Demystifying Mixed Outcomes of Self-Training: Pre-training Analyses on Non-Toy LLMs

Yusuke Nakamura[1,2], **Hirokazu Kiyomaru**[2], **Shuhei Kurita**[2]
Chaoran Liu[2], **Daisuke Kawahara**[1,2]
[1]Waseda University
[2]Research and Development Center for LLMs, National Institute of Informatics
yusuke69@ruri.waseda.jp, dkw@waseda.jp
{kiyomaru,skurita,cliu}@nii.ac.jp

Abstract

We investigate self-training of large language models (LLMs), in which models are recursively trained on their own generated text. Prior studies have reported conflicting outcomes in this setting: some find evidence of performance gains (i.e., *self-improvement*), while others observe performance degradation (i.e., *model collapse*). To clarify this discrepancy, we use the OLMo-2 models as non-toy LLMs and perform multiple rounds of continual pre-training using self-generated text with varying data synthesis and filtering strategies. Our experiments show that naive self-training does not improve either perplexity on the original pre-training corpus nor downstream task performance, regardless of model size. These results suggest that model collapse observed in naive self-training is inherent to the training procedure, while self-improvement likely owes its success not to the model's autonomous refinement but to human-designed, strategic synthetic pipelines that inject external intelligence.

1 Introduction

Training large language models (LLMs) on LLM-generated synthetic data has become an increasingly popular research paradigm (Gunasekar et al., 2023; Grattafiori et al., 2024; Yang et al., 2025; Qin et al., 2025; DatologyAI et al., 2025). A typical approach uses a sufficiently powerful model to produce synthetic text, which is then used to train a smaller, more efficient model. However, a fundamental question remains: who generates the synthetic data for improving the powerful model itself? This question naturally leads to the idea of self-training, in which a model is recursively trained on its own generated text.

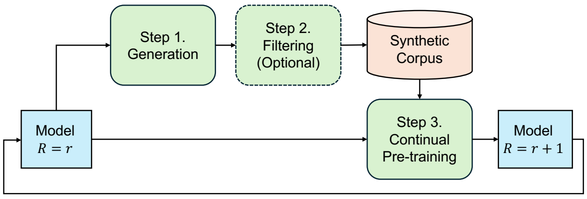


Figure 1: Overview of the continual pre-training pipeline using self-generated text. In Step 1, we sample texts from the current model. In Step 2, we optionally filter the sampled texts based on their perplexities. In Step 3, we continue pre-training the model on its own generated texts. This process is repeated iteratively.

Recent studies on self-training report seemingly contradictory outcomes. Some claim *self-improvement*, where model performance continues to increase through iterative self-training (Huang et al., 2022; Li et al., 2024; Shang et al., 2025). Others warn of *model collapse*, where model performance deteriorates over iterations (Shumailov et al., 2024; Wang et al., 2025a; Drayson et al., 2025; Herel and Mikolov, 2024; Wang et al., 2025b). Understanding the cause of this discrepancy is the focus of this work.

A closer look at prior research reveals that these opposing findings often stem from different experimental designs. Studies reporting self-improvement typically employ strong, non-toy LLMs, adopt strategic data generation pipelines (e.g., crafted prompts, task-aware sampling, and/or quality filtering), and evaluate improvements on downstream tasks. In contrast, studies reporting model collapse usually rely on small toy models, use naive data generation procedures (e.g., unconditional text generation), and evaluate degradation in perplexity on the original pre-training corpus. Such discrepancies obscure whether the success or failure of self-training arises from the initial model quality, the data synthesis pipeline, or the evaluation metrics.

To disentangle these factors, we conduct a systematic investigation using non-toy LLMs while deliberately adopting a simple data synthesis pipeline. Specifically, we perform multiple rounds of recursive, continual pre-training on the OLMo-2 fam-

Findings of the Association for Computational Linguistics: EACL 2026, pages 4107–4113
March 24-29, 2026 ©2026 Association for Computational Linguistics

ily (Team OLMo et al., 2025). We examine the effects of data synthesis strategies and quality filtering methods on both downstream task performance and perplexity on the original pre-training corpus.

Experiments show that, across all conditions, model performance consistently degrades in both metrics, indicating model collapse rather than self-improvement. These results suggest that model collapse observed in naive self-training is not merely due to weak base models or evaluation metric choices, but rather inherent to the training procedure itself. Conversely, studies that report self-improvement likely owe their success not to autonomous model refinement, but to human-designed, strategic synthetic pipelines that inject external intelligence into the loop.

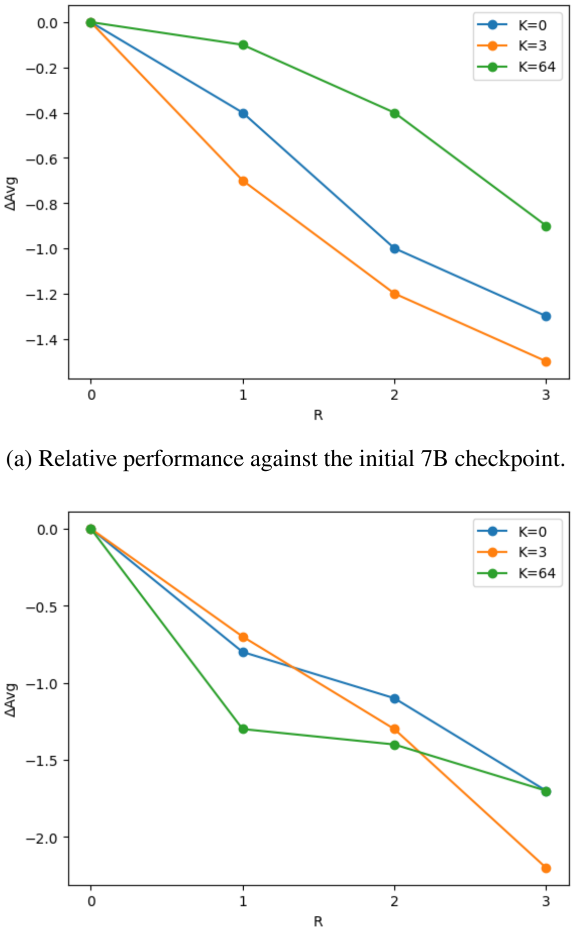
2 Methodology

We study recursive pre-training, in which an LLM generates text that is then used to further pre-train the model. Figure 1 illustrates the pipeline. We start with a capable, non-toy base LLM. In each round $R \in \mathbb{N}$, the current model repeatedly samples a prefix of $K \in \mathbb{Z}_{\geq 0}$ tokens[1] uniformly at random from the pre-training corpus and generates the continuation, until a total of $N \in \mathbb{N}$ synthetic tokens is accumulated. To examine the effect of quality consideration, we optionally apply perplexity-based filtering (Marion et al., 2023), which partitions generated texts into low-, middle-, and high-perplexity tiers. To this end, We use the current model to compute sample-level perplexity for each generated text. When this filtering is enabled, the model over-generates $3N$ tokens, and we retain N tokens according to the perplexity. The resulting synthetic corpus is then used for continual pre-training.

At the end of each round, we evaluate the current model's perplexity on the original pre-training corpus and its performance on downstream tasks.

3 Experiments

We conducted experiments using the OLMo-2 family (Team OLMo et al., 2025), a suite of non-toy LLMs that deliver competitive performance on diverse downstream tasks. Crucially, OLMo-2 provided access to its pre-training corpora, allowing us to assess the perplexity on the original training data. This accessibility was an essential requirement for investigating our research question.



(a) Relative performance against the initial 7B checkpoint.

(b) Relative performance against the initial 13B checkpoint.

Figure 2: Average downstream task score shift across generations.

3.1 Training Setup

We used the OLMo-2 family (Team OLMo et al., 2025) for our experiments. Specifically, we employed the 7B-[2] and 13B-parameter[3] models. These models were trained through a two-stage pre-training process. In Stage-1, the model was pre-trained from scratch on the OLMo-Mix-1124 corpus[4]; in Stage-2, continual pre-training (also known as mid-training) was conducted on the Dolmino-Mix-1124 corpus[5]. Unless otherwise noted, we used the model checkpoints obtained after the Stage-1 pre-training. For each round, we generated one billion tokens (i.e., $N = 10^9$) in to-

[1] $K = 0$ indicates an empty prompt.

[2] https://huggingface.co/allenai/OLMo-2-1124-7B

[3] https://huggingface.co/allenai/OLMo-2-1124-13B

[4] https://huggingface.co/datasets/allenai/olmo-mix-1124

[5] https://huggingface.co/datasets/allenai/dolmino-mix-1124

	MMLU	ARC-C	HSwag	WinoG	NQ	DROP	AVG	PPL
OLMo-2-7B								
$R = 0$ (Base model)	59.8	72.6	81.3	75.8	29.0	40.7	59.9	11.038
$K = 0,\ R = 1$	59.9	72.0	81.1	75.3	28.3	40.2	59.5	11.340
$K = 0,\ R = 2$	59.2	71.7	81.0	74.7	26.9	39.8	58.9	11.654
$K = 0,\ R = 3$	59.4	71.8	80.9	74.3	26.0	38.9	58.6	11.836
$K = 3,\ R = 1$	59.5	72.1	80.9	74.7	27.9	40.4	59.2	11.643
$K = 3,\ R = 2$	58.9	70.7	80.7	73.3	28.4	40.3	58.7	11.790
$K = 3,\ R = 3$	58.6	70.6	80.6	73.4	27.5	40.0	58.4	11.987
$K = 64,\ R = 1$	59.7	73.0	81.8	74.7	29.1	40.2	59.8	11.498
$K = 64,\ R = 2$	59.7	72.8	81.7	74.4	28.5	40.1	59.5	11.791
$K = 64,\ R = 3$	59.1	72.2	81.2	74.1	27.8	39.3	59.0	12.011
OLMo-2-13B								
$R = 0$ (Base model)	63.4	80.2	84.8	79.4	34.5	49.6	65.3	9.607
$K = 0,\ R = 1$	62.9	79.5	84.1	77.7	33.7	49.1	64.5	9.316
$K = 0,\ R = 2$	62.6	78.7	83.9	78.5	32.9	48.7	64.2	9.955
$K = 0,\ R = 3$	61.9	78.6	83.3	78.1	33.0	46.9	63.6	10.242
$K = 3,\ R = 1$	63.5	79.0	84.5	77.9	33.6	49.3	64.6	9.635
$K = 3,\ R = 2$	62.8	78.5	83.7	78.1	33.0	47.8	64.0	9.861
$K = 3,\ R = 3$	61.4	77.1	83.3	77.5	32.9	46.6	63.1	10.072
$K = 64,\ R = 1$	62.8	78.6	83.8	77.7	32.2	48.8	64.0	9.633
$K = 64,\ R = 2$	62.2	78.7	84.0	77.7	32.2	48.7	63.9	9.883
$K = 64,\ R = 3$	62.3	78.5	83.8	77.1	32.0	48.2	63.6	10.116

Table 1: Downstream performance and perplexity on the pre-training corpora. Rows are grouped by the initial checkpoint. In each group, the first row reports the results for the initial model; the indented rows are the results after applying continual pre-training.

	MMLU	ARC-C	HSwag	WinoG	NQ	DROP	AVG	PPL
OLMo-2-7B								
$R = 0$ (Base model)	59.8	72.6	81.3	75.8	29.0	40.7	59.9	11.038
$K = 0,\ R = 1$	59.9	72.0	81.1	75.3	28.3	40.2	59.5	11.340
$K = 0,\ R = 1$, *Low-PPL*	60.0	72.9	81.8	75.1	28.5	40.6	59.8	11.432
$K = 0,\ R = 1$, *Mid-PPL*	59.9	72.8	82.3	75.6	28.1	41.3	59.9	11.374
$K = 0,\ R = 1$, *High-PPL*	59.8	72.8	81.4	74.8	27.8	40.2	59.5	11.398

Table 2: Downstream performance and perplexity by different verification settings.

tal to train the next round model. To generate each instance, we conditioned the model on a prompt of $K = \{0, 3, 64\}$ tokens sampled from OLMo-Mix-1124, the pre-training corpus for Stage-1. We performed recursive continual training up to $R = 3$ rounds for each setting.

We followed the training configurations used in the Stage-2 pre-training of the OLMo-2 family. Specifically, we employed the AdamW optimizer with peak learning rates of 6e-5 for the 7B models and 9e-5 for the 13B models, scheduled to decay to zero by the end of training.

We ran experiments on a GPU cluster in which each node had eight NVIDIA H200 GPUs. Generating one billion tokens took less than one node-day for both 7B and 13B models. Continual pre-training on 1B tokens required again less than one node-day for both 7B and 13B models.

3.2 Evaluation Metrics

We evaluated the performance on downstream tasks as well as perplexity on the original training corpus. As for the downstream tasks, we targeted the following benchmark datasets which were used in Team OLMo et al. (2025): MMLU (Hendrycks et al., 2020), ARC-Challenge (ARC-C) (Clark et al., 2018), HellaSwag (HSwag) (Zellers et al., 2019), WinoGrande (WinoG) (Sakaguchi et al., 2021), Natural Questions (NQ) (Kwiatkowski et al., 2019), and DROP (Dua et al., 2019). As a summary for these tasks, we computed the average score across the benchmarks (AVG). We calculated the perplexity (PPL) on the OLMo-Mix-1124 corpus.

3.3 Results

Table 1 shows our main results, while Figure 2 illustrates the score shift across generations for each setting. We observed that recursive self-training failed

	MMLU	ARC-C	HSwag	WinoG	NQ	DROP	AVG	PPL
OLMo-2-7B								
$R = 0$ (Base model)	59.8	72.6	81.3	75.8	29.0	40.7	59.9	11.434
$K = 0$, $R = 1$ *from Stage-1*	59.9	72.0	81.5	75.1	29.0	40.9	59.7	11.752
dolmino-mix, $R = 1$	60.9	73.8	82.1	76.2	29.3	41.0	60.6	11.104
olmo-mix, $R = 1$	63.9	79.5	83.9	76.4	34.3	59.8	66.3	11.104
OLMo-2-13B								
$R = 0$ (Base model)	63.4	80.2	84.8	79.4	34.5	49.6	65.3	10.500
$K = 0$, $R = 1$	62.9	79.5	84.1	77.7	33.7	49.1	64.5	10.603
dolmino-mix, $R = 1$	59.1	80.1	85.2	81.3	35.1	50.9	65.3	10.060
olmo-mix, $R = 1$	64.5	80.7	84.4	77.9	34.5	48.7	65.1	10.216

Table 3: Downstream performance and corpus perplexity, using the pre-training corpus for training.

	Win-SC (%)	Win-S1-PTC (%)	Tie (%)
OLMo-2-7B			
$K = 0$, $R = 1$	10.82	86.32	2.86
$K = 0$, $R = 2$	6.24	91.42	2.34
$K = 0$, $R = 3$	1.44	98.16	0.40
$K = 3$, $R = 1$	7.80	91.06	1.14
$K = 3$, $R = 2$	2.76	96.64	0.60
$K = 3$, $R = 3$	1.44	98.16	0.40
$K = 64$, $R = 1$	19.14	78.94	1.92
$K = 64$, $R = 2$	15.89	81.11	3.00
$K = 64$, $R = 3$	12.94	83.58	3.48
OLMo-2-13B			
$K = 0$, $R = 1$	18.02	77.24	4.74
$K = 0$, $R = 2$	11.72	85.00	3.28
$K = 0$, $R = 3$	7.42	90.34	2.24
$K = 3$, $R = 1$	12.32	86.02	1.66
$K = 3$, $R = 2$	4.20	94.74	1.06
$K = 3$, $R = 3$	2.48	96.90	0.62
$K = 64$, $R = 1$	19.38	78.04	2.58
$K = 64$, $R = 2$	15.10	82.78	2.12
$K = 64$, $R = 3$	14.78	82.64	2.58

Table 4: Pairwise evaluation results comparing the synthetic corpus (SC) against the original Stage-1 pre-training corpus (S1-PTC; OLMo-Mix-1124). "Win-SC" and "Win-S1-PTC" denote the percentages of samples in which SC and S1-PTC are judged better, respectively, and "Tie" denotes the percentage of samples judged equally good.

to improve model performance in all cases, indicating model collapse rather than self-improvement. Regardless of the choice of K token prefixes, downstream task accuracy did not increase in any generations ($R = \{1, 2, 3\}$), and perplexity either remained unchanged or worsened. Additionally, as shown in Table2, filtering generated data by perplexity did not contribute to performance gains.

4 Analyses

4.1 Control Experiments

We conducted control experiments to disentangle whether the observed performance changes stemmed from the nature of the generated data or from other factors (e.g., an overly high learning rate). Concretely, starting from the Stage-1 checkpoints, we continued pre-training on 1B tokens sampled from Dolmino-Mix-1124, the corpus used for Stage-2 pre-training. We contrasted this setting with continued pre-training on a 1B-token synthetic corpus generated from the Stage-1 checkpoints using our pipeline with $K = 0$. Apart from the corpus, all hyper-parameters and procedures were the same as those described in Section 3.1.

Table 3 shows the results. Continual pre-training on the actual corpus for 7B model improved performance across all of downstream tasks and reduced perplexity and for 13B model, its score in some tasks decreased from that of base model, but in most of tasks these were higher than $K = 0, R = 1$ setting. These results indicate that the failure of self-improvement arises from the quality of the synthetic data rather than the training configurations.

	Win-SC (%)	**Win-S2-PTC (%)**	**Tie (%)**
OLMo-2-7B			
$K = 0$, $R = 1$	5.56	92.54	1.90
$K = 0$, $R = 2$	3.08	95.96	0.96
$K = 0$, $R = 3$	1.52	97.76	0.72
$K = 3$, $R = 1$	3.42	95.94	0.65
$K = 3$, $R = 2$	0.90	98.78	0.32
$K = 3$, $R = 3$	0.36	99.40	0.24
$K = 64$, $R = 1$	11.44	87.12	1.44
$K = 64$, $R = 2$	8.98	89.60	1.42
$K = 64$, $R = 3$	7.26	91.78	0.96
OLMo-2-13B			
$K = 0$, $R = 1$	18.02	77.24	4.74
$K = 0$, $R = 2$	11.72	85.00	3.28
$K = 0$, $R = 3$	7.42	90.34	2.24
$K = 3$, $R = 1$	6.52	92.27	1.20
$K = 3$, $R = 2$	1.73	97.78	0.49
$K = 3$, $R = 3$	0.72	99.05	0.23
$K = 64$, $R = 1$	10.58	87.38	2.04
$K = 64$, $R = 2$	9.04	89.50	1.46
$K = 64$, $R = 3$	8.22	90.22	1.56

Table 5: Pairwise evaluation results comparing the synthetic corpus (SC) against the original Stage-2 pre-training corpus (S2-PTC; Dolmino-Mix-1124). “Win-SC” and “Win-S2-PTC” denote the percentages of samples in which SC and S2-PTC are judged better, respectively, and “Tie” denotes the percentage of samples judged equally good.

4.2 Analysis on Diversity

Motivated by the observation that perplexity-based filtering did not translate into performance gains, we hypothesized that synthetic data might degraded diversity. For each synthetic corpus, we encoded its texts into dense vectors with the the embedding model and computed the cosine similarity matrix between all embeddings. Let S be the set of off-diagonal, upper-triangular cosine similarities. Our diversity score is:

$$\text{Diversity} = 1 - \text{mean}(S),$$

the complement of the average pairwise similarity (higher is more diverse). In addition, we report the statistics (mean, median, min, max) for descriptive analysis. To quantify the diversity of generated corpora, we used the Multilingual-E5 sentence embedding model (Wang et al., 2024).

Table 6 shows the results. However, no consistent relationship was observed between the diversity scores and downstream task performance. This suggests that insufficient data diversity is unlikely to be the primary factor driving the observed performance degradation.

4.3 Analysis on Quality

As additional investigation of the synthetic data, we conducted a relative evaluation of its quality as pre-training data compared to original corpus, using GPT-4o (OpenAI et al., 2024).[6] As shown in Table 4 and Table 5, across all settings, OLMo-Mix-1124 and Dolmino-Mix-1124 were rated as better than the synthetic data by an large proportion. Moreover, this proportion increased with later rounds. These results indicate that the synthetic data is markedly inferior to original corpus as pre-training data. Based on these results, the difference of data quality is considered as a major factor that contributed to the performance degradation.

5 Conclusion

In this work, we investigated about the mixed results of self-training using non-toy large language models through simple recursive pre-training on self-generated data, conducting experiments across multiple model sizes and data synthesis strategies. Our results consistently showed that self-improvement did not occur under any of the tested methods. These results suggest that model collapse observed in naive recursive training is inherent to the training procedure itself, wheres self-improvement likely owes its success not to the model’s autonomous refinement but to human-designed, strategic synthetic pipelines that inject external intelligence.

[6] The prompt can be found in Appendix B.

Limitations

Our experiments were conducted using open-source models from the OLMo-2 family, selected for their high reproducibility and accessibility. While these models are among the strongest open models currently available, they remain substantially smaller than state-of-the-art closed models such as GPT-5 or Claude 4. Consequently, our findings may not fully capture the behaviors of larger proprietary systems, which could exhibit different dynamics in self-improvement or model collapse.

Acknowledgments

We used ABCI 3.0 provided by AIST and AIST Solutions with support from “ABCI 3.0 Development Acceleration Use.”

References

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

DatologyAI, Pratyush Maini, Vineeth Dorna, Parth Doshi, Aldo Carranza, Fan Pan, Jack Urbanek, Paul Burstein, Alex Fang, Alvin Deng, Amro Abbas, Brett Larsen, Cody Blakeney, Charvi Bannur, Christina Baek, Darren Teh, David Schwab, Haakon Mongstad, Haoli Yin, and 11 others. 2025. Beyondweb: Lessons from scaling synthetic data for trillion-scale pretraining. *Preprint*, arXiv:2508.10975.

George Drayson, Emine Yilmaz, and Vasileios Lampos. 2025. Machine-generated text detection prevents language model collapse. *Preprint*, arXiv:2502.15654.

Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. 2019. DROP: A reading comprehension benchmark requiring discrete reasoning over paragraphs. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2368–2378, Minneapolis, Minnesota. Association for Computational Linguistics.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, and 542 others. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Suriya Gunasekar, Yi Zhang, Jyoti Aneja, Caio César Teodoro Mendes, Allie Del Giorno, Sivakanth Gopi, Mojan Javaheripi, Piero Kauffmann, Gustavo de Rosa, Olli Saarikivi, Adil Salim, Shital Shah, Harkirat Singh Behl, Xin Wang, Sébastien Bubeck, Ronen Eldan, Adam Tauman Kalai, Yin Tat Lee, and Yuanzhi Li. 2023. Textbooks are all you need. *arXiv preprint arXiv:2306.11644*.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2020. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*.

David Herel and Tomas Mikolov. 2024. Collapse of self-trained language models. *Preprint*, arXiv:2404.02305.

Jiaxin Huang, Shixiang Shane Gu, Le Hou, Yuexin Wu, Xuezhi Wang, Hongkun Yu, and Jiawei Han. 2022. Large language models can self-improve. *Preprint*, arXiv:2210.11610.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. 2019. Natural questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:452–466.

Siheng Li, Cheng Yang, Zesen Cheng, Lemao Liu, Mo Yu, Yujiu Yang, and Wai Lam. 2024. Large language models can self-improve in long-context reasoning. *Preprint*, arXiv:2411.08147.

Max Marion, Ahmet Üstün, Luiza Pozzobon, Alex Wang, Marzieh Fadaee, and Sara Hooker. 2023. When less is more: Investigating data pruning for pretraining llms at scale.

OpenAI, Aaron Hurst, Adam Lerer, Adam P. Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, Aleksander Mądry, Alex Baker-Whitcomb, Alex Beutel, Alex Borzunov, Alex Carney, Alex Chow, Alex Kirillov, Alex Nichol, and 400 others. 2024. Gpt-4o system card. *Preprint*, arXiv:2410.21276.

Zeyu Qin, Qingxiu Dong, Xingxing Zhang, Li Dong, Xiaolong Huang, Ziyi Yang, MAHMOUD KHADEMI, Dongdong Zhang, Hany Hassan Awadalla, Yi R. Fung, Weizhu Chen, Minhao Cheng, and Furu Wei. 2025. Scaling laws of synthetic data for language model. In *Proceedings of the Second Conference on Language Modeling*.

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

Ziyu Shang, Jianghan Liu, Zhizhao Luo, Peng Wang, Wenjun Ke, Jiajun Liu, Zijie Xu, and Guozheng Li. 2025. Acquisition and application of novel knowledge in large language models. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 18348–18368, Vienna, Austria. Association for Computational Linguistics.

Ilia Shumailov, Zakhar Shumaylov, Yiren Zhao, Nicolas Papernot, Ross Anderson, and Yarin Gal. 2024. Ai models collapse when trained on recursively generated data. *Nature*, 631(8022):755–759.

Team OLMo, Pete Walsh, Luca Soldaini, Dirk Groeneveld, Kyle Lo, Shane Arora, Akshita Bhagia, Yuling Gu, Shengyi Huang, Matt Jordan, Nathan Lambert, Dustin Schwenk, Oyvind Tafjord, Taira Anderson, David Atkinson, Faeze Brahman, Christopher Clark, Pradeep Dasigi, Nouha Dziri, and 21 others. 2025. 2 olmo 2 furious. *Preprint*, arXiv:2501.00656.

Lecheng Wang, Xianjie Shi, Ge Li, Jia Li, Xuanming Zhang, Yihong Dong, Wenpin Jiao, and Hong Mei. 2025a. Theoretical proof that auto-regressive language models collapse when real-world data is a finite set. *Preprint*, arXiv:2412.14872.

Liang Wang, Nan Yang, Xiaolong Huang, Linjun Yang, Rangan Majumder, and Furu Wei. 2024. Multilingual e5 text embeddings: A technical report. *arXiv preprint arXiv:2402.05672*.

Ze Wang, Zekun Wu, Jeremy Zhang, Xin Guan, Navya Jain, Skylar Lu, Saloni Gupta, and Adriano Koshiyama. 2025b. Bias amplification: Large language models as increasingly biased media. *Preprint*, arXiv:2410.15234.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, and 41 others. 2025. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. HellaSwag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4791–4800, Florence, Italy. Association for Computational Linguistics.

A Diversity of Generated Texts

Table 6 provides the detailed diversity statistics of generated texts.

Model	mean	median
OLMo-2-7B	0.2423	0.2333
$K = 0$, $R = 1$	0.2340	0.2268
$K = 0$, $R = 2$	0.2258	0.2195
$K = 0$, $R = 3$	0.2182	0.2132
$K = 3$, $R = 1$	0.2221	0.2191
$K = 3$, $R = 2$	0.2105	0.2081
$K = 3$, $R = 3$	0.2001	0.1980
$K = 64$, $R = 1$	0.2246	0.2218
$K = 64$, $R = 2$	0.2120	0.2094
$K = 64$, $R = 3$	0.1947	0.1927
$K = 0$, $R = 1$, *Low-PPL*	0.2456	0.2383
$K = 0$, $R = 1$, *Mid-PPL*	0.2412	0.2318
$K = 0$, $R = 1$, *High-PPL*	0.2141	0.2097
OLMo-2-13B	0.2365	0.2313
$K = 0$, $R = 1$	0.2475	0.2392
$K = 0$, $R = 2$	0.2313	0.2268
$K = 0$, $R = 3$	0.2312	0.2268
$K = 3$, $R = 1$	0.2248	0.2219
$K = 3$, $R = 2$	0.2113	0.2087
$K = 3$, $R = 3$	0.2012	0.1989
$K = 64$, $R = 1$	0.2246	0.2218
$K = 64$, $R = 2$	0.2120	0.2094
$K = 64$, $R = 3$	0.2016	0.1993

Table 6: Diversity of generated texts.

B Evaluation Prompt for Synthetic Data

Listing 1: Prompt used for pairwise data quality evaluation

```
Please act as an impartial judge and evaluate
    the quality of two text passages.
You should choose the passage that is of higher
    overall quality as potential training data
    for large language models.
Your evaluation should consider clarity,
    coherence, informativeness, naturalness,
    factuality, and absence of noise or
    artifacts.
Avoid any position biases and ensure that the
    order of the texts does not influence your
    decision.
Do not favor verbosity. Be objective and provide
     a short explanation.
After providing your explanation, output your
    final verdict strictly in one of these
    formats:
"[[A]]" if text A is better, "[[B]]" if text B
    is better, or "[[C]]" for a tie.
```