLMo (Groeneveld et al., 2024) from scratch, using data mixtures containing 50B tokens each. We define the mixing ratio between human-produced and synthetic data as α, where $0 \leq \alpha \leq 1$. The total amount of training data D_{total} is a combination of human-produced data D_{human} and synthetic data $D_{\text{synthetic}}$, represented by the formula: $D_{\text{total}} = \alpha D_{\text{human}} + (1-\alpha) D_{\text{synthetic}}$. We use Dolma (Soldaini et al., 2024) as source human-produced data. We use Cosmopedia (Ben Allal et al., 2024) as the source synthetic data, which is distilled from `Mixtral-8x7B-Instruct-v0.1` (Jiang et al., 2024). To ensure rigorous validation and prevent data leakage, we construct a three-tier evaluation: (1) the Paloma benchmark (Magnusson et al., 2023), including carefully decontaminated test sets for Dolma; (2) comprehensive PPL evaluation across 22 subdomains from the Pile (Gao et al., 2020b); and (3) seven general downstream tasks as outlined in (Maini et al., 2024).

Finding I: Incorporating synthetic data harms the language models pre-training. PPL results of Paloma benchmark and 22 subdomains are presented in Figure 2 and Table 1, respectively. These results demonstrate that PPL on real-world validation sets increases as the proportion of synthetic data grows, indicating degraded model performance. When training from scratch, synthetic data does not benefit the model and may even hinder its learning process. However, incorporating human-produced data into the training mixture mitigates model collapse to some extent. Further results on general downstream tasks in Table 9 and 11 also corroborate the above findings. The overall trend shows a decline as the proportion of synthetic data increases, with models trained on purely synthetic data performing the worst. Compared to previous research on iterative model collapse (Shumailov et al., 2024; Dohmatob et al., 2024a;c), the non-iterative damage caused by synthetic data is more concerning and directly relevant to the training of next-generation language models.

3.2. Why Does Synthetic Data Fail in Pre-training?

We conduct three statistical analyses: (1) sample-level distribution, (2) feature-based overlap, and (3) distribution-reference data selection. The experimental results reveal that, compared to human-produced data, synthetic data lacks the long-tail samples and suffers from coverage narrowing. The limited diversity and concentrated features in synthetic data make using human-produced data as a reference to select synthetic data particularly challenging.

Setup We conduct statistical and feature-based analyses to explore why synthetic data fails in pre-training. (1) We lever-

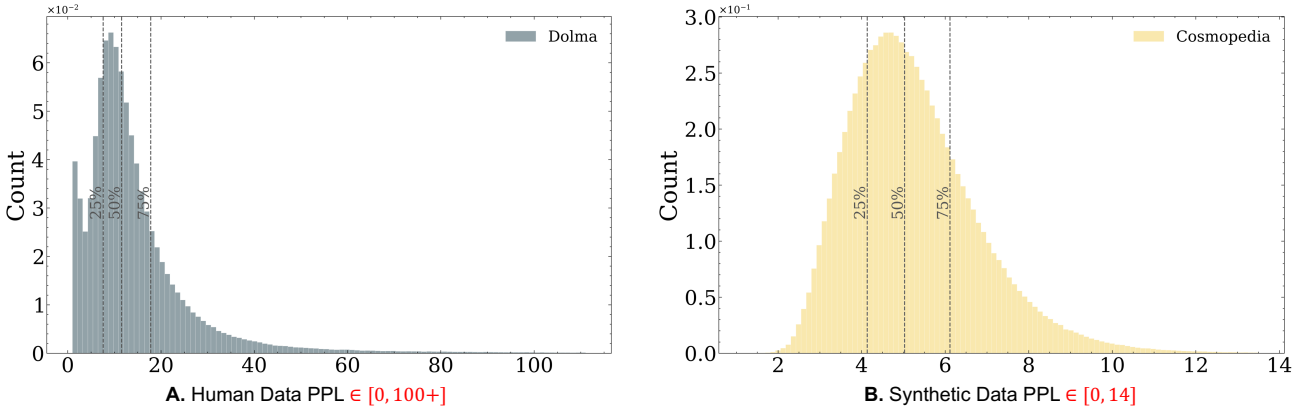


Figure 3. PPL distribution of human and synthetic data estimated by Llama-3-8B. The synthetic data lacks the long tail of the human-produced data and is also concentrated within the first 25% of the human-produced data distribution. **A.** Distribution of human-produced data is sharp with a long tail, spanning a wide range from 0 to over 100. **B.** The values are concentrated within a much narrower range, mostly between 0 and 14. The same trend estimated by StableLM-3B is demonstrated in Figure 10.

age a prior distribution to estimate the human-produced and synthetic data. We use Llama-3-8B (AI@Meta, 2024) and StableLM-Zephyr-3B (Bellagente et al., 2024). Different priors consistently yield the same results. (2) We analyze the n-gram features of human-produced and synthetic data from a feature-based perspective, such as n-gram response values. (3) Based on the features of human-produced data, we apply importance sampling (Xie et al., 2023) to filter synthetic data that closely aligns with human-produced features. More details of importance sampling are in § G.5.

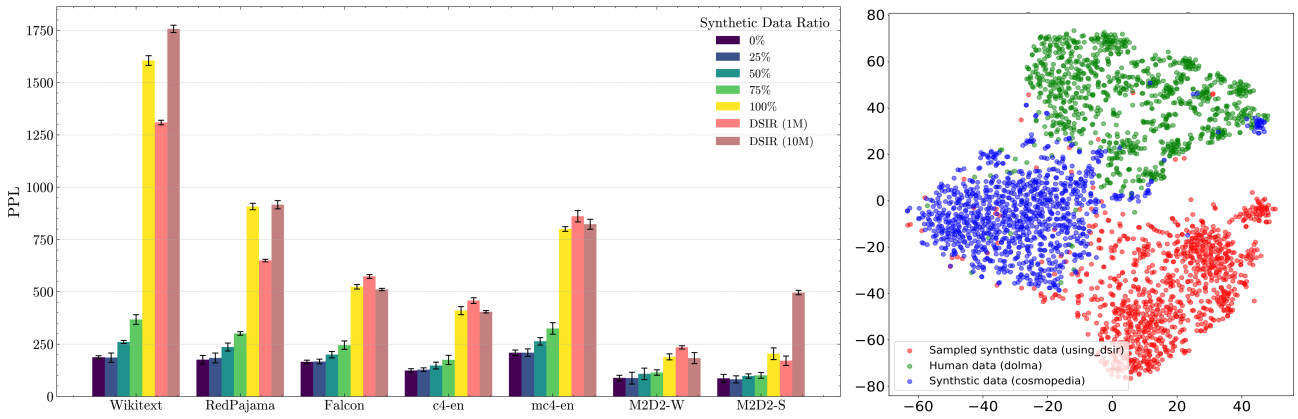
Finding II: Synthetic data distribution not only lacks long tails but also exhibits significant coverage narrowing. Figure 3 illustrate that the PPL of synthetic data is confined to the lower 25% of the human-produced data, failing to capture the full range and complexity of human-produced data distributions. Specifically, as illustrated in Figure 3A, human-produced data exhibit a wide distribution in the range $[1, 100+]$, characterized by a sharp peak and a pronounced long tail. In contrast, as shown in Figure 3B, the synthetic data is confined to a narrower range of $[0, 14]$, displaying a smoother distribution. Additional results of StabLM are shown in Figure 10. While the absolute PPL ranges estimated by different models may vary, the relative shapes and proportional ranges of these two distributions remain consistent. This phenomenon demonstrate that when scaling up to larger synthetic datasets, there will be a notable absence of the long tail, leading to severe coverage narrowing. This limited coverage reduces the generalization ability and contribute to model collapse.

Finding III: Synthetic data over-concentrates N-gram features. Based on the above distribution estimate, we further analyze why synthetic data fails at the feature level. Figure 12 and 13 demonstrate that synthetic data exhibits higher frequencies of certain bi-grams compared to human-produced data. To further examine feature-level differences, we hash uni-gram and bi-gram features into 10,000 hash buckets. As illustrated in Figure 11, human-produced data displays a noticeably broader response range, while synthetic data features are concentrated in a few specific buckets. This indirectly supports our earlier observation of feature over-concentration. We then expanded the hash bucket range to 1,000 × 20,000 buckets and used a locality-sensitive hashing method to differentiate the features more precisely. The results remain consistent. As shown in Figure 14, most response values are near zero. Distinguishing features in synthetic data remains challenging.

Finding IV: Distribution shifting cannot be mitigated through data selection. Inspired by recent data selection works (Xie et al., 2023; Albalak et al., 2024), we try to leverage the human-produced data features as a reference distribution for selecting synthetic samples. We apply importance sampling from DSIR (Xie et al., 2023) to filter synthetic data. As illustrated in Figure 4A, the training results of selected synthetic samples still fluctuates around the original performance of the synthetic data, indicating that even biased sampling cannot correct the distributional shift. As shown in Figure 4B, the sampled data still fails to align with human-produced data in the embedding space, even at the boundary regions of the synthetic data.

3.3. Proposed Strategy

Following these lessons so far, due to the coverage and feature over-concentration properties of synthetic data, the best approach is to rely entirely on human-produced data and avoid including synthetic data. However, we are still wondering how synthetic data can be used to enhance human-produced data. This leads to a general guideline for synthetic data: relying solely on synthetic data leads to model collapse, so preserving the primary human-produced data distribution is essential. In that case, we propose token-level



A. PPL Results for OLMo-237M **B.** Embedding Visualization using t-SNE

Figure 4. **A.** Pre-training results for selected synthetic data and other data mixtures on OLMo-237M. **B.** Embedding visualization between human-produced, synthetic, and DSIR-selected data using sentence-transformer.

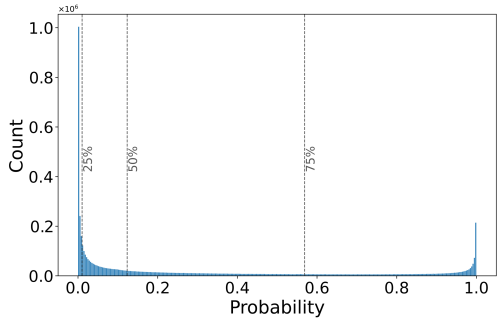


Figure 5. U-shape token probability distribution of Dolma-sampled V6 estimated by Qwen-0.5B-Instruct (qwe, 2024).

editing, which leverages a prior distribution to adjust the data. Our method can maintain the source distribution while improving the source data, called semi-synthetic data.

4. Token-Level Editing

We introduce token-level editing as a method for generating semi-synthetic data. Furthermore, we present a theoretical analysis and proof demonstrating that the test squared error of our method has a finite upper bound, regardless of the number of iterations. This ensures prevention of model collapse while enhancing performance.

4.1. Method

We formulate data synthesis as follows: assuming P is a prior distribution, given a sequence of tokens $\mathbf{x} = (x_1, \ldots, x_t)$, the full synthetic data is $\mathbf{y} = (y_1, \ldots, y_n)$. The synthesis process is derived as:

$$P(y_1, \ldots, y_n \mid \mathbf{x}) = \prod_{i=1}^{n} P(y_i \mid \ldots, y_{<i}, \mathbf{x}). \quad (1)$$

This conditional probability formulation outlines the generation of synthetic data conditioned on the given token sequence. Then the synthetic data is used to train models.

Inspired by previous studies of data selection (Mindermann et al., 2022; Ankner et al., 2024; Lin et al., 2024), prior distributions can serve as pointers indicating useless or learnable samples. In this case, we use a pre-trained language model to infer the pre-training corpus. As illustrated in Figure 5, even a model pre-trained on trillions of tokens can not fit the pre-training corpus perfectly. Specifically, 75% is under 0.6 probability. The tokens with both high and low probabilities are the most concentrated, suggesting the potential for data filtering. We leverage this U-shape distribution as a pointer to resample tokens. Specifically, we use a language model as prior distribution to compute each token's conditional probability $P(\cdot|\mathbf{x})$ if the probability exceeds a certain threshold $P(\cdot|\mathbf{x}) \geq p$, it indicates that the token is easy to learn, and we proceed with resampling at that point. The filtering potential of the U-shaped distribution is discussed in § G.2.

Token-level Editing doesn't generate the entire sequence but leverages conditional probability $P(x_i \mid x_1, \ldots, x_{i-1})$ to revise the input sequence. In this way, we can avoid using purely synthetic data while modifying the dataset to preserve long-tail features of human-produced data, aiming to obtain higher-quality semi-synthetic data. Token-level Editing can be formulated as follows:

$$x'_i = \begin{cases} x_i, & \text{if } P(x_i \mid x_1, \ldots, x_{i-1}) < p \\ \tilde{x}_i, & \text{if } P(x_i \mid x_1, \ldots, xi-1) \geq p \end{cases} \quad (2)$$

Where x'_i is the final token in the edited sequence. $\tilde{x}_i$ is a token resampled from a prior distribution. We can adjust the threshold p to balance retaining the structure of human-produced data while avoiding overfitting to synthetic data.

4.2. Theoretical Analysis

To gain deeper mathematical insights, we utilize an analytical framework of the linear model and adopt notations in prior research (Mobahi et al., 2020; Dohmatob et al., 2024a; Gerstgrasser et al., 2024). This theoretical framework primarily considers a linear model that iteratively trains on its own generated data, similar to pipelines like self-play and self-distillation, but without complex constraints. The process involves training continuously on the data generated by the previous generation of the model. (Dohmatob et al., 2024a) point out that with iterative training, test errors accumulate progressively, eventually leading to model collapse. Building on this theoretical framework, we integrate our proposed token editing method and analyze whether our method can prevent model collapse.

Notation and Preliminaries For a given distribution P_{Σ,w,σ^2}, the data $(x, y) \sim P_{\Sigma,w,\sigma^2}$ on $\mathbb{R}^d \times \mathbb{R}$, where x is drawn from a multivariate normal distribution $x \sim \mathcal{N}(0, \Sigma)$, ϵ is an independent noise term sampled from $\mathcal{N}(0, \sigma^2)$, and the label y is given by the linear model $y = x \cdot w^* + \epsilon$.

Iterative Data Editing Process We utilize the model obtained from the previous round of training to make a limited number of modifications. Specifically, we resample and replace data points with relatively high confidence. The editing operations are defined by the matrices $\{M_1, M_2, \ldots, M_n\}$. The iterative data synthesis and model-fitting process is formalized as follows:

$$P_{\Sigma,w^*,\sigma^2} \to P_{\Sigma,\hat{w}_1,\sigma^2} \to \ldots \to P_{\Sigma,\hat{w}_n,\sigma^2},$$

where n is the number of iterations. The detailed process of data editing and iterations is described as follows:

For $n = 1$, we begin by initializing the covariates/features as $\tilde{X}_1 = X$. The target values are defined as $\tilde{Y}_1 = \hat{Y}_1 = Xw^* + E_1$, where $E_1 \sim \mathcal{N}(0, \sigma^2 I_T)$. The linear model is then fitted by solving for $\hat{w}_1 = \tilde{X}_1^\dagger \tilde{Y}_1$. To proceed to the next iteration, we resample the data, obtaining $\hat{Y}_2 = X\hat{w}_1 + E_2$, with $E_2 \sim \mathcal{N}(0, \sigma^2 I_T)$.

For $n \geq 2$, the input covariates/features remain as $\tilde{X}_n^\top = X$, while the target values are updated using the edited targets, following the equation $\tilde{Y}_n^\top = M_{n-1}\hat{Y}_n + (1 - M_{n-1})\tilde{Y}_{n-1}$. The linear model is then fitted by computing $\hat{w}_n = \tilde{X}_n^\dagger \tilde{Y}_n$. Finally, the data is resampled for the next iteration, yielding $\hat{Y}_{n+1} = X\hat{w}_n + E_{n+1}$, where $E_{n+1} \sim \mathcal{N}(0, \sigma^2 I_T)$.

The matrix M_n is a diagonal matrix, where some diagonal elements are 1, while others are 0. The multiplication by M can be interpreted as an operation that selectively modifies certain data points (those corresponding to 1s) while retaining others (those corresponding to 0s). Then, the data editing process can be formulated as follows:

$$\tilde{Y}_n^\top = M_{n-1}\hat{Y}_n + (1 - M_{n-1})\tilde{Y}_{n-1} \tag{3}$$

where $\tilde{Y}_{n-1}$ is the data after editing in the $n-1$ generation, and $\hat{Y}_n$ is the synthetic data from the n-th generation. This process can be described as: firstly, synthesizing labels for all inputs; secondly, the M matrix determining which data is edited and which is retained. For a matrix A with full column rank, its Moore-Penrose pseudo-inverse is $A^+ = (A^\top A)^{-1} A^\top$. The noise terms $E_1, E_2, \ldots, E_n$ are independent of each other and the covariates/features. Since X has full column rank, $\tilde{X}_n$ retains this property for all $n \geq 1$.

Test Error Model collapse is ultimately reflected through test error. Following previous work, we adopt the standard test error definition as presented in (Gerstgrasser et al., 2024). For any linear estimator $\hat{w}$ derived from the training data, we evaluate the test error using the standard method:

$$E_{test}(w) \overset{\text{def}}{=} \mathbb{E}\left[(x_{test}^T w - y_{test})^2\right] - \sigma^2 = \mathbb{E}[\|w - w^*\|_\Sigma^2], \tag{4}$$

where the expectation is computed with respect to the training data, while the test pair $(x_{\text{test}}, y_{\text{test}})$ is sampled independently from P_{Σ,w^*,σ^2} of the training set.

4.3. Test Error Under Data Editing

Our goal is to derive an analytical expression for the test error of the n-th model in the data editing setting. As indicated by the test error in Eq. 4, this process involves two main steps: (1) establishing the relationship between the fitted linear parameters w_n and the true parameters w^*, and (2) simplifying the test error expression. We begin by formulating the relationship between w_n and w^*. Proofs are detailed in Appendix A.

Theorem 1. *In the data editing setting, $\forall n \geq 1$, the fitted linear parameters $\hat{w}_{n+1}$ can be derived as:*

$$\hat{w}_{n+1} = w^* + (X^\top X)^{-1} X^\top \left(E_1 + \sum_{i=1}^{n} M_i E_{i+1}\right) \tag{5}$$

where, w^* is the true parameter, X is the original design matrix, E_i is the extra noise added at the i'th iteration, and M_i is an idempotent diagonal matrix, defining the editing operation.

Theorem 2. *Consider an $n+1$ fold data editing process with $T \geq d+2$ samples per iteration and isotropic features ($\Sigma \overset{\text{def}}{=} I_d$), the test error for the ridgeless linear model $\hat{w}_n$ learned on the edited data up to iteration $n+1$, is bounded by:*

$$E_{test}(\hat{w}_{n+1}) \leq \frac{2\sigma^2 d}{T - d - 1} \tag{6}$$

Furthermore, assuming the editing operation satisfies $||M_i|| = ||M_{i-1}||\eta$ with $\eta \in (0, 1)$, the test error can be

Table 2. General performance of the pre-trained base models. PT indicates we pre-train OLMo-1B from scratch.

	PIQA	BoolQ	OBQA	ARC-c	ARC-e	HellaSwag	SIQA	Winogrande	Avg
OLMo-1B (PT)	53.97	38.26	12.20	17.23	28.36	26.02	34.80	51.14	32.75
Δ ToEdit	54.13	38.65	12.80	18.43	27.48	25.94	34.95	52.49	**33.11**

Table 3. Performance on domain-specific tasks for continual pre-training models. CPT indicates continual pre-training. Δ denotes training with our edited data. Our method demonstrates consistent improvements across three domains on both OLMo-1B and Llama-3-8B.

Biomedicine						
Models	MQP	ChemProt	PubMedQA	RCT	USMLE	Avg
OLMo-1B	52.59	17.2	51.40	32.70	28.90	36.63
CPT	52.29	21.00	58.50	34.90	27.49	38.83
Δ ToEdit	54.59	22.40	65.00	34.50	27.96	**40.89**
Llama-3-8B	66.80	28.59	60.8	73.85	40.61	54.13
CPT	72.29	29.4	69.1	72.65	36.76	56.04
Δ ToEdit	76.39	30.2	65.3	73.30	37.23	**56.48**
Finance						
Models	HeadLine	FPB	FiQA-SA	ConvFinQA	NER	Avg
OLMo-1B	69.00	47.03	48.05	4.83	62.19	46.22
CPT	70.31	49.78	40.36	18.72	60.44	47.92
Δ ToEdit	71.77	51.39	46.06	18.85	62.97	**50.21**
Llama-3-8B	81.28	63.58	81.60	52.88	72.53	70.37
CPT	85.68	54.22	81.88	67.78	67.43	71.40
Δ ToEdit	83.83	61.61	80.82	67.31	67.62	**72.24**
Math						
Models	ARC-c	GPQA	GSM8K	MATH	MMLU	Avg
OLMo-1B	28.67	24.23	1.67	0.00	26.56	16.23
CPT	28.41	24.03	1.52	0.10	27.23	16.26
Δ ToEdit	28.92	28.12	2.20	0.10	23.63	**16.59**

further bounded by:

$$E_{test}(\hat{w}_{n+1}) \leq \frac{\sigma^2 d}{T-d-1} + \sigma^2 \sqrt{\mathbb{E}\left[tr\left((X^\top X)^{-2}\right)\right]} \cdot \frac{\sqrt{\mathbb{E}\left[tr(M_1)\right]}}{1-\eta}.$$

We provide supporting evidence for the assumption in § G.3. Recalling model collapse (Dohmatob et al., 2024a): training iteratively on synthetic data leads to an accumulation of error over iterations, as shown in the following equation:

$$E_{\text{test}}^{\text{collapse}}(\hat{w}_n) = \frac{\sigma^2 d}{T-d-1} \times n \tag{7}$$

Comparing Eq. 6 with Eq. 7, the test error under data editing is bounded by a fixed value, preventing continuous error accumulation and thus avoiding model collapse. Based on the theoretical derivations and statistical analysis of synthetic data (§ 3.1), the underlying reason is that our approach retains the coverage of the initial distribution. We move away from pure data synthesis toward token-level editing, which allows us to obtain better data while avoiding model collapse. Moreover, remarkable previous studies (Dohmatob et al., 2024c; Gerstgrasser et al., 2024) pointed out similar conclusions. They indicated mixing real data with synthetic data will break model collapse and provide an upper bound under data accumulation. Different from their work, our data editing aims to yield better data, enabling synthetic data to perform well both in theory and practice.

5. Experiments

We validate our method in three language model training stages: pre-training from scratch (PT), continual pre-training (CPT), and supervised fine-tuning (SFT).

5.1. Implementation

We use the Llama-3-8B (AI@Meta, 2024) as a prior distribution to estimate the token distribution in each text sample. The modification probability is set to $p = 0.99$. This means that we resample tokens in positions where the probability exceeds p, and the resampling is based on the conditional probability given the preceding context. The entire process requires only a single forward pass, without autoregressive generation. We integrate the fast inference engine vLLM (Kwon et al., 2023), allowing the entire data editing

Table 4. Performance of the SFT tasks. We fine-tune LLaMA-3-8B using instruction tuning and code reasoning tasks, comparing performance with the edited version produced by our method. HS and WG are short for HellaSwag and Winogrande respectively.

	Models	PIQA	BoolQ	HS	SIQA	WG	Avg
Instruction Tuning							
Natural Ins.	Llama-3	79.82	87.06	58.32	46.83	74.66	69.34
	Δ ToEdit	80.58	87.80	58.27	46.93	74.90	**69.70**
CoT	Llama-3	79.87	81.28	59.72	49.69	74.51	69.01
	Δ ToEdit	80.25	81.16	59.74	50.56	74.59	**69.26**
FLANv2	Llama-3	80.79	84.04	59.98	51.43	74.66	70.18
	Δ ToEdit	80.69	85.20	59.99	52.00	75.37	**70.65**
Open Assist.	Llama-3	79.65	83.18	60.51	48.52	74.11	69.19
	Δ ToEdit	79.98	83.91	60.34	48.31	74.66	**69.44**
	Models	ARC-c	GPQA	GSM8K	MMLU		Avg
Code Reasoning							
OSS-Inst.	Llama-3	51.28	27.46	49.58	62.14		45.76
	Δ ToEdit	51.79	28.79	49.36	62.04		**46.13**
Evol-Ins.	Llama-3	52.90	27.90	50.87	62.40		46.62
	Δ ToEdit	52.22	29.69	50.87	62.60		**46.92**

process to be completed on a single 4090 GPU. We use top-k as the sampling strategy with $k = 8$. We also incorporate top-p sampling and rejection sampling in our ablation studies.

5.2. Datasets and Models

We provide an overview of our experimental setup, more details are in Appendix F. **For pre-training**, we pre-train the 1B OLMo model (Groeneveld et al., 2024) from scratch using Dolma-sampled V6 (6B tokens) and evaluate on 8 general tasks. **For continual pre-training**, we follow (Cheng et al., 2024a;b;c) to continual pre-train OLMo-1B (Groeneveld et al., 2024) and Llama-3-8B (AI@Meta, 2024) on corpora of Biomedicine, Finance, and Math, evaluating on 5 downstream tasks per domain. **For supervised fine-tuning**, we fine-tune Llama-3-8B on instruction tuning and code reasoning tasks, evaluating on 9 downstream tasks.

5.3. Main Results

Table 2, 3, and 4 respectively demonstrate the effectiveness of our method in pre-training from scratch, continual pre-training and fine-tuning tasks. Across these three stages of language model training, our method consistently enhances the model performance on downstream tasks without increasing the data size. This consistency is validated across two models. This indicates that our method unlocks the potential of existing data, demonstrating that semi-synthetic data is a viable path to improving model performance. A further numerical analysis is provided in § C.

Table 5. Ablations on resampled token condition (p) in biomedicine domain.

	PubMedQA	MQP	RCT	USMLE	ChemProt	Avg
$p \geq 0.99$	64.50	55.73	30.95	27.65	14.60	38.69
$p \geq 0.999$	63.60	55.40	29.09	28.12	16.20	38.48
$p \leq 0.1$	62.40	51.47	25.60	29.14	10.00	35.72
$p \leq 0.01$	65.40	54.91	28.19	27.80	11.00	37.46

Table 6. Token distribution across different probability intervals in the biomedicine domain dataset.

Interval	Percent	# Tokens	Interval	Percent	# Tokens
$[0.0, 0.1)$	34.7%	389M	$[0.5, 0.6)$	3.6%	40M
$[0.1, 0.2)$	8.1%	91M	$[0.6, 0.7)$	3.7%	41M
$[0.2, 0.3)$	5.4%	60M	$[0.7, 0.8)$	4.0%	44M
$[0.3, 0.4)$	4.4%	49M	$[0.8, 0.9)$	5.2%	58M
$[0.4, 0.5)$	3.8%	43M	$[0.9, 1.0)$	27.1%	303M

5.4. Ablation Studies

We conduct experiments on hyper-parameter p, including: (1) ablation studies on different values, (2) token percentage statistics, (3) comparisons of sampling strategies, and (4) an ablation study on sampling size.

Table 5 shows the impact of different p values on the model performance, with fluctuations observed across various settings. Table 6 presents the distribution percentages across different probability value ranges. As mentioned above, we need to refine the data while preserving mainly source distribution. As shown in Figure 5, a larger p indicates fewer tokens will be resampled, while a smaller p results in more tokens being resampled. To balance model performance and data distribution preservation, we set $p = 0.99$ as threshold for our experiments. Table 7 presents the results of different sampling strategies. Specifically, to control variables, we set $k = 8$ for top-k sampling and $p = 0.99$ for top-p sampling. We use rejection sampling implementation in (Liu et al., 2023). The results of reject sampling, top-p, and top-k are comparable. However, top-p involves a dynamic sampling range, and reject sampling requires multiple rounds of computation, leading to increased overhead. Considering computational efficiency, we choose top-k for sampling. Table 8 shows the ablation study on sampling size of top-k. The performance gain from increasing k is relatively small. Therefore, we set $k = 8$ in our experiments. And a detailed case for token editing is provided in Table 12.

6. Conclusion

With the growing prevalence of generative AI models, when training next-generation AI models, it will be inevitable to use a mixture of synthetic data and human-produced data. Therefore, we focus on two key questions: (1) What is the impact of synthetic data on language model pre-training,

Table 7. Ablations on sampling strategy.

Strategy	PubMedQA	MedMCQA	MedQA
Top-k	64.50	26.13	24.82
Top-p	63.80	27.11	25.61
Rejection Sampling	64.50	28.90	28.20

Table 8. Ablations on sampling size k for top-k.

Sampling Size (k)	PubMedQA	MedMCQA	MedQA
$k = 8$	64.50	26.13	24.82
$k = 64$	63.80	28.14	27.34

and what are the underlying causes? (2) How can we prevent model collapse and synthesize high-quality data? We found that synthetic data can impair the effectiveness of pre-training when mixed with human-produced data, leading to non-iterative model collapse. Statistical analysis reveals that synthetic data suffers from significant distribution gaps and overly concentrated n-gram features. We propose token-level editing instead of relying purely on synthetic data. Specifically, we perform token resampling guided by a trained prior. Theoretically, our method can prevent model collapse. Our approach demonstrates improvements over the source data across pre-training, continual pre-training, and supervised fine-tuning.

Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none which we feel must be specifically highlighted here.

Acknowledgement

This work is sponsored by the National Key Research and Development Program of China (No. 2023ZD0121402). X.Z., H.L. and Z.Z. are supported by the National Natural Science Foundation of China (62376031).

References

Qwen2 technical report. 2024.

Achiam, J., Adler, S., Agarwal, S., Ahmad, L., Akkaya, I., Aleman, F. L., Almeida, D., Altenschmidt, J., Altman, S., Anadkat, S., et al. Gpt-4 technical report. arXiv preprint arXiv:2303.08774, 2023.

AI@Meta. Llama 3 model card. 2024. URL https://github.com/meta-llama/llama3/blob/main/MODEL_CARD.md.

Albalak, A., Elazar, Y., Xie, S. M., Longpre, S., Lambert, N., Wang, X., Muennighoff, N., Hou, B., Pan, L., Jeong, H., et al. A survey on data selection for language models. arXiv preprint arXiv:2402.16827, 2024.

Alemohammad, S., Casco-Rodriguez, J., Luzi, L., Humayun, A. I., Babaei, H., LeJeune, D., Siahkoohi, A., and Baraniuk, R. G. Self-consuming generative models go mad. arXiv preprint arXiv:2307.01850, 4:14, 2023.

Ankner, Z., Blakeney, C., Sreenivasan, K., Marion, M., Leavitt, M. L., and Paul, M. Perplexed by perplexity: Perplexity-based data pruning with small reference models. arXiv preprint arXiv:2405.20541, 2024.

Bai, F., Zhang, H., Tao, T., Wu, Z., Wang, Y., and Xu, B. Picor: Multi-task deep reinforcement learning with policy correction. Proceedings of the AAAI Conference on Artificial Intelligence, 37(6):6728–6736, Jun. 2023.

Bai, F., Wang, M., Zhang, Z., Chen, B., Xu, Y., Wen, Y., and Yang, Y. Efficient model-agnostic alignment via bayesian persuasion. arXiv preprint arXiv:2405.18718, 2024.

Bai, F., Liu, R., Du, Y., Wen, Y., and Yang, Y. Rat: Adversarial attacks on deep reinforcement agents for targeted behaviors. In Proceedings of the AAAI Conference on Artificial Intelligence, volume 39, pp. 15453–15461, 2025.

Bellagente, M., Tow, J., Mahan, D., Phung, D., Zhuravinskyi, M., Adithyan, R., Baicoianu, J., Brooks, B., Cooper, N., Datta, A., et al. Stable lm 2 1.6 b technical report. arXiv preprint arXiv:2402.17834, 2024.

Ben Allal, L., Lozhkov, A., Penedo, G., Wolf, T., and von Werra, L. Cosmopedia, 2024. URL https://huggingface.co/datasets/HuggingFaceTB/cosmopedia.

Bertrand, Q., Bose, A. J., Duplessis, A., Jiralerspong, M., and Gidel, G. On the stability of iterative retraining of generative models on their own data. arXiv preprint arXiv:2310.00429, 2023.

Briesch, M., Sobania, D., and Rothlauf, F. Large language models suffer from their own output: An analysis of the self-consuming training loop. arXiv preprint arXiv:2311.16822, 2023.

Cheng, D., Gu, Y., Huang, S., Bi, J., Huang, M., and Wei, F. Instruction pre-training: Language models are supervised multitask learners. In Conference on Empirical Methods in Natural Language Processing, 2024a. URL https://api.semanticscholar.org/CorpusID:270620509.

Cheng, D., Huang, S., and Wei, F. Adapting large language models via reading comprehension. In The Twelfth

International Conference on Learning Representations, 2024b. URL https://openreview.net/forum?id=y886UXPEZ0.

Cheng, D., Huang, S., Zhu, Z., Zhang, X., Zhao, W. X., Luan, Z., Dai, B., and Zhang, Z. On domain-specific post-training for multimodal large language models. arXiv preprint arXiv:2411.19930, 2024c.

Dohmatob, E., Feng, Y., and Kempe, J. Model collapse demystified: The case of regression. arXiv preprint arXiv:2402.07712, 2024a.

Dohmatob, E., Feng, Y., Subramonian, A., and Kempe, J. Strong model collapse. arXiv preprint arXiv:2410.04840, 2024b.

Dohmatob, E., Feng, Y., Yang, P., Charton, F., and Kempe, J. A tale of tails: Model collapse as a change of scaling laws. arXiv preprint arXiv:2402.07043, 2024c.

Eldan, R. and Li, Y. Tinystories: How small can language models be and still speak coherent english? arXiv preprint arXiv:2305.07759, 2023.

Feng, Y., Dohmatob, E., Yang, P., Charton, F., Kempe, J., and Meta, F. Beyond model collapse: Scaling up with syn-thesized data requires verification. arXiv preprint arXiv:2406.07515, 2024.

Ferbach, D., Bertrand, Q., Bose, A. J., and Gidel, G. Self-consuming generative models with curated data provably optimize human preferences. arXiv preprint arXiv:2407.09499, 2024.

Gao, L., Biderman, S., Black, S., Golding, L., Hoppe, T., Foster, C., Phang, J., He, H., Thite, A., Nabeshima, N., Presser, S., and Leahy, C. The pile: An 800gb dataset of diverse text for language modeling. ArXiv, abs/2101.00027, 2020a. URL https://api.semanticscholar.org/CorpusID:230435736.

Gao, L., Biderman, S., Black, S., Golding, L., Hoppe, T., Foster, C., Phang, J., He, H., Thite, A., Nabeshima, N., et al. The pile: An 800gb dataset of diverse text for language modeling. arXiv preprint arXiv:2101.00027, 2020b.

Gao, L., Tow, J., Abbasi, B., Biderman, S., Black, S., DiPofi, A., Foster, C., Golding, L., Hsu, J., Le Noac'h, A., Li, H., McDonell, K., Muennighoff, N., Ociepa, C., Phang, J., Reynolds, L., Schoelkopf, H., Skowron, A., Sutawika, L., Tang, E., Thite, A., Wang, B., Wang, K., and Zou, A. A framework for few-shot language model evaluation, 07 2024. URL https://zenodo.org/records/12608602.

Gerstgrasser, M., Schaeffer, R., Dey, A., Rafailov, R., Sleight, H., Hughes, J., Korbak, T., Agrawal, R., Pai, D., Gromov, A., et al. Is model collapse inevitable? breaking the curse of recursion by accumulating real and synthetic data. arXiv preprint arXiv:2404.01413, 2024.

Groeneveld, D., Beltagy, I., Walsh, P., Bhagia, A., Kinney, R., Tafjord, O., Jha, A., Ivison, H., Magnusson, I., Wang, Y., Arora, S., Atkinson, D., Authur, R., Chandu, K. R., Cohan, A., Dumas, J., Elazar, Y., Gu, Y., Hessel, J., Khot, T., Merrill, W., Morrison, J. D., Muennighoff, N., Naik, A., Nam, C., Peters, M. E., Pyatkin, V., Ravichander, A., Schwenk, D., Shah, S., Smith, W., Strubell, E., Subramani, N., Wortsman, M., Dasigi, P., Lambert, N., Richardson, K., Zettlemoyer, L., Dodge, J., Lo, K., Soldaini, L., Smith, N. A., and Hajishirzi, H. Olmo: Accelerating the science of language models. arXiv preprint, 2024. URL https://api.semanticscholar.org/CorpusID:267365485.

Gulcehre, C., Paine, T. L., Srinivasan, S., Konyushkova, K., Weerts, L., Sharma, A., Siddhant, A., Ahern, A., Wang, M., Gu, C., et al. Reinforced self-training (rest) for language modeling. arXiv preprint arXiv:2308.08998, 2023.

Gunasekar, S., Zhang, Y., Aneja, J., Mendes, C. C. T., Del Giorno, A., Gopi, S., Javaheripi, M., Kauffmann, P., de Rosa, G., Saarikivi, O., et al. Textbooks are all you need. arXiv preprint arXiv:2306.11644, 2023.

Hu, E. J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., and Chen, W. Lora: Low-rank adaptation of large language models. arXiv preprint arXiv:2106.09685, 2021.

Jia, X., Yang, Z., Li, Q., Zhang, Z., and Yan, J. Bench2drive: Towards multi-ability benchmarking of closed-loop end-to-end autonomous driving. arXiv preprint arXiv:2406.03877, 2024.

Jiang, A. Q., Sablayrolles, A., Roux, A., Mensch, A., Savary, B., Bamford, C., Chaplot, D. S., Casas, D. d. l., Hanna, E. B., Bressand, F., et al. Mixtral of experts. arXiv preprint arXiv:2401.04088, 2024.

Kazdan, J., Schaeffer, R., Dey, A., Gerstgrasser, M., Rafailov, R., Donoho, D. L., and Koyejo, S. Collapse or thrive? perils and promises of synthetic data in a self-generating world, 2025. URL https://openreview.net/forum?id=Xr5iINA3zU.

Kopf, A., Kilcher, Y., von Rutte, D., Anagnostidis, S., Tam, Z. R., Stevens, K., Barhoum, A., Duc, N. M., Stanley, O., Nagyfi, R., Shahul, E., Suri, S., Glushkov, D., Dantuluri, A., Maguire, A., Schuhmann, C., Nguyen, H., and Mattick, A. Openassistant conversations - democratizing

large language model alignment. ArXiv, abs/2304.07327, 2023. URL https://api.semanticscholar.org/CorpusID:258179434.

Kwon, W., Li, Z., Zhuang, S., Sheng, Y., Zheng, L., Yu, C. H., Gonzalez, J. E., Zhang, H., and Stoica, I. Efficient memory management for large language model serving with pagedattention. In Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles, 2023.

Li, L., Lyu, J., Ma, G., Wang, Z., Yang, Z., Li, X., and Li, Z. Normalization enhances generalization in visual reinforcement learning. arXiv preprint arXiv:2306.00656, 2023.

Li, Q., Jia, X., Wang, S., and Yan, J. Think2drive: Efficient reinforcement learning by thinking in latent world model for quasi-realistic autonomous driving (in carla-v2). arXiv preprint arXiv:2402.16720, 2024.

Li, X., Yang, Z., and Wu, H. Face detection based on receptive field enhanced multi-task cascaded convolutional neural networks. IEEE access, 8:174922–174930, 2020.

Lin, Z., Gou, Z., Gong, Y., Liu, X., Shen, Y., Xu, R., Lin, C., Yang, Y., Jiao, J., Duan, N., et al. Rho-1: Not all tokens are what you need. arXiv preprint arXiv:2404.07965, 2024.

Liu, R., Wei, J., Liu, F., Si, C., Zhang, Y., Rao, J., Zheng, S., Peng, D., Yang, D., Zhou, D., et al. Best practices and lessons learned on synthetic data for language models. arXiv preprint arXiv:2404.07503, 2024.

Liu, T., Zhao, Y., Joshi, R., Khalman, M., Saleh, M., Liu, P. J., and Liu, J. Statistical rejection sampling improves preference optimization. ArXiv, abs/2309.06657, 2023. URL https://api.semanticscholar.org/CorpusID:261705578.

Longpre, S., Hou, L., Vu, T., Webson, A., Chung, H. W., Tay, Y., Zhou, D., Le, Q. V., Zoph, B., Wei, J., et al. The flan collection: Designing data and methods for effective instruction tuning. arXiv preprint arXiv:2301.13688, 2023.

Magnusson, I., Bhagia, A., Hofmann, V., Soldaini, L., Jha, A., Tafjord, O., Schwenk, D., Walsh, P., Elazar, Y., Lo, K., Groeneveld, D., Beltagy, I., Hajishirzi, H., Smith, N. A., Richardson, K., and Dodge, J. Paloma: A benchmark for evaluating language model fit. ArXiv, abs/2312.10523, 2023. URL https://api.semanticscholar.org/CorpusID:266348815.

Maini, P., Seto, S., Bai, R. H., Grangier, D., Zhang, Y., and Jaitly, N. Rephrasing the web: A recipe for compute and data-efficient language modeling. In Annual Meeting of the Association for Computational Linguistics, 2024. URL https://api.semanticscholar.org/CorpusID:267312030.

Martínez, G., Watson, L., Reviriego, P., Hernández, J. A., Juarez, M., and Sarkar, R. Towards understanding the interplay of generative artificial intelligence and the internet. In International Workshop on Epistemic Uncertainty in Artificial Intelligence, pp. 59–73. Springer, 2023.

Mindermann, S., Brauner, J. M., Razzak, M. T., Sharma, M., Kirsch, A., Xu, W., Höltgen, B., Gomez, A. N., Morisot, A., Farquhar, S., et al. Prioritized training on points that are learnable, worth learning, and not yet learnt. In International Conference on Machine Learning, pp. 15630–15649. PMLR, 2022.

Mobahi, H., Farajtabar, M., and Bartlett, P. Self-distillation amplifies regularization in hilbert space. Advances in Neural Information Processing Systems, 33:3351–3361, 2020.

Niu, Y., Pu, Y., Yang, Z., Li, X., Zhou, T., Ren, J., Hu, S., Li, H., and Liu, Y. Lightzero: A unified benchmark for monte carlo tree search in general sequential decision scenarios. Advances in Neural Information Processing Systems, 36, 2024.

Pu, Y., Niu, Y., Yang, Z., Ren, J., Li, H., and Liu, Y. Unizero: Generalized and efficient planning with scalable latent world models. arXiv preprint arXiv:2406.10667, 2024.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., and Sutskever, I. Language models are unsupervised multitask learners. 2019.

Rombach, R., Blattmann, A., Lorenz, D., Esser, P., and Ommer, B. High-resolution image synthesis with latent diffusion models, 2021.

Rudin, W. Principles of Mathematical Analysis. McGraw-Hill, New York, 3rd edition, 1976.

Seddik, M. E. A., Chen, S.-W., Hayou, S., Youssef, P., and Debbah, M. How bad is training on synthetic data? a statistical analysis of language model collapse. arXiv preprint arXiv:2404.05090, 2024.

Shumailov, I., Shumaylov, Z., Zhao, Y., Papernot, N., Anderson, R., and Gal, Y. Ai models collapse when trained on recursively generated data. Nature, 631(8022):755–759, 2024.

Singh, A., Co-Reyes, J. D., Agarwal, R., Anand, A., Patil, P., Garcia, X., Liu, P. J., Harrison, J., Lee, J., Xu, K., et al. Beyond human data: Scaling self-training for problem-solving with language models. arXiv preprint arXiv:2312.06585, 2023.

Soldaini, L., Kinney, R., Bhagia, A., Schwenk, D., Atkinson, D., Authur, R., Bogin, B., Chandu, K., Dumas, J., Elazar, Y., Hofmann, V., Jha, A. H., Kumar, S., Lucy, L., Lyu, X., Lambert, N., Magnusson, I., Morrison, J., Muennighoff, N., Naik, A., Nam, C., Peters, M. E., Ravichander, A., Richardson, K., Shen, Z., Strubell, E., Subramani, N., Tafjord, O., Walsh, P., Zettlemoyer, L., Smith, N. A., Hajishirzi, H., Beltagy, I., Groeneveld, D., Dodge, J., and Lo, K. Dolma: An Open Corpus of Three Trillion Tokens for Language Model Pretraining Research. arXiv preprint, 2024. URL https://arxiv.org/abs/2402.00159.

Tan, Z., Li, D., Wang, S., Beigi, A., Jiang, B., Bhattacharjee, A., Karami, M., Li, J., Cheng, L., and Liu, H. Large language models for data annotation and synthesis: A survey. In Al-Onaizan, Y., Bansal, M., and Chen, Y.-N. (eds.), Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing, pp. 930–957, Miami, Florida, USA, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.emnlp-main.54. URL https://aclanthology.org/2024.emnlp-main.54/.

Trinh, T., Wu, Y., Le, Q., He, H., and Luong, T. Solving olympiad geometry without human demonstrations. Nature, 2024. doi: 10.1038/s41586-023-06747-5.

Ulmer, D., Mansimov, E., Lin, K., Sun, J., Gao, X., and Zhang, Y. Bootstrapping llm-based task-oriented dialogue agents via self-talk. arXiv preprint arXiv:2401.05033, 2024.

Wang, Y., Kordi, Y., Mishra, S., Liu, A., Smith, N. A., Khashabi, D., and Hajishirzi, H. Self-instruct: Aligning language models with self-generated instructions. arXiv preprint arXiv:2212.10560, 2022.

Wei, J., Wang, X., Schuurmans, D., Bosma, M., hsin Chi, E. H., Xia, F., Le, Q., and Zhou, D. Chain of thought prompting elicits reasoning in large language models. ArXiv, abs/2201.11903, 2022. URL https://api.semanticscholar.org/CorpusID:246411621.

Xia, M., Malladi, S., Gururangan, S., Arora, S., and Chen, D. Less: Selecting influential data for targeted instruction tuning. ArXiv, abs/2402.04333, 2024. URL https://api.semanticscholar.org/CorpusID:267522839.

Xie, S. M., Santurkar, S., Ma, T., and Liang, P. S. Data selection for language models via importance resampling. Advances in Neural Information Processing Systems, 36: 34201–34227, 2023.

Yang, Z., Jia, X., Li, H., and Yan, J. Llm4drive: A survey of large language models for autonomous driving. arXiv e-prints, pp. arXiv–2311, 2023.

Zhang, K., Zeng, S., Hua, E., Ding, N., Chen, Z.-R., Ma, Z., Li, H., Cui, G., Qi, B., Zhu, X., et al. Ultramedical: Building specialized generalists in biomedicine. arXiv preprint arXiv:2406.03949, 2024.

Zhang, M., Zhang, S., Yang, Z., Chen, L., Zheng, J., Yang, C., Li, C., Zhou, H., Niu, Y., and Liu, Y. Gobigger: A scalable platform for cooperative-competitive multi-agent interactive simulation. In The Eleventh International Conference on Learning Representations, 2023.

Zheng, Y., Zhang, R., Zhang, J., Ye, Y., Luo, Z., Feng, Z., and Ma, Y. Llamafactory: Unified efficient fine-tuning of 100+ language models. In Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations), Bangkok, Thailand, 2024. Association for Computational Linguistics. URL http://arxiv.org/abs/2403.13372.

Zhu, X., Guan, J., Huang, M., and Liu, J. Storytrans: Non-parallel story author-style transfer with discourse representations and content enhancing. arXiv preprint arXiv:2208.13423, 2022.

Zhu, X., Fu, Y., Zhou, B., and Lin, Z. Critical data size of language models from a grokking perspective. arXiv preprint arXiv:2401.10463, 2024a.

Zhu, X., Qi, B., Zhang, K., Long, X., Lin, Z., and Zhou, B. PaD: Program-aided distillation can teach small models reasoning better than chain-of-thought fine-tuning. In Duh, K., Gomez, H., and Bethard, S. (eds.), Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers), pp. 2571–2597, Mexico City, Mexico, June 2024b. Association for Computational Linguistics. doi: 10.18653/v1/2024.naacl-long.142. URL https://aclanthology.org/2024.naacl-long.142.

A. Proof

A.1. Proof of Theorem 1

For $n = 1$, we have:

$$\hat{w}_1 = \tilde{X}_1^\dagger \tilde{Y}_1 = (X^\top X)^{-1} X^\top (X w^* + E_1) = w^* + (X^\top X)^{-1} X^\top E_1$$

For $n \geq 1$, we have:

$$\begin{aligned}
\hat{w}_{n+1} &= \tilde{X}_{n+1}^\dagger \tilde{Y}_{n+1} \\
&= (\tilde{X}_{n+1}^\top \tilde{X}_{n+1})^{-1} \tilde{X}_{n+1}^\top \tilde{Y}_{n+1} \\
&= (X^\top X)^{-1} X^\top \tilde{Y}_{n+1}
\end{aligned}$$

Recalling that:

$$\tilde{Y}_i = \begin{cases} X w^* + E_1, & \text{if } i = 1 \\ M_{i-1}(X \hat{w}_{i-1} + E_i) + (1 - M_{i-1}) \tilde{Y}_{i-1}, & \text{if } 2 \leq i \leq n+1 \end{cases}$$

Substituting this $\tilde{Y}_i$ into the expression for $\hat{w}_{n+1}$:

We begin the data editing data process:

$$\tilde{Y}_2 = M_1(X \hat{w}_1 + E_2) + (1 - M_1) \tilde{Y}_1 \tag{8}$$

Then:

$$\tilde{Y}_3 = M_2(X \hat{w}_2 + E_3) + (1 - M_2) \tilde{Y}_2 \tag{9}$$

We have:

$$\begin{aligned}
\tilde{Y}_3 &= M_2(X \hat{w}_2 + E_3) + (1 - M_2) \left(M_1(X \hat{w}_1 + E_2) + (1 - M_1) \tilde{Y}_1 \right) \\
&= M_2(X \hat{w}_2 + E_3) + (1 - M_2) M_1(X \hat{w}_1 + E_2) + (1 - M_2)(1 - M_1) \tilde{Y}_1
\end{aligned}$$

We can expand $\tilde{Y}_{n+1}$ by recursively substituting the previous expressions:

$$\begin{aligned}
\tilde{Y}_{n+1} &= M_n(X \hat{w}_n + E_{n+1}) + (1 - M_n) \tilde{Y}_n && (10) \\
&= M_n(X \hat{w}_n + E_{n+1}) + (1 - M_n) \left[M_{n-1}(X \hat{w}_{n-1} + E_n) + (1 - M_{n-1}) \tilde{Y}_{n-1} \right] && (11) \\
&= M_n(X \hat{w}_n + E_{n+1}) + (1 - M_n) M_{n-1}(X \hat{w}_{n-1} + E_n) + (1 - M_n)(1 - M_{n-1}) \tilde{Y}_{n-1} && (12) \\
&\vdots && (13) \\
&= \sum_{i=1}^{n} \left[\left(\prod_{j=i+1}^{n} (1 - M_j) \right) M_i(X \hat{w}_i + E_{i+1}) \right] + \left(\prod_{j=1}^{n} (1 - M_j) \right) \tilde{Y}_1 && (14)
\end{aligned}$$

Recalling properties of M_i:

$$\begin{aligned}
M_i(1 - M_i) = 0 \quad \text{and} \quad (1 - M_i) M_i = 0 && (15) \\
M_i M_j = 0 \quad \text{for} \quad i \neq j && (16) \\
(1 - M_i)(1 - M_j) = 1 - M_i - M_j \quad \text{for} \quad i \neq j && (17) \\
&& (18)
\end{aligned}$$

Then we have:

$$\tilde{Y}_{n+1} = \sum_{i=1}^{n} M_i(X\hat{w}_i + E_{i+1}) + \left(1 - \sum_{i=1}^{n} M_i\right)\tilde{Y}_1 \tag{19}$$

$$= \sum_{i=1}^{n} M_i(X\hat{w}_i + E_{i+1}) + \left(1 - \sum_{i=1}^{n} M_i\right)(Xw^* + E_1) \tag{20}$$

$$= Xw^* + E_1 + \sum_{i=1}^{n} M_i \left(X(\hat{w}_i - w^*) + (E_{i+1} - E_1)\right) \tag{21}$$

Substituting this back into the expression for $\hat{w}_{n+1}$:

$$\hat{w}_{n+1} = (X^\top X)^{-1}X^\top \left[Xw^* + E_1 + \sum_{i=1}^{n} M_i \left(X(\hat{w}_i - w^*) + (E_{i+1} - E_1)\right)\right] \tag{22}$$

$$= w^* + (X^\top X)^{-1}X^\top \left[E_1 + \sum_{i=1}^{n} M_i X(\hat{w}_i - w^*) + \sum_{i=1}^{n} M_i(E_{i+1} - E_1)\right] \tag{23}$$

We can observe:

$$\hat{w}_1 = (X^\top X)^{-1}X^\top(Xw^* + E_1) = w^* + (X^\top X)^{-1}X^\top E_1 \tag{24}$$

$$\hat{w}_2 = w^* + (X^\top X)^{-1}X^\top \left(M_1 X(X^\top X)^{-1}X^\top E_1 + M_1 E_2 + (1 - M_1)E_1\right) \tag{25}$$

$$= w^* + (X^\top X)^{-1}X^\top \left(E_1 + M_1 E_2\right) \tag{26}$$

We prove this Theorem 1 by induction.

Inductive Step: Assume the formula holds for n, we have:

$$\hat{w}_{n+1} = w^* + (X^\top X)^{-1}X^\top \left(E_1 + M_1 E_2 + M_2 E_3 + \cdots + M_n E_{n+1}\right) \tag{27}$$

$$= w^* + (X^\top X)^{-1}X^\top \left(E_1 + \sum_{i=1}^{n} M_i E_{i+1}\right) \tag{28}$$

Substitute $\hat{w}_i$ into $\hat{w}_{n+1}$:

Then we can get:

$$\hat{w}_{n+1} = w^* + (X^\top X)^{-1}X^\top \left[E_1 + \sum_{i=1}^{n} M_i P \left(E_1 + \sum_{j=1}^{i-1} M_j E_{j+1}\right) + \sum_{i=1}^{n} M_i(E_{i+1} - E_1)\right] \tag{29}$$

$$= w^* + (X^\top X)^{-1}X^\top \left[E_1 + \sum_{i=1}^{n} M_i \left(E_{i+1} + \sum_{j=1}^{i-1} M_j E_{j+1}\right)\right] \tag{30}$$

$$= w^* + (X^\top X)^{-1}X^\top \left(E_1 + \sum_{i=1}^{n} M_i E_{i+1}\right) \tag{31}$$

$$\text{where} \quad P = X(X^\top X)^{-1}X^\top, \tag{32}$$

The above derivation aligns with Theorem 1, and the proof is complete.

A.2. Proof of Theorem 4.3

We substitute the Eq. 28 into Test Error Eq. 4:

$$E_{test}(\hat{w}_{n+1}) = \mathbb{E}\left[\left\|(X^\top X)^{-1}X^\top\left(E_1 + \sum_{i=1}^{n} M_i E_{i+1}\right)\right\|_\Sigma^2\right] \tag{33}$$

$$= \mathbb{E}\left[\left(E_1 + \sum_{i=1}^{n} M_i E_{i+1}\right)^\top X(X^\top X)^{-2}X^\top\left(E_1 + \sum_{i=1}^{n} M_i E_{i+1}\right)\right] \tag{34}$$

$$= \sigma^2\mathbb{E}\left[\text{tr}\left((X^\top X)^{-1}\right)\right] + \sigma^2\sum_{i=1}^{n}\mathbb{E}\left[\text{tr}\left(M_i(X^\top X)^{-1}M_i\right)\right] \tag{35}$$

$$= \sigma^2\mathbb{E}\left[\text{tr}\left((X^\top X)^{-1}\right)\right] + \sigma^2\sum_{i=1}^{n}\mathbb{E}\left[\text{tr}\left((X^\top X)^{-1}M_i\right)\right] \tag{36}$$

Further, by applying the Cauchy-Schwarz inequality (Rudin, 1976), we obtain:

$$E_{test}(\hat{w}_{n+1}) \leq \sigma^2\mathbb{E}\left[\text{tr}\left((X^\top X)^{-1}\right)\right] + \sigma^2\sqrt{\mathbb{E}\left[\text{tr}\left((X^\top X)^{-2}\right)\right]}\cdot\sum_{i=1}^{n}\sqrt{\mathbb{E}\left[\text{tr}(M_i)\right]} \tag{37}$$

We refer to the following lemma (Dohmatob et al., 2024a), which is essential for proving Theorem 2:

Lemma 3. *Let T and d be positive integers with $T \geq d+2$, and let $X \in \mathbb{R}^{T\times d}$ be a random matrix with i.i.d. rows from $\mathcal{N}(0,\Sigma)$ with Σ positive definite. Then, X has full rank a.s. Moreover, it holds that:*

$$\mathbb{E}_X\left[(X^\top X)^{-1}\right] = \frac{1}{T-d-1}\Sigma^{-1}. \tag{38}$$

Using Lemma 3, we have:

$$E_{test}\left[\text{tr}\left((X^\top X)^{-1}\right)\right] = \frac{d}{T-d-1} \tag{39}$$

Then, we have:

$$E_{test}(\hat{w}_{n+1}) = \sigma^2\mathbb{E}\left[\text{tr}\left((X^\top X)^{-1}\right)\right] + \sigma^2\sum_{i=1}^{n}\mathbb{E}\left[\text{tr}\left((X^\top X)^{-1}M_i\right)\right] \tag{40}$$

$$\leq \frac{\sigma^2 d}{T-d-1} + \sigma^2\sqrt{\mathbb{E}\left[\text{tr}\left((X^\top X)^{-2}\right)\right]}\cdot\sum_{i=1}^{n}\sqrt{\mathbb{E}\left[\text{tr}(M_i)\right]} \tag{41}$$

In our setting, the data is incrementally modified over iterations and modifications decreases progressively. This behavior can be modeled by the sum of a geometric series, where the amount of modified data decreases by a fixed ratio η with each iteration. Then, we assume the editing operation as $||M_i|| = ||M_{i-1}||\eta$, for $i = 1, 2, \ldots, n$. Therefore, the test error for data editing can be bounded:

$$E_{test}(\hat{w}_{n+1}) \leq \frac{\sigma^2 d}{T-d-1} + \sigma^2\sqrt{\mathbb{E}\left[\text{tr}\left((X^\top X)^{-2}\right)\right]}\cdot\frac{\sqrt{\mathbb{E}\left[\text{tr}(M_1)\right]}}{1-\eta} \tag{42}$$

Additionally, since M_i is not full-rank, as seen from Eq. 36, we can apply a more relaxed and simplified bound, as follows:

$$E_{test}(\hat{w}_{n+1}) \leq \frac{2\sigma^2 d}{T-d-1} \tag{43}$$

Thus, the above derivation satisfies the Theorem 4.3.

Algorithm 1 Token-level Editing

Input: Sequence of tokens $\mathbf{x} = (x_1, \ldots, x_t)$, prior distribution P, probability threshold p
Output: Edited sequence $\mathbf{x'} = (x'_1, \ldots, x'_t)$
for each token x_i in sequence **x do**
  Compute conditional probability $P(x_i \mid x_1, \ldots, x_{i-1})$
  if $P(x_i \mid x_1, \ldots, x_{i-1}) \geq p$ **then**
    Resample token $\tilde{x}_i$ from prior distribution
    Set $x'_i \leftarrow \tilde{x}_i$
  else
    Set $x'_i \leftarrow x_i$
  end if
end for
Return: Edited sequence $\mathbf{x'} = (x'_1, \ldots, x'_t)$

Table 9. Comparison of human and synthetic data performance across downstream tasks in (Maini et al., 2024), based on training with GPT-2.

	TruthfulQA	LogiQA	Wino.	PIQA	ARC-E	BoolQ	OBQA	Avg
Human Data	32.68	23.03	51.3	64.42	44.4	60.98	15	41.69
25% Synthetic Data	27.91	21.37	50.12	63.93	43.94	62.29	15.4	40.71
50% Synthetic Data	30.84	22.58	52.41	63.33	44.02	62.14	16	41.62
75% Synthetic Data	29.5	22.65	49.8	63.44	44.53	61.56	17.2	41.24
Synthetic Data	28.89	22.58	49.72	63	46.3	54.53	16.8	40.26

B. More Related Work

Phi-1/2 (Gunasekar et al., 2023) demonstrate that the synthetic data can boost training efficiency and performance compared to raw data in language model pre-training. Furthermore, Feng et al. (2024) introduce a verifier to filter synthetic samples, theoretically avoiding model collapse. Liu et al. (2024); Tan et al. (2024) highlight that synthetic data will play a crucial role in the development of AI. For example, synthetic data can be used to construct highly specialized datasets, enhancing the performance of downstream tasks. Trinh et al. (2024) utilize synthetic math data to train a 125M language model, which successfully solved 25 out of 30 selected problems from the International Mathematical Olympiad (IMO) problem set. Zhang et al. (2024) develop a biomedical instruction dataset that was used to train specialized bio-models, enabling them to excel in answering questions related to medical exams and clinical scenarios. Eldan & Li (2023) introduce a novel synthetic dataset and evaluation paradigm that enables small language models to generate coherent, diverse, and grammatically sound stories. As outlined above, in the post-training stages of LLMs, synthetic data enhances downstream task performance and aligns foundation models with humans. And Maini et al. (2024) propose rephrasing the pre-training data into a Wikipedia or Q/A style to achieve better alignment with downstream tasks. Synthetic data is a powerful tool for training. Our approach is also based on synthetic data methods. Instead of sampling data solely based on this prior, we modify the data using the prior as a guide.

Bertrand et al. (2023) develop a rigorous framework to demonstrate the importance of real data in maintaining the stability of iterative training. Ferbach et al. (2024) theoretically demonstrate that the impact of data curation can be formalized as an implicit preference optimization mechanism. Kazdan et al. (2025) reveal the detailed training dynamics of model collapse under three different training workflows. Of course, there are also some remarkable studies that successfully used synthetic data. Wang et al. (2022) propose the Self-Instruct data generation framework, enhancing instruction-following capabilities. Ulmer et al. (2024) employ the self-talk method to generate high-quality data. ReST (Gulcehre et al., 2023) uses a policy model to generate datasets and then employs offline RL to fine-tune LLMs on generated datasets. Singh et al. (2023) demonstrate that self-training with binary feedback filtering can reduce reliance on real data. Alemohammad et al. (2023) demonstrate that without enough fresh real images, future generative models will gradually decline. Briesch et al. (2023) illustrates that real data in the iterative training process can slow the decline of LLMs, but cannot fully prevent it. Martínez et al. (2023) shows that the quality and diversity of generated images degrade over time.

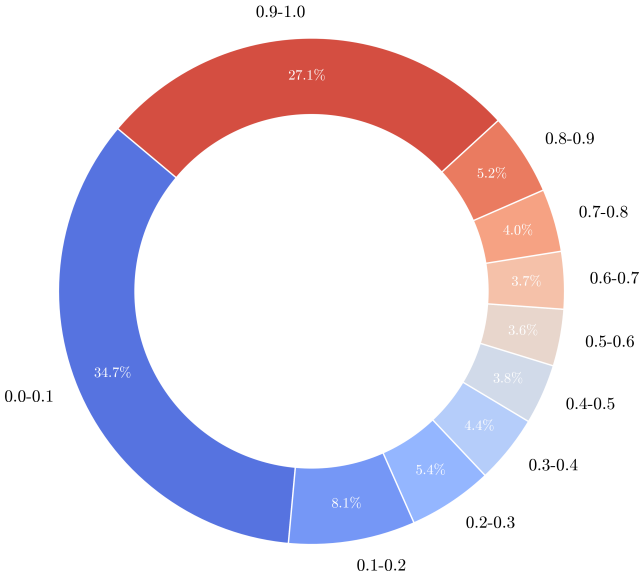


Figure 6. Token distribution across different probability ranges in BioMed dataset.

C. More Discussion of Main Results

As shown in Table 3, our method shows consistent improvements over the source data across OLMo-1B and LLaMA-3-8B. For instance, in the Biomedicine domain, the average score for OLMo-1B increased from 36.63 to 40.89 with ToEdit, while LLaMA-3-8B saw an increase from 54.13 to 56.48. Table 2 further supports the effectiveness of our approach in pre-training. The average performance of OLMo-1B increases from 32.75 to 33.11, reflecting improved generalization capabilities. While the improvement is modest, the consistent trend across tasks like PIQA, BoolQ, and ARC-c highlights the broader applicability of our method. As for SFT results in Table 4, using both the original and edited data, the results indicate a small but consistent improvement. Specifically, ToEdit improves original FLAN v2, with average performance increasing from 70.18 to 70.65. For Natural Instructions, the average performance of LLaMA-3-8B improves from 69.34 to 69.70, with gains in tasks like Winogrande and SIQA. These improvements demonstrate the adaptability of our method to instruction-tuning tasks. For code-related tasks, the improvements demonstrate better reasoning and code comprehension.

D. Comparison with Pure Synthetic Data and Reformat Methods

Definition and Characteristics of Synthetic Data Synthetic data (D_s) can be categorized based on its relationship with the distributions of a language model (P_{LM}) and human-produced data (P_{data}) during the generation process, quantified as $d = \text{KL}(\cdot||P_{\text{data}})$:

$$D_s = \begin{cases} D_s^{\text{pure}} \sim P_{\text{LM}}, & \text{if } \text{KL}(P_{\text{LM}}||P_{\text{data}}) > \epsilon, \\ D_s^{\text{semi}} \sim P_{\text{semi}}, & \text{if } \text{KL}(P_{\text{semi}}||P_{\text{data}}) \leq \epsilon. \end{cases} \tag{44}$$

where **Pure Synthetic Data D_s^{pure}:** Generated entirely from the language model ($D_s^{\text{pure}} \sim P_{\text{LM}}$), with a KL divergence $\text{KL}(P_{\text{LM}}\|P_{\text{data}})$ exceeding a threshold ϵ. This implies a significant deviation of the language model's distribution from the human-produced data distribution. **Semi-Synthetic Data D_s^{semi}:** Derived from limited modifications to human-produced data (P_{data}), ensuring that the resulting distribution (P_{semi}) has a KL divergence $\text{KL}(P_{\text{semi}}||P_{\text{data}})$ bounded by ϵ. This reflects a closer alignment of semi-synthetic data with human-produced data.

From the generation process, pure synthetic data D_s^{pure}: This data is induced by a language model through prompts and does not modify human-produced data, resulting in low overlap content with human-produced data. For example, Cosmopedia (Ben Allal et al., 2024) expands human-produced data and generates data without human-produced data. Semi-Synthetic Data D_s^{semi}: This data is generated by directly modifying human-produced data, such as paraphrasing or token-level editing. It derives from transformations of human-produced data. For example, WRAP (Maini et al., 2024) generates paraphrases of human-produced data. ToEdit (ours) performs token editing on human-produced data.

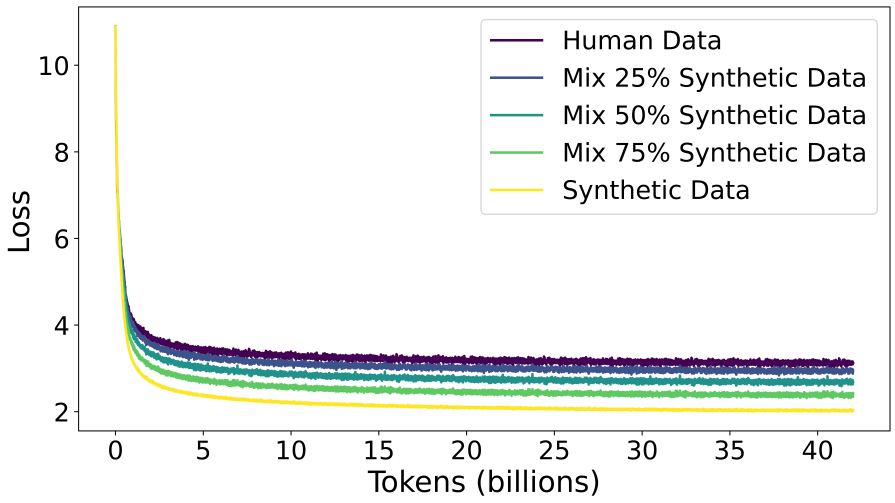


Figure 7. Pre-training loss of GPT-2 Small (124M) on human (Dolma (Soldaini et al., 2024)) and synthetic (Cosmopedia (Ben Allal et al., 2024)) data. As the proportion of synthetic data increases, the model's loss decreases.

Specifically, both *Rephrasing the Web* (Maini et al., 2024) and our token-level editing aim to refine data while preserving the original distribution, producing semi-synthetic data. In contrast, purely synthetic data in Cosmopedia lacks the long-tail distribution and overly concentrates on n-gram features. Ultimately, semi-synthetic data enhances training performance, whereas purely synthetic data results in model collapse. Moreover, replacing a whole real sample with synthetic data can damage the performance.

The primary distinction between Cosmopedia, Rephrasing the Web (Maini et al., 2024), and our approach lies in how much of the original human data distribution is preserved. We provide a detailed comparison of these synthetic methods in Table 10.

Table 10. Comparison of different synthetic data methods.

Method	Data Type	Approach	Result
Cosmopedia (Ben Allal et al., 2024)	Pure synthetic	Using a prompt to induce data from LLMs.	Reveal non-iterative model collapse.
Rephrasing the Web (Maini et al., 2024)	Semi-synthetic	Using a prompt and source content to guide LLMs to reformat source content.	Improve training performance.
ToEdit (Ours)	Semi-synthetic	Using the distribution of source content estimated by LLMs (single forward pass) to replace tokens.	Improve training performance.

E. More Results of Human and Synthetic Data Mixture Training

We provide more training results for the human and synthetic data mixture. The main results and analysis can be found in Sec 3.1. Except for GPT-2 pre-training, we also use the OLMo models (Groeneveld et al., 2024) for further experiments.

As shown in Figure 8, the training loss continues to decrease as the amount of synthetic data increases, which is consistent with GPT-2 pre-training in Figure 2. More synthetic data can lead to better fitting. However, a lower loss does not necessarily mean a better model. As illustrated in Figure 2B and 9, models that fits better perform worse in real world tasks.

Furthermore we follow (Maini et al., 2024) to conduct more experiments including PPL results on 22 validation sets of Pile (Gao et al., 2020a) and general understanding tasks. The additional results in Table 9, 11 and 1 are consistent with our findings. Specifically, the PPL increases as the proportion of purely synthetic data grows, while the performance on downstream tasks similarly exhibits a gradual decline with the increase in synthetic data.

Table 11. Comparison of human and synthetic data performance across downstream tasks in (Maini et al., 2024), based on training with OLMo-237M. ± indicates the standard error.

	TruthfulQA	LogiQA	Wino.	PIQA	ARC-E	OBQA	Avg
Human Data	26.81 ± 1.550	21.06 ± 1.028	52.01 ± 1.404	56.69 ± 1.156	31.73 ± 0.9550	13.80 ± 1.543	33.68
25% Synthetic Data	26.44 ± 1.543	21.25 ± 1.032	52.64 ± 1.403	57.02 ± 1.155	31.78 ± 0.9552	12.40 ± 1.475	33.59
50% Synthetic Data	25.95 ± 1.534	20.04 ± 1.099	52.25 ± 1.408	56.64 ± 1.126	31.82 ± 0.9557	12.80 ± 1.495	33.25
75% Synthetic Data	25.34 ± 1.522	20.87 ± 1.025	50.43 ± 1.405	55.60 ± 1.159	32.74 ± 0.9629	12.00 ± 1.454	32.83
Synthetic Data	23.01 ± 1.473	20.29 ± 1.014	49.33 ± 1.405	55.93 ± 1.158	33.33 ± 0.9673	14.20 ± 1.562	32.68

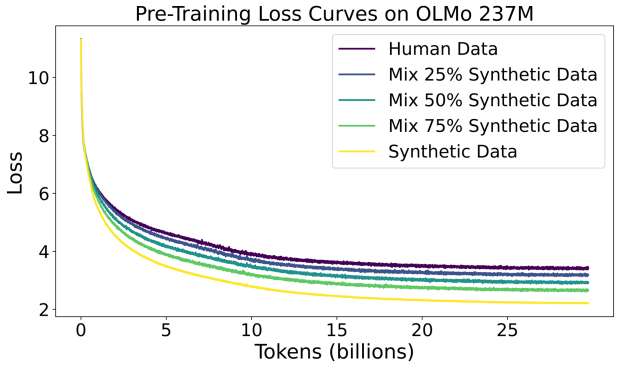


Figure 8. OLMo-237M pretraining with mixed human and synthetic data proportions. We pretrain the OLMo-237M model using a mixture of human data (Dolma (Soldaini et al., 2024)) and synthetic data (Cosmopedia (Ben Allal et al., 2024)).

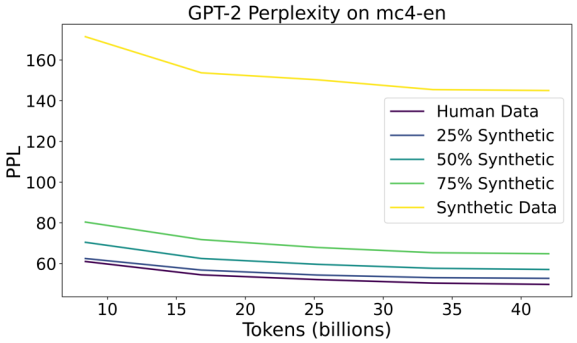


Figure 9. GPT-2 perplexity (PPL) on validation sets, trained from scratch.

F. Experiment Settings

In this section, we describe our experiments settings in detail.

F.1. Training

Pre-training We utilized both GPT-2 and OLMo models. The pre-training datasets included Dolma, representing real data, and Cosmopedia, representing synthetic data. For GPT-2, we employed the official FSDP (Fully Sharded Data Parallel) framework provided by Torch for training. For OLMo[3], we used the official open-source computational code, which also incorporates the FSDP framework alongside Flash Attention for acceleration.

Continual Pre-training We follow (Cheng et al., 2024b) to conduct continual pre-training on biomedicine, finance, and math domains. Specifically, PubMed Abstracts from the Pile are utilized as the pre-training corpora for the biomedicine domain. For the finance domain, financial news data covering over 7,000 stocks from May 2022 to May 2023 is collected using the FinGPT framework. We continue pre-training OLMo-1B and LLaMA-3-8B on each domain. For implementation, we utilized the official training framework for OLMo-1B, leveraging Fully Sharded Data Parallel (FSDP) for continual pre-training. For LLaMA, we adopted the LLaMA-Factory framework to carry out the continual pre-training process. Our experiments was primarily conducted on OLMo-1B and Llama-3-8B models, with Llama-3-8B utilizing LoRA (Low-Rank Adaptation) for parameter-efficient fine-tuning. The data and evaluation are given in this repo[4]. We conducted the continual pre-training on a total of 1B tokens.

Supervised Fine-tuning We used the Llama-Factory (Zheng et al., 2024) framework to fine-tune Llama-3-8B. As for general instruction tuning tasks, we adopt instruction tuning datasets from (Xia et al., 2024) [5], including CoT (Wei et al., 2022) , FLAN v2 (Longpre et al., 2023), and Open Assistant 1 (Kopf et al., 2023). As for code-related reasoning tasks,

[3] https://github.com/allenai/OLMo
[4] https://github.com/microsoft/LMOps/tree/main/adaptllm
[5] https://huggingface.co/datasets/princeton-nlp/less_data

Table 12. Case Study.

Before (source)	After (edited)	Changes
Construct a function using PHP language that applies lexical analysis on a provided text string to analyze the individual, non-repeated words elements present.	Construct a function using PHP language that applies lexical analysis on a provided text string to quantify unique words.	"analyze" → "quantify"
Test with provided string, `$str = 'Greetings, Planet Earth!'`.	Test with provided string, `$str = 'Greetings, Planet Earth!'`.	No changes.
Implements `wordCount` to remove punctuation, convert text to lowercase, split into words, and count unique words.	Implements `wordCount` to remove punctuation, convert text to lowercase, split into words, and calculate unique words.	"count" → "calculate"
Returns `{'greetings': 1, 'planet': 1, 'earth': 1}`.	Returns `{'greetings': 1, 'planet': 1, 'earth': 1}`.	No changes.

we utilize OSS-Instruct-75K [6] and Evol-Instruct-110K [7]. These datasets provide sufficient diversity for verification on fine-tuning. We apply LoRA (Hu et al., 2021) to Llama-3-8B experiments.

F.2. Evaluation

Pre-training We use PPL and downstream tasks to conduct analysis and performance test. As for PPL, it stands for perplexity, a commonly used metric in NLP to evaluate the quality of language models. It measures how well a probabilistic model predicts a given dataset, with lower values indicating better performance. Formally, the perplexity of a language model is calculated as:

$$\text{PPL} = 2^{-\frac{1}{N}\sum_{i=1}^{N}\log_2 P(x_i)}$$

Alternatively, it can also be expressed as:

$$\text{PPL} = \exp\left(-\frac{1}{N}\sum_{i=1}^{N}\log P(x_i)\right)$$

Where N is the number of tokens in the dataset, and $P(x_i)$ is the predicted probability of the i-th token. Perplexity essentially represents the exponential of the average negative log-likelihood of the predicted tokens, indicating how "perplexed" the model is when making predictions.

As for downstream tasks, we use general understanding tasks in (Maini et al., 2024) to analyze model collapse in Table 9 and general test tasks in (Cheng et al., 2024a) to test our methods in Table 2. All downstream tasks we used can be found in (Gao et al., 2024)[8].

Continual Pre-training We use the test data and code in (Cheng et al., 2024b)[9] to test domain specific task performance after CPT.

Supervised Fine-tuning We utilize the general downstream tasks from (Cheng et al., 2024a) to evaluate instruction-tuning performance and reasoning tasks to assess reasoning capabilities. All downstream tasks we used can be found in (Gao et al.,

[6] https://huggingface.co/datasets/ise-uiuc/Magicoder-OSS-Instruct-75K
[7] https://huggingface.co/datasets/ise-uiuc/Magicoder-Evol-Instruct-110K
[8] https://github.com/EleutherAI/lm-evaluation-harness
[9] https://github.com/microsoft/LMOps/tree/main/adaptllm

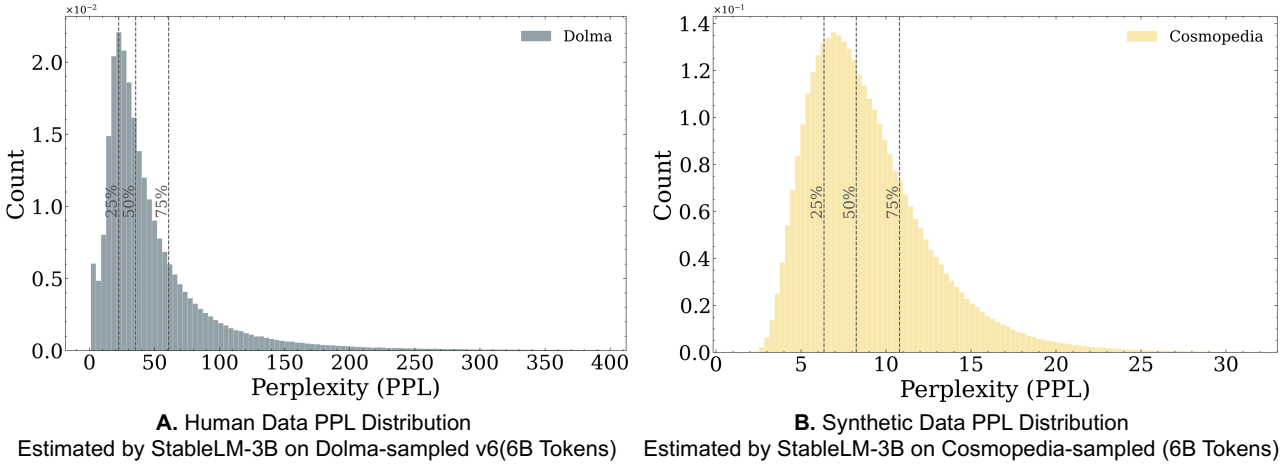


Figure 10. PPL distribution of human and synthetic data estimated by StabLM-Zephyr-3B. This indicates that different prior distributions yielded the same result, which is consistent with Figure 3. The synthetic data lacks a long tail and is concentrated within a narrow portion of the distribution.

2024)[10].

Table 13. PPL results of GPT-2 124M pre-training on mixture of human and synthetic data.

Synthetic Data Ratio	25%					50%					75%				
Tokens Size	8.4B	16.8B	25.2B	33.6B	42B	8.4B	16.8B	25.2B	33.6B	42B	8.4B	16.8B	25.2B	33.6B	42B
Epochs	1	2	3	4	5	1	2	3	4	5	1	2	3	4	5
Wikitext-103	45.97	39.87	37.65	36.91	36.32	50.29	43.15	40.46	39.43	38.65	58.66	48.75	45.20	43.42	42.95
RedPajama	42.28	37.62	35.72	34.66	34.24	46.89	41.42	39.37	38.21	37.72	55.72	49.26	46.27	44.81	44.30
Falcon-RefinedWeb	56.40	50.62	48.26	47.13	46.66	61.06	54.34	51.72	50.39	49.87	69.32	61.50	58.28	56.77	56.19
c4-en	48.15	43.14	40.98	39.91	39.41	51.79	46.06	43.90	42.73	42.23	58.60	52.22	49.26	47.87	47.27
mc4-en	62.46	56.80	54.35	53.06	52.71	70.43	62.48	59.61	57.66	57.07	80.37	71.77	67.90	65.31	64.82

Table 14. PPL results of OLMo-237M pretraining on mixture of human and synthetic data.

Synthetic Data Ratio	0%	25%	50%	75%	100%	DSIR (1M)	DSIR (10M)	Edu Classifier (1M)	Edu Classifier (10M)	PPL Filter (1M)	PPL Filter (10M)	Density Sampling (1M)	Density Sampling (10M)
Unique Tokens	8.4B	8.4B	8.4B	8.4B	8.4B	0.6B	8.4B	0.75B	7.4B	0.97B	9B	0.6B	7.1B
Training Tokens	8.4B	8.4B	8.4B	8.4B	8.4B	8.4B	8.4B	10.5B	7.4B	13.68B	9B	8.9B	7.1B
Epochs	1	1	1	1	1	14	1	14	1	14	1	14	1
Wikitext-103	187.36	185.5	260.08	367.46	1605.73	1309.53	1757.03	1111.29	1612.95	738.36	1193.25	1188.40	1753.89
RedPajama	175.38	183.93	236.33	301.09	907.91	649.36	916.51	811.14	1104.75	376.36	645.82	789.67	896.18
Falcon-RefinedWeb	165.17	166.69	199.68	245.15	523.93	573.61	510.96	522.97	612.72	344.82	449.86	501.99	560.92
c4-en	123.88	127.68	147.69	174.48	410.19	457.96	404.63	415.88	487.97	286.95	367.44	414.55	457.71
mc4-en	208.91	208.94	263.35	324.91	800.40	861.01	823.12	769.86	955.70	476.81	662.00	740.75	844.53
M2D2-Wiki	88.24	87.34	107.77	114.19	189.06	234.45	183.17	161.58	206.45	130.43	162.08	167.20	205.50
M2D2-S2ORC	86.15	81.53	97.61	100.64	204.22	170.78	496.40	145.27	201.52	117.44	163.38	131.22	192.97

G. Discussion

G.1. Non-Iterative vs Iterative Model Collapse.

We define *non-iterative model collapse* as the performance degradation caused by directly mixing general synthetic data with human-produced data, without iterative training. Theoretically, without additional regularization constraints to guide data generation, the variance of the model-generated data gradually decreases during this process. The diversity of the generated data diminishes over time, ultimately leading to the collapse of the model itself.

The difference between the two lies in their scope. Non-iterative model collapse is not confined to training on self-generated data, allowing it to uncover broader properties of synthetic data. For instance, in our experiments, we train GPT-2 on the Cosmopedia dataset in a single generation, which was generated by `Mixtral-8x7B-Instruct-v0.1`. In contrast,

[10] https://github.com/EleutherAI/lm-evaluation-harness

iterative model collapse focuses on training the model over multiple generations using self-generated data.

Furthermore, the non-iterative model collapse emphasizes the gap between human data and general purely synthetic data, particularly regarding distributional properties and n-gram features. In contrast, the iterative model collapse illustrates the iterative evolution of the model, resembling a self-play process. This process illustrates the gradual evolution of self-generated data. It does not involve an analysis of the differences in nature between self-generated and human-produced data. They both ultimately lead to model collapse, driven by the same underlying cause—synthetic data, though they investigate different aspects of synthetic data. We often face that training a model on a mixture of human and synthetic data, where the synthetic data is not generated by the model itself, and its exact origin may be unknown.

G.2. Why Does the Observed Probability Distribution Exhibit Filtering Potential?

From the perspective of information theory, we can analyze the filtering potential of the U-shape distribution as follows: We utilize the U-shape distribution in Figure 5 to re-sample tokens in the high-probability region, to adjust the U-shaped distribution toward a uniform distribution. By doing so, we can maximize the information entropy. According to information theory, maximizing information entropy is achieved when the distribution is uniform.

Lemma 1: Let X be a discrete random variable with n possible outcomes. If the probability of each outcome is uniform, i.e., $P(x_i) = \frac{1}{n}$ for all $i \in \{1, 2, \ldots, n\}$, the Shannon entropy is maximized, given by:

$$H(X) = -\sum_{i=1}^{n} \frac{1}{n} \log \frac{1}{n} = \log n. \tag{45}$$

This represents the maximum uncertainty achievable, implying that the dataset carries the maximum possible information content. Thus, the uniform distribution, which assigns equal probability to all outcomes, possesses the maximum information entropy. To leverage this property, we utilize the U-shape distribution to re-sample tokens in the high-probability region, adjusting the U-shaped distribution toward a uniform distribution. By doing so, we can maximize the information entropy.

From the perspective of language model learning, our method emphasizes the importance of poorly learned data. Specifically, we resample easy tokens and encourage the model to focus on learning more challenging ones. Our method can enhance the learning of underrepresented data by resampling high-probability tokens.

G.3. Gradual Decline in Editing

Table 15. Percentage of tokens requiring edits in the Natural-Instructions dataset. The total number of tokens is 4,671,834.

	Gen 1 (source)	**Gen 2**	**Gen 3**
Tokens ($p > 0.99$)	584,103	549,519	517,433
Percentage	12.5%	11.76%	11.08%

We present the percentage statistics of edited tokens in Table 15 and performance in an iterative process in Table 16, demonstrating that the edited tokens indeed exhibit a progressive decrease. Specifically, We observe that the percentage of edited tokens (above the threshold $p > 0.99$) decreases as the generation number increases. Theoretically, this is a process of distribution shifting. When tokens ($p > 0.99$) are resampled, randomness is introduced. The sampling process can select tokens with lower probabilities. Then, tokens ($p > 0.99$) is replaced, leading to a reduction of edited tokens in subsequent generations. The Table 15 provides empirical evidence for this pattern of decay.

Table 16. Performance in an iterative process on Instruction tuning data.

	PIQA	BoolQ	HS	SIQA	WG	Avg
Gen 0	79.87	81.28	59.72	49.69	74.51	**69.01**
Gen 1	80.25	81.16	59.74	50.56	74.59	**69.26**
Gen 2	80.14	82.69	59.82	50.51	73.80	**69.39**

G.4. What is Coverage Narrowing?

'coverage narrowing' refers to a phenomenon in which the distribution of synthetic data covers a significantly narrower range of values compared to human data, even when the data sizes are identical. For instance, as shown in Figure 3, the PPL range of synthetic data is limited to $[0, 14]$, whereas the PPL range of human data extends from $[0, 100+]$. Despite this disparity, the overall coverage, represented by the area under the distribution curves, remains the same. This significant distribution gap is what we define as 'coverage narrowing.'

G.5. How Does the DSIR Work?

DSIR (Xie et al., 2023) works by estimating importance weights for each data sample to measure its relevance to the target distribution. This involves three main steps: first, we leverage n-gram models to estimate two distributions of human and synthetic data, q_{feat} and p_{feat}, which represent the target and raw distributions, respectively. We use them to compute the likelihood ratio for each sample. Next, we calculate the importance weight for each sample z_i as $w_i = \frac{\hat{p}_{\text{feat}}(z_i)}{\hat{q}_{\text{feat}}(z_i)}$. The weight w_i quantifies how well the sample aligns with the target distribution. Finally, we perform importance-weighted sampling without replacement to select examples, ensuring that the selected data is more representative of the target distribution.

We use DSIR in our data analysis as it allows for principled and computationally efficient selection of synthetic data points that align with the target distribution. Moreover, the importance weight also reflects the alignment between the n-gram features of synthetic data and human data. Using DSIR, we can analyze the differences between synthetic and human data across n-gram feature distributions and data matching. As shown in Figure 11, it is challenging to select synthetic data that matches human data characteristics under the significant distribution difference. To obtain high-quality synthetic data, it is essential to focus on improving the data synthesis methods.

G.6. Non-autoregressive Token Replacement May Compromise Text Coherence.

When designing data synthesis algorithms, we must balance synthesis efficiency and effectiveness, considering both autoregressive and non-autoregressive approaches. Autoregressive methods leverage the inherent capabilities of language models to generate coherent text sequentially. In contrast, non-autoregressive methods resample individual tokens based on their probability distributions. Since data synthesis is a prerequisite for model training, we aim to ensure that the cost of data synthesis does not exceed the cost of training itself.

Specifically, our ToEdit modifies data using the probability distribution in a single forward pass. For instance, if the generated sequence length is 1024, the computational cost of autoregressive methods would be 1024 times higher than ours. This efficiency advantage is why our method can run effectively on GPUs like the 3090 or 4090 series.

However, this efficiency may come at the cost of coherence, as resampled tokens may not fit seamlessly into a given sentence. To address this issue, we introduce a hyperparameter, resampling probability p, to control the resampling threshold. We perform sampling in high-probability regions, focusing on tokens that are relatively easier to predict. We manually verify and tune on a small validation set before applying it across all experiments. In our experiments, we set $p = 0.99$.

Additionally, we supplement more experiments and discussion about hyper-parameter p. As Table 5 shows, different values of p influence BioMed performance, leading to fluctuations in data quality. Table 6 presents the distribution percentages of the token probabilities across different value ranges. We need to refine the data while primarily preserving the source distribution. A larger p indicates fewer tokens will be resampled, while a smaller p results in more tokens being resampled. Balancing performance and the preservation of data distribution, we set $p = 0.99$ as the threshold for our experiments.

G.7. Must We Assume the Data is 100% Human-authored?

We do not need to assume that the data is 100% human authored; In experimental settings, some datasets used in our experiments include partially synthetic data:

- Datasets used in continual pretraining (e.g., Biomed, Finance) include partially synthetic data, which has been reformatted into a reading comprehension structure (Cheng et al., 2024b).
- OSS-Instruct-75K and Evol-Instruct-110K also contain samples synthesized by ChatGPT.

In the theoretical framework, synthetic data is generated iteratively through an n-generation process. (1) If the starting point is a real distribution, our method preserves most of the initial distribution to generate higher-quality data. (2) If the starting

Table 17. PPL results of GPT-2 124M pretraining on pure Human or Synthetic data.

Data Type	Human Data (Dolma)					Synthetic Data (Cosmopedia)				
Tokens Size	8.4B	16.8B	25.2B	33.6B	42B	8.4B	16.8B	25.2B	33.6B	42B
Epochs	1	2	3	4	5	1	2	3	4	5
Wikitext-103	43.62	38.57	36.11	34.89	34.55	169.38	147.73	135.23	131.78	128.05
RedPajama	40.18	35.84	33.97	32.74	32.34	116.37	103.25	99.27	96.81	96.03
Falcon-RefinedWeb	54.85	49.10	46.93	45.43	44.90	146.97	132.60	127.68	124.32	122.69
c4-en	45.87	41.00	39.10	37.95	37.56	128.25	114.41	109.73	107.53	106.55
mc4-en	61.00	54.44	52.11	50.38	49.74	171.44	153.70	150.28	145.44	144.99

point is a mixture of synthetic and real data, the modifications are minimal, ensuring the original distribution remains largely unaffected. Therefore, applying our method in any generation i, we can further avoid issues, such as reduced variance and diminished diversity, which are key factors contributing to model collapse.

In other words, whether the current data is fully real or a mix of real and synthetic, using it as anchor data to synthesize data, our method builds upon the current data distribution to achieve improvements, rather than causing model collapse.

In summary, we aim to improve the data synthesis method, specifically focusing on how to obtain higher-quality data from the existing datasets. We do not need to assume that the data at hand is 100% human-generated. Our algorithm is designed to minimize excessive distribution truncation of the original data.

G.8. Can ToEdit Help with Already Strongly Collapsed Data or is a Minimum Quality of the Data Necessary?

The ToEdit algorithm was initially designed to preserve the long-tail distribution during the data generation process, thereby avoiding model collapse. For already collapsed data, the variance is typically very small, and enhancing diversity is crucial. We can also adjust the threshold to introduce more randomness in data. Through this operation, we can inject randomness into the collapsed data. However, this is a theoretical scenario, and as we know, data situations are highly complex. In practice, there will be many more challenges to address.

G.9. Could We Generate Tons of Data Using LLMs and Select Only the Long-Tail Ones?

For now, generating long-tail samples is currently difficult for language models. The reason lies in the sampling strategy of LLMs. Current LLMs adopt top-p, top-k or other sampling strategy for better performance. However, these sampling strategy will lead to cut-off distribution. When the data synthesizing scale up, this drawback will finally scaling law cut-off on synthetic data. However, human corpus data follows a Zipf distribution. The truncated output distribution causes the LLMs to nearly fail to sample long-tail samples. In other words, it is currently difficult to induce long-tail samples from LLMs that are as diverse as human data.

On the other hand, if we force the language model to generate long-tail samples, these may contain both noisy and high-information samples, which are like two sides of a coin, both distributed in the long tail of the data. This necessitates further filtering of the high-information samples. Unfortunately, such samples are challenging to automatically identify in practice and may require extensive human annotation.

H. Potential Applications and Future Work

Based on the above discussion, our approach can be applied to optimize the current data, even if it is a mixture of real and synthetic data. From the findings and proposed method in our paper, we can influence future research in the following aspects:

Potential applications of our work: (1) Data optimizations. We can quickly modify and optimize the current data, using a trained language model with a single forward pass. (2) Regularization in the data synthesizing process. When synthetic data becomes excessive, we can introduce real data as an anchor to balance the issues of excessive homogeneity and tail distribution cut-off in synthetic data, thereby preventing mode collapse.

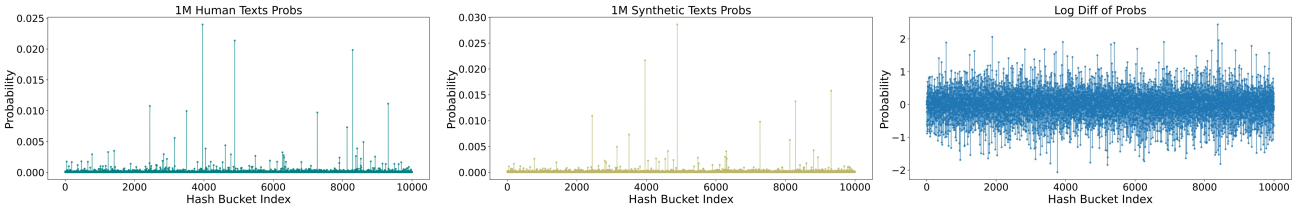


Figure 11. Uni/Bi-gram feature distribution across 10,000 hash buckets.

Lessons from our work: The key to improving the quality of synthetic data lies in balancing long-tail distribution preservation and optimizing synthetic data approaches. In other words, we should focus on two questions: how to generate more informative synthetic data and how to integrate it with real data effectively. Building on this foundation, future improvements can focus on two aspects: first, obtaining more information gain by designing more efficient generation mechanisms to inject valuable information into the synthetic data; and second, optimizing methods to reduce noise during the synthesis process. This approach ensures that synthetic data retains its authenticity while enhancing its utility in practical tasks.

Extended related applications A broader range of recent works across domains also explore synthetic data generation and usage for diverse applications (Bai et al., 2025; 2024; 2023; Jia et al., 2024; Yang et al., 2023; Li et al., 2020; Niu et al., 2024; Zhang et al., 2023; Li et al., 2024; Pu et al., 2024; Li et al., 2023; Zhu et al., 2024a;b; 2022).

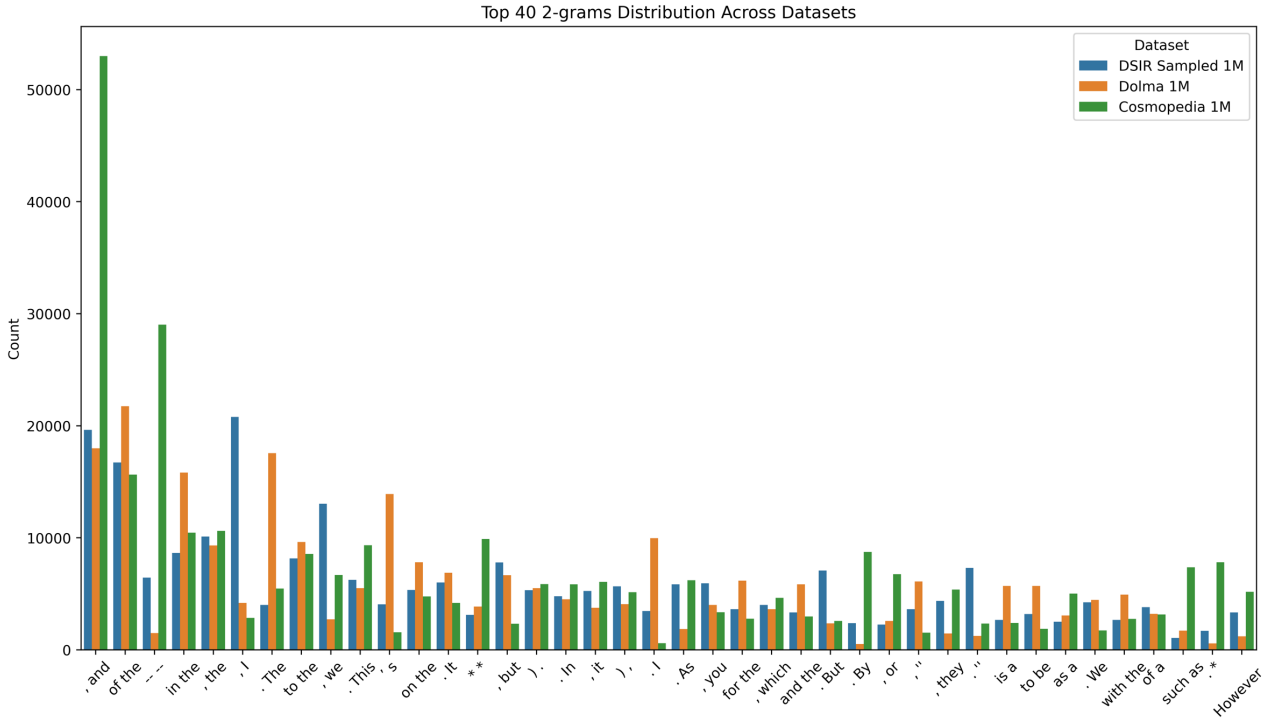


Figure 12. The top 40 bi-grams from separately sampled 1M subsets of Dolma, Cosmopedia, and DSIR-selected datasets.

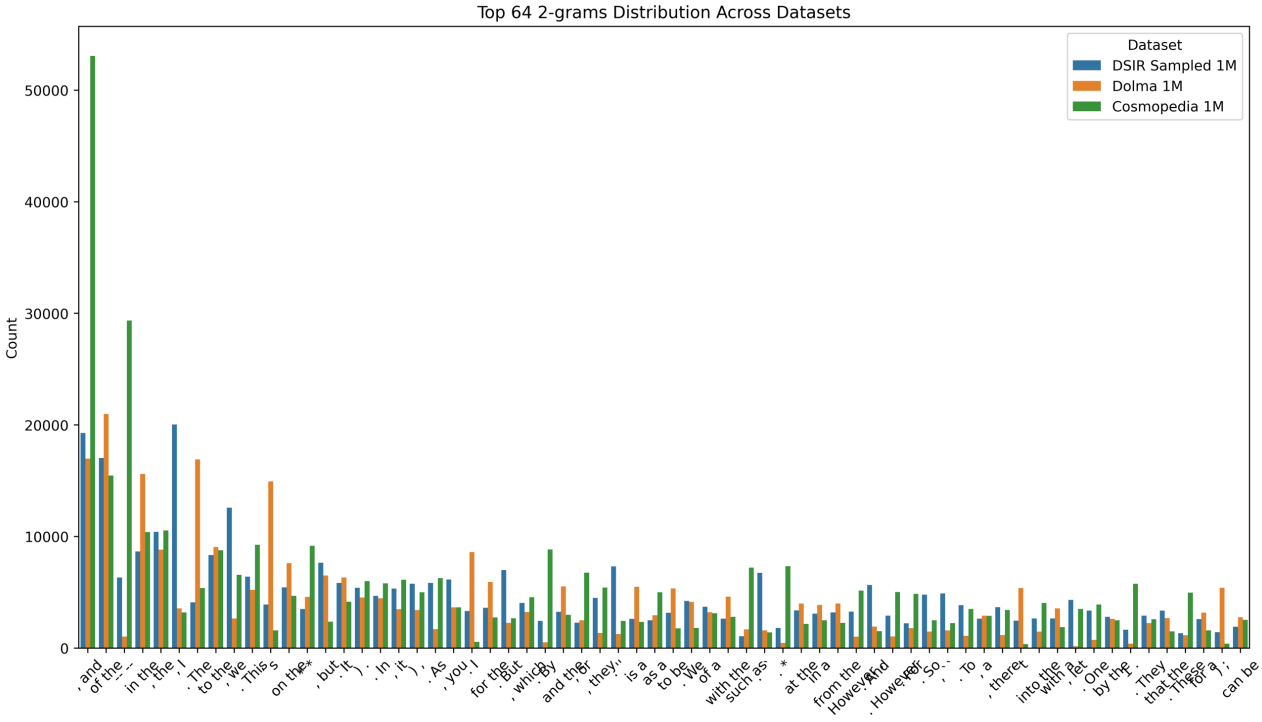


Figure 13. The top 64 bi-grams from separately sampled 1M subsets of Dolma, Cosmopedia, and DSIR-selected datasets.

Table 18. Dolma dataset statistics (v1.6), quoted from source (Soldaini et al., 2024).

Source	Doc Type	UTF-8 bytes (GB)	Documents (millions)	Unicode words (billions)	Llama tokens (billions)
Common Crawl	web pages	9,022	3,370	1,775	2,281
The Stack	code	1,043	210	260	411
C4	web pages	790	364	153	198
Reddit	social media	339	377	72	89
PeS2o	STEM papers	268	38.8	50	70
Project Gutenberg	books	20.4	0.056	4.0	6.0
Wikipedia, Wikibooks	encyclopedic	16.2	6.2	3.7	4.3
Total		**11,519**	**4,367**	**2,318**	**3,059**

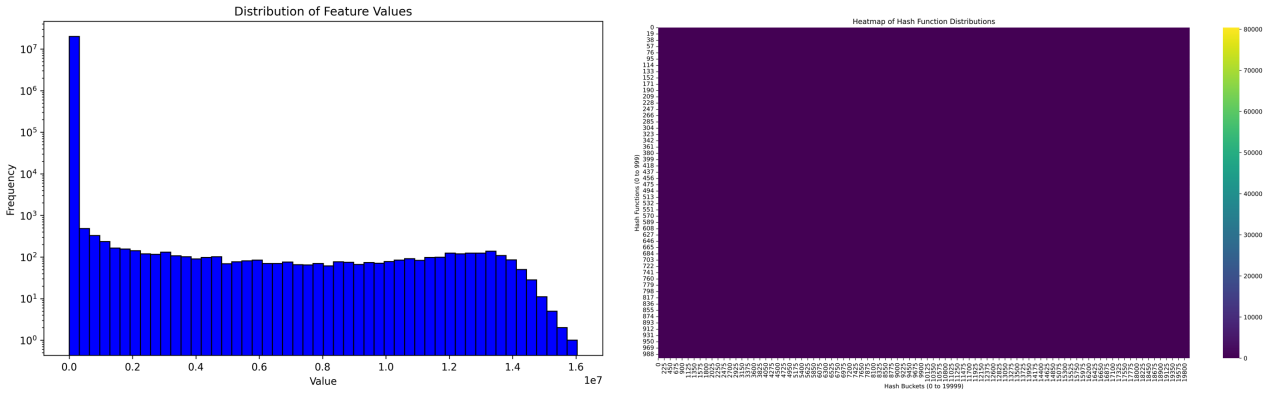


Figure 14. Density sampling response values. This result further confirms the issue of feature collapse in synthetic data.