DATA MIXING CAN INDUCE PHASE TRANSITIONS IN KNOWLEDGE ACQUISITION

ABSTRACT

Large Language Models (LLMs) are typically trained on data mixtures: most data come from web scrapes, while a small portion is curated from high-quality sources with dense domain-specific knowledge. In this paper, we show that when training LLMs on such data mixtures, knowledge acquisition from knowledge-dense datasets does not always follow a smooth scaling law but can exhibit phase transitions with respect to the mixing ratio and model size. First, through controlled experiments on a synthetic biography dataset mixed with web-scraped data, we demonstrate that: (1) as we increase the model size to a critical value, the model suddenly transitions from memorizing very few to most of the biographies; (2) below a critical mixing ratio, the model memorizes almost nothing even with extensive training, but beyond this threshold, it rapidly memorizes more biographies. We then adopt an information-theoretic perspective to understand and characterize the existence and value of the thresholds. Based on these insights, we identify two mitigation strategies that improve the efficiency of knowledge acquisition from knowledge-dense datasets, and validate their effectiveness on both synthetic and real-world Wikipedia datasets.

1 INTRODUCTION

Large Language Models (LLMs) are often trained on two types of datasets. The first is a web-scraped corpus (Raffel et al., 2020; Penedo et al., 2024; Li et al., 2024), often spanning billions to trillions of tokens across diverse topics and styles. Due to the scale, it is hard to ensure its information density and relevance to downstream tasks. The second type of data involves smaller datasets curated from high-quality sources (e.g., Wikipedia, OpenWebMath (Paster et al., 2024), StarCoder (Li et al., 2023; Kocetkov et al., 2022)), providing dense knowledge on specific tasks or domains such as world knowledge, math, and code. As the second-type data usually take only a small fraction of the entire corpus, one may wonder: *How much knowledge can an LLM acquire from these knowledge-dense datasets?* Answering this question is crucial to understanding the ultimate performance that a model can achieve as we continue to improve data quality and scale up training compute.

One crucial aspect is the *model size*. In a controlled setting where the pre-training data only contains synthetic biographies, Allen-Zhu & Li (2024a) found that the amount of knowledge a sufficiently trained model stores scales linearly with the model size. Similar scaling was observed for memorizing Wikidata fact triples by Lu et al. (2024), and theoretically analyzed by Nichani et al. (2025).

This paper quantitatively analyzes how much knowledge a model can acquire from a knowledge-dense dataset under *data mixing*, where the dataset constitutes only a fraction r of the pre-training data, with the rest being a large-scale web corpus. We show that knowledge acquisition from knowledge-dense datasets, when mixed with the web text, no longer follows a linear scaling law but instead exhibits a more intricate behavior with notable phase transitions with respect to mixing ratios and model sizes.

More specifically, we study factual knowledge acquisition. We follow Allen-Zhu & Li (2024a) to curate a synthetic dataset of biographies with uniform data format and content, which enables us to quantify how much knowledge the model has stored simply by counting the number of memorized biographies. We then pre-train Pythia (Biderman et al., 2023) models of different sizes on a mixture of this biography dataset and FineWeb-Edu (Penedo et al., 2024), where the model passes the biography dataset for multiple epochs but FineWeb-Edu for less than one epoch. See Appendix A for detailed experimental setup.

Of course, setting r closer to 1 will make the model memorize more biographies, but it also hurts the model's general capabilities that are supposed to be learned from the more diverse web corpus. Therefore, the essence of our study is to understand whether models can still memorize a decent number of biographies for relatively small r ($< 40\%$). Our main findings are as follows (Section 2):

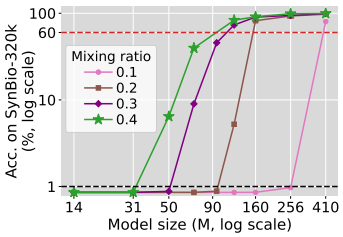
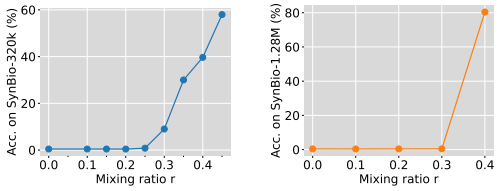


Figure 1: Phase transition in model size.



(a) 70M models. (b) 410M models.

Figure 2: Phase transition in mixing ratio.

Finding 1: Phase Transition in Model Size (Figure 1). When varying the model size while keeping the mixing ratio r fixed, the number of biographies memorized by the model does not scale linearly with its size but instead exhibits a phase transition behavior. When the model size is smaller than a critical model size N_0, the number of memorized biographies can be nearly zero, and only until the model size reaches N_0, the model *suddenly* memorizes most of the biographies. The threshold N_0 is higher for smaller r.

Finding 2: Phase Transition in Mixing Ratio (Figure 2). When varying the mixing ratio r while keeping the model size fixed, we find that below a critical mixing ratio r_0, the model memorizes almost nothing even after significantly longer training, during which each biography appears tens of times more in total (Figures 3(a) and 7), but beyond r_0, the number of memorized biographies grows rapidly with r.

We further find that as we gradually decrease r, the number of steps needed to memorize a fixed number of biographies initially grows linearly with $1/r$ (Figure 3(b)), but soon becomes exponential and even faster than exponential (Figure 3(c)), making it impossible or practically infeasible for the model to memorize a nontrivial number of biographies despite extensive training passes.

The above findings reveal a caution for practitioners that the mixing ratio should be set with care for the model: mixing in knowledge-dense datasets with small mixing ratios can offer no benefit at all, especially when training small LMs.

Theoretical Analysis (Section 3). We further present an information-theoretic explanation for the observed phase transitions. We show that these behaviors are not unique to LLM pre-training but can also arise in any learning algorithm that optimally minimizes the overall test loss with a bounded model capacity, which we refer to as *optimal bounded-capacity learner*. By assuming that the optimal test loss follows a power law in model size, we show how phase transition depends on the model size, mixing ratio, and the exposure frequency of each fact in a knowledge-dense dataset. We also derive a power-law relationship between the threshold frequency of a fact and the model size, and verify this power-law relationship empirically (see Appendix D).

Mitigation Strategies. Inspired by our theory, we further propose two mitigation strategies: increasing the frequency just a few times—either by subsampling or rewriting with a more compact format—can significantly improve the efficiency of knowledge acquisition. See Appendix E.

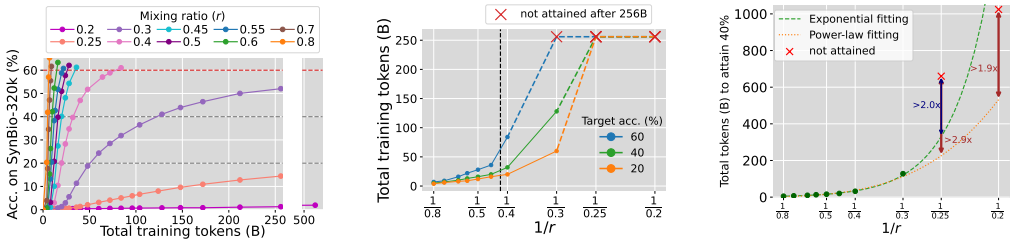
2 Phase Transitions Under Data Mixing

2.1 Phase Transition in Model Size

For each $r \in \{0.1, 0.2, 0.3, 0.4\}$, we train models with sizes ranging from 14M to 410M on the mixture of FineWeb-Edu and SynBio-320k for 32B tokens, which is approximately four times the optimal computation for 410M models predicted by the Chinchilla scaling law (Hoffmann et al., 2022). As shown in Figure 1, as model size increases, accuracy on SynBio initially remains near zero. Once the model size surpasses some threshold, accuracy rapidly grows above 60%. The transition is consistently sharp across different mixing ratios while larger r leads to a smaller critical point.

2.2 Phase Transition in Mixing Ratio

Accuracy on the knowledge dataset undergoes a phase transition as mixing ratio increases. We begin by training models of the same size with different mixing ratios r. Specifically, we train 70M models on the mixture of FineWeb-Edu and SynBio-320K, varying r from 0.1 to 0.45 (stepsize 0.05), and 410M models on the mixture of FineWeb-Edu and SynBio-1.28M, varying r from 0.1 to 0.4 (stepsize 0.1). All models are trained for a total of 32B tokens. As shown in Figure 2(a), for 70M models, as r increases from 0.1 to 0.25, its accuracy on SynBio remains near zero. Only when r exceeds 0.3 does the accuracy begin to steadily improve with increasing r. In Figure 2(b), the accuracy for 410M models exhibit similar trends where it remains near zero for $r \leq 0.3$ and suddenly attains 80% when r grows to 0.4.



(a) Train until accuracy achieves 60% accuracy or until a total of 256B tokens are passed.

(b) Relationship between required training steps to achieve target accuracy and $1/r$.

(c) Fitting required training steps to attain 40% accuracy against $1/r$.

Figure 3: Training efficiency declines sharply as r decreases. We train 70M models on the mixture of FineWeb-Edu and SynBio-128k. (a) For each r, we train until accuracy achieves 60% accuracy or until a total of 256B tokens are passed. Notably, accuracy for $r = 0.2$ remains near zero even after extensive training up to 512B tokens. (b) Required training steps to achieve target accuracy initially grows linearly with $1/r$, but then escalate rapidly. (c) Required training steps increase exponentially or even superexponentially with $1/r$.

Training longer barely helps for low mixing ratios. We extend the training horizon for $r = 0.2$ to 512B tokens for the 70M model and 128B for the 410M model, where the model passes SynBio hundreds or even thousands of times. However, as shown in Figures 3(a) and 7, the accuracy on SynBio remains near zero for both model sizes, even with these substantial extensions.

Required training steps to achieve target accuracy initially grows linearly with $1/r$, but then escalate rapidly. We further examine how mixing ratio affects the training efficiency in knowledge acquisition by measuring the total number of training steps required to reach a target accuracy, denoted as T, across different mixing ratios r. Specifically, we train 70M models with mixing ratios from $\{0.2, 0.25, 0.3, 0.4, 0.45, 0.5, 0.55, 0.6, 0.7, 0.8\}$. For each mixing ratio, we evaluate 20 training horizons, approximately evenly spaced on a logarithmic scale with a factor of 1.2, ranging from 0 to 256B tokens. For $r > 0.4$, where accuracy improves rapidly, we assess additional training horizons for more precise estimations. Training continues until the model reaches 60% accuracy or exhausts 256B tokens. As shown in Figures 3(a) and 3(b), as we decrease r from 0.8, T initially increase linearly with $1/r$ for $r > 0.4$ and quickly deviates from the linear trend as r falls below 0.4.

Quantifying the growth rate: required training steps increase exponentially or even superexponentially with $1/r$. To quantify the growth rate of T, we analyze the results for target accuracy 40% and fit a scaling law for T against $1/r$. Specifically, we use an exponential function to fit T against $1/r$ for all $r \geq 0.3$, where 40% accuracy is achieved within 256B tokens. Additionally, we use a power-law function to fit T against $1/r$ for $r \in \{0.3, 0.4, 0.45, 0.5, 0.55\}$. Details on the fitting process can be found in Appendix H.3. To examine whether T follows the fitted trend as r decreases further, we extend the training for $r = 0.2$ and 0.25 to 660B and 1024B tokens, respectively. However, neither configurations attain 40% accuracy. As shown in Figure 3(c), the actual T is more than 2.9 times the power-law prediction for $r = 0.25$ and more than 1.9 times for $r = 0.2$. Notably, the actual T for $r = 0.25$ is even more than twice the exponential prediction. These significant deviations suggest exponential or even superexponential growth of T with respect to $1/r$ as r decreases.

We further present ablation studies on hyperparameters in Appendix B.

3 Theoretical Analysis

We take an information-theoretic perspective and point out that the observed phenomena are not unique to the current LLM architectures or optimization methods but can also happen for any model trained to maximally utilize its capacity to minimize the next-token prediction loss.

Data distribution and Learning Algorithm. The essence of language modeling is to model the distribution of the next token y for a given input context x consisting of all previous tokens. We take a Bayesian view: there is a latent variable $\theta \in \Theta$ that determines the data distribution of (x, y), denoted as $(x, y) \sim \mathcal{D}_\theta$. The universe first draws θ from a prior distribution $\mathcal{P}$ before we observe the data distribution $\mathcal{D}_\theta$. A learning algorithm $\mathcal{A}$ is a procedure that takes samples from a data distribution $\mathcal{D}$ of (x, y) and outputs a predictor $h = \mathcal{A}(\mathcal{D})$ in the end. For a given predictor h, we measure its performance by the expected cross-entropy loss $\mathcal{L}(h; \mathcal{D}) := \mathbb{E}_{(x,y)\sim\mathcal{D}}[-\log p(y \mid h, x)]$, where $p(y \mid h, x)$ denotes the predicted distribution of y given x by the predictor h. We also define the expected loss of a learning algorithm $\mathcal{A}$ under the prior $\mathcal{P}$ as $\bar{\mathcal{L}}_\mathcal{P}(\mathcal{A}) := \mathbb{E}_{\theta\sim\mathcal{P}}[\mathcal{L}(\mathcal{A}(\mathcal{D}_\theta); \mathcal{D}_\theta)]$. In practice, a predictor h can be a transformer model, and $\mathcal{A}$ can be the pre-training algorithm.

Model Capacity and Mutual Information. We measure the "effective" model capacity by the mutual information (MI) between the model and the data distribution $\mathcal{D}_\theta$, namely $I(\mathcal{A}(\mathcal{D}_\theta); \mathcal{D}_\theta)$. Practical learning algorithms usually have a bounded capacity. If $\mathcal{A}$ always outputs a model h with at

most N parameters, each represented by a b-bit floating-point number, then $I(\mathcal{A}(\mathcal{D}_\theta);\mathcal{D}_\theta) \le bN$. Empirically, Allen-Zhu & Li (2024a) found that $I(\mathcal{A}(\mathcal{D}_\theta);\mathcal{D}_\theta) \approx 2N$ in their controlled setting.

Optimal Bounded-Capacity Learner. Now, imagine that we train a model under a capacity constraint $I(\mathcal{A}(\mathcal{D}_\theta);\mathcal{D}_\theta) \le M$, where M is a constant. How does the resulting model behave when the training procedure is optimized, and massive computation is dedicated to minimizing the loss as much as possible? This question motivates us to define the following notion:

Definition 3.1. For a given prior $\mathcal{P}$ and $M > 0$, the best achievable loss under a capacity constraint M is defined as $F_{\mathcal{P}}(M) := \inf_{\mathcal{A}} \left\{ \bar{\mathcal{L}}_{\mathcal{P}}(\mathcal{A}) : I(\mathcal{A}(\mathcal{D}_\theta);\mathcal{D}_\theta) \le M \right\}$, where the infimum is taken over all learning algorithms. An optimal M-bounded-capacity learner is a learning algorithm $\mathcal{A}$ such that $I(\mathcal{A}(\mathcal{D}_\theta);\mathcal{D}_\theta) \le M$ and $\bar{\mathcal{L}}_{\mathcal{P}}(\mathcal{A}) = F_{\mathcal{P}}(M)$.

Data Mixing Induces Phase Transitions. In Appendix C, we show that in the simple case where there is no data mixing with other domains and the data distribution $\mathcal{D}_\theta$ consists of only K random facts, the optimal bounded-capacity learner reduces the expected loss linearly with the capacity M, thus no phase transition in capacity. See also Appendix C and Appendix I for full details.

What if we mix the random facts with another domain, say web text? Imagine that the data distribution $\mathcal{D}_\theta$ consists of two domains. The first is a mixture of K random facts. The second is another data domain that can have a much more complex structure. We assume that the latent variable θ is structured as $\theta = (\theta_1, \theta_2)$ and the data distribution $\mathcal{D}_\theta$ is mixed together as $\mathcal{D}_\theta = (1-r)\mathcal{D}^{(1)}_{\theta_1} + r\mathcal{D}^{(2)}_{\theta_2}$, where $r \in (0,1)$ is the mixing ratio, and $\mathcal{D}^{(1)}_{\theta_1}$ and $\mathcal{D}^{(2)}_{\theta_2}$ are the data distributions of the two domains. We use p to denote the exposure frequency of any fact in the first domain, which is the probability for the fact to occur in $\mathcal{D}_\theta$. Let H_{tot} be the total entropy of the target tokens in the first domain. We define $\bar{\mathcal{L}}_1(\mathcal{A}) := \mathbb{E}_{\theta\sim\mathcal{P}_1}[\mathcal{L}(\mathcal{A}(\mathcal{D}_\theta);\mathcal{D}^{(1)}_{\theta_1})]$, which measures the model's performance on the first domain. If $\bar{\mathcal{L}}_1(\mathcal{A}) = F_{\mathcal{P}_1}(0)$, then the learner's performance is the same as random guessing without seeing any data. If $\bar{\mathcal{L}}_1(\mathcal{A}) = F_{\mathcal{P}_1}(\infty)$, the learner learns the facts perfectly. The following theorem shows that the optimal bounded-capacity learner no longer learns the facts linearly, but instead exhibits a sharp phase transition in M. Two key functions characterizing the transition are

$$M_0^-(x) := \sup\{M \ge 0 : -F'_{\mathcal{P}_2}(M) > x\}, \qquad M_0^+(x) := \inf\{M \ge 0 : -F'_{\mathcal{P}_2}(M) < x\}.$$

Theorem 3.2. *For any optimal M-bounded-capacity learner $\mathcal{A}$, (1) if $M \le M_0^-(\frac{r}{1-r}\cdot p)$, then $\bar{\mathcal{L}}_1(\mathcal{A}) = F_{\mathcal{P}_1}(0)$; (2) if $M \ge M_0^+(\frac{r}{1-r}\cdot p) + H_{\text{tot}}$, then $\bar{\mathcal{L}}_1(\mathcal{A}) = F_{\mathcal{P}_1}(\infty)$.*

Key Example: When Web Data Loss Follows a Power Law in Model Size. Consider the case where $F_{\mathcal{P}_2}(M)$ is a power-law function of M, *i.e.*, $F_{\mathcal{P}_2}(M) = C + A \cdot M^{-\alpha}$. Here, $\alpha \in (0,1)$ and A is a large constant. This is a reasonable assumption since LLM pre-training usually exhibits such a power-law scaling behavior in model size, as observed by many works (Kaplan et al., 2020; Hoffmann et al., 2022). If the second domain is a large-scale dataset scraped from the web, then we should expect that the best achievable loss on this domain is a power-law function of the model capacity. In this case, taking the derivative of $F_{\mathcal{P}_2}(M)$ gives $-F'_{\mathcal{P}_2}(M) = A \cdot \alpha \cdot M^{-\alpha-1}$. Then, $M_0^-(x) = M_0^+(x) = (\frac{A\alpha}{x})^{1/(\alpha+1)}$. Applying Theorem 3.2, we quantify the transition point by

$$M_{\text{thres}} \sim \left(\frac{A}{rp}\right)^{1/(\alpha+1)}. \tag{1}$$

This implies that even if the model has the capacity to learn the first domain, it may still need to be very large to acquire any knowledge from it, especially when r or p is small. Arranging the terms in (1), we can also obtain the transition point in the mixing ratio r:

$$r_{\text{thres}} \sim \frac{A}{p \cdot M^{\alpha+1}}, \tag{2}$$

which aligns with the empirical observation that r has to be larger than a critical mixing ratio for the model to learn any of the facts.

4 Discussions and Future Directions

While our experiments focus on mixing factual knowledge with web corpus, such transitions may also happen for other types of knowledge, such as math, coding, and procedural knowledge (Ruis et al., 2024). We leave the extension to more types of knowledge for future work. Another important future direction is to apply our theory-inspired mitigation strategies to accelerate LLMs' knowledge acquisition, especially for small models with limited capacity.

BROADER IMPACT

This paper identifies two phase transitions in knowledge acquisition within data mixtures and provides theoretical understanding of these phenomena. Building on our theory, we propose two mitigation strategies to enhance the efficiency of knowledge acquisition. Our findings offer deeper insights into LLM behavior and can be applied to improve the factual accuracy of LLMs.

REFERENCES

Zeyuan Allen-Zhu and Yuanzhi Li. Physics of language models: Part 3.1, knowledge storage and extraction. *arXiv preprint arXiv:2309.14316*, 2023.

Zeyuan Allen-Zhu and Yuanzhi Li. Physics of language models: Part 3.3, knowledge capacity scaling laws. *arXiv preprint arXiv:2404.05405*, 2024a.

Zeyuan Allen-Zhu and Yuanzhi Li. Physics of language models: Part 3.2, knowledge manipulation, 2024b.

Alex Andonian, Quentin Anthony, Stella Biderman, Sid Black, Preetham Gali, Leo Gao, Eric Hallahan, Josh Levy-Kramer, Connor Leahy, Lucas Nestler, Kip Parker, Michael Pieler, Jason Phang, Shivanshu Purohit, Hailey Schoelkopf, Dashiell Stander, Tri Songz, Curt Tigges, Benjamin Thérien, Phil Wang, and Samuel Weinbach. GPT-NeoX: Large Scale Autoregressive Language Modeling in PyTorch, 9 2023. URL `https://www.github.com/eleutherai/gpt-neox`.

Stella Biderman, Hailey Schoelkopf, Quentin Gregory Anthony, Herbie Bradley, Kyle O'Brien, Eric Hallahan, Mohammad Aflah Khan, Shivanshu Purohit, USVSN Sai Prashanth, Edward Raff, et al. Pythia: A suite for analyzing large language models across training and scaling. In *International Conference on Machine Learning*, pp. 2397–2430. PMLR, 2023.

Stella Biderman, Usvsn Prashanth, Lintang Sutawika, Hailey Schoelkopf, Quentin Anthony, Shivanshu Purohit, and Edward Raff. Emergent and predictable memorization in large language models. *Advances in Neural Information Processing Systems*, 36, 2024.

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

Cody Blakeney, Mansheej Paul, Brett W. Larsen, Sean Owen, and Jonathan Frankle. Does your data spark joy? performance gains from domain upsampling at the end of training. In *First Conference on Language Modeling*, 2024.

Nicholas Carlini, Daphne Ippolito, Matthew Jagielski, Katherine Lee, Florian Tramer, and Chiyuan Zhang. Quantifying memorization across neural language models. In *The Eleventh International Conference on Learning Representations*, 2023.

Hoyeon Chang, Jinho Park, Seonghyeon Ye, Sohee Yang, Youngkyung Seo, Du-Seong Chang, and Minjoon Seo. How do large language models acquire factual knowledge during pretraining? In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

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

Jeff Da, Ronan Le Bras, Ximing Lu, Yejin Choi, and Antoine Bosselut. Analyzing commonsense emergence in few-shot knowledge models. In *3rd Conference on Automated Knowledge Base Construction*, 2021.

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

Vitaly Feldman. Does learning require memorization? a short tale about a long tail. In *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing*, pp. 954–959, 2020.

Steven Feng, Shrimai Prabhumoye, Kezhi Kong, Dan Su, Mostofa Patwary, Mohammad Shoeybi, and Bryan Catanzaro. Maximize your data's potential: Enhancing llm accuracy with two-phase pretraining. *arXiv preprint arXiv:2412.15285*, 2024.

Leo Gao, Jonathan Tow, Baber Abbasi, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Alain Le Noac'h, Haonan Li, Kyle McDonell, Niklas Muennighoff, Chris Ociepa, Jason Phang, Laria Reynolds, Hailey Schoelkopf, Aviya Skowron, Lintang Sutawika, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. A framework for few-shot language model evaluation, 07 2024.

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

Gaurav Rohit Ghosal, Tatsunori Hashimoto, and Aditi Raghunathan. Understanding finetuning for factual knowledge extraction. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 15540–15558. PMLR, 21–27 Jul 2024.

Dirk Groeneveld, Iz Beltagy, Evan Walsh, Akshita Bhagia, Rodney Kinney, Oyvind Tafjord, Ananya Jha, Hamish Ivison, Ian Magnusson, Yizhong Wang, Shane Arora, David Atkinson, Russell Authur, Khyathi Chandu, Arman Cohan, Jennifer Dumas, Yanai Elazar, Yuling Gu, Jack Hessel, Tushar Khot, William Merrill, Jacob Morrison, Niklas Muennighoff, Aakanksha Naik, Crystal Nam, Matthew Peters, Valentina Pyatkin, Abhilasha Ravichander, Dustin Schwenk, Saurabh Shah, William Smith, Emma Strubell, Nishant Subramani, Mitchell Wortsman, Pradeep Dasigi, Nathan Lambert, Kyle Richardson, Luke Zettlemoyer, Jesse Dodge, Kyle Lo, Luca Soldaini, Noah Smith, and Hannaneh Hajishirzi. OLMo: Accelerating the science of language models. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Bangkok, Thailand, August 2024. Association for Computational Linguistics.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, Thomas Hennigan, Eric Noland, Katherine Millican, George van den Driessche, Bogdan Damoc, Aurelia Guy, Simon Osindero, Karén Simonyan, Erich Elsen, Oriol Vinyals, Jack Rae, and Laurent Sifre. An empirical analysis of compute-optimal large language model training. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 30016–30030. Curran Associates, Inc., 2022.

Shengding Hu, Yuge Tu, Xu Han, Chaoqun He, Ganqu Cui, Xiang Long, Zhi Zheng, Yewei Fang, Yuxiang Huang, Weilin Zhao, et al. Minicpm: Unveiling the potential of small language models with scalable training strategies. *arXiv preprint arXiv:2404.06395*, 2024.

Jing Huang, Diyi Yang, and Christopher Potts. Demystifying verbatim memorization in large language models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 10711–10732, 2024.

Adam Ibrahim, Benjamin Thérien, Kshitij Gupta, Mats L Richter, Quentin Anthony, Timothée Lesort, Eugene Belilovsky, and Irina Rish. Simple and scalable strategies to continually pre-train large language models. *arXiv preprint arXiv:2403.08763*, 2024.

Nikhil Kandpal, Haikang Deng, Adam Roberts, Eric Wallace, and Colin Raffel. Large language models struggle to learn long-tail knowledge. In *International Conference on Machine Learning*, pp. 15696–15707. PMLR, 2023.

Feiyang Kang, Yifan Sun, Bingbing Wen, Si Chen, Dawn Song, Rafid Mahmood, and Ruoxi Jia. Autoscale: Automatic prediction of compute-optimal data composition for training llms. *arXiv preprint arXiv:2407.20177*, 2024.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020.

Denis Kocetkov, Raymond Li, Loubna Ben Allal, Jia Li, Chenghao Mou, Carlos Muñoz Ferrandis, Yacine Jernite, Margaret Mitchell, Sean Hughes, Thomas Wolf, et al. The stack: 3 tb of permissively licensed source code. *arXiv preprint arXiv:2211.15533*, 2022.

Jeffrey Li, Alex Fang, Georgios Smyrnis, Maor Ivgi, Matt Jordan, Samir Gadre, Hritik Bansal, Etash Guha, Sedrick Keh, Kushal Arora, et al. Datacomp-lm: In search of the next generation of training sets for language models. *arXiv preprint arXiv:2406.11794*, 2024.

R Li, LB Allal, Y Zi, N Muennighoff, D Kocetkov, C Mou, M Marone, C Akiki, J Li, J Chim, et al. Starcoder: May the source be with you! *Transactions on machine learning research*, 2023.

Qian Liu, Xiaosen Zheng, Niklas Muennighoff, Guangtao Zeng, Longxu Dou, Tianyu Pang, Jing Jiang, and Min Lin. Regmix: Data mixture as regression for language model pre-training. *arXiv preprint arXiv:2407.01492*, 2024.

Xingyu Lu, Xiaonan Li, Qinyuan Cheng, Kai Ding, Xuanjing Huang, and Xipeng Qiu. Scaling laws for fact memorization of large language models. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen (eds.), *Findings of the Association for Computational Linguistics: EMNLP 2024*, pp. 11263–11282, Miami, Florida, USA, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-emnlp.658.

Alex Mallen, Akari Asai, Victor Zhong, Rajarshi Das, Daniel Khashabi, and Hannaneh Hajishirzi. When not to trust language models: Investigating effectiveness of parametric and non-parametric memories. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki (eds.), *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 9802–9822, Toronto, Canada, July 2023. Association for Computational Linguistics.

Eshaan Nichani, Jason D. Lee, and Alberto Bietti. Understanding factual recall in transformers via associative memories. In *The Thirteenth International Conference on Learning Representations*, 2025.

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, Michal Guerquin, Hamish Ivison, Pang Wei Koh, Jiacheng Liu, Saumya Malik, William Merrill, Lester James V. Miranda, Jacob Morrison, Tyler Murray, Crystal Nam, Valentina Pyatkin, Aman Rangapur, Michael Schmitz, Sam Skjonsberg, David Wadden, Christopher Wilhelm, Michael Wilson, Luke Zettlemoyer, Ali Farhadi, Noah A. Smith, and Hannaneh Hajishirzi. 2 olmo 2 furious, 2025.

Denis Paperno, Germán Kruszewski, Angeliki Lazaridou, Quan Ngoc Pham, Raffaella Bernardi, Sandro Pezzelle, Marco Baroni, Gemma Boleda, and Raquel Fernández. The lambada dataset: Word prediction requiring a broad discourse context. *arXiv preprint arXiv:1606.06031*, 2016.

Keiran Paster, Marco Dos Santos, Zhangir Azerbayev, and Jimmy Ba. Openwebmath: An open dataset of high-quality mathematical web text. In *The Twelfth International Conference on Learning Representations*, 2024.

Guilherme Penedo, Hynek Kydlíček, Loubna Ben allal, Anton Lozhkov, Margaret Mitchell, Colin Raffel, Leandro Von Werra, and Thomas Wolf. The FineWeb datasets: Decanting the web for the finest text data at scale. In *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2024.

Fabio Petroni, Tim Rocktäschel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, Alexander H Miller, and Sebastian Riedel. Language models as knowledge bases? *arXiv preprint arXiv:1909.01066*, 2019.

Qwen, :, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. Qwen2.5 technical report, 2024.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21(140):1–67, 2020.

Adam Roberts, Colin Raffel, and Noam Shazeer. How much knowledge can you pack into the parameters of a language model? *arXiv preprint arXiv:2002.08910*, 2020.

Laura Ruis, Maximilian Mozes, Juhan Bae, Siddhartha Rao Kamalakara, Dwarak Talupuru, Acyr Locatelli, Robert Kirk, Tim Rocktäschel, Edward Grefenstette, and Max Bartolo. Procedural knowledge in pretraining drives reasoning in large language models. *arXiv preprint arXiv:2411.12580*, 2024.

Kai Sun, Yifan Xu, Hanwen Zha, Yue Liu, and Xin Luna Dong. Head-to-tail: How knowledgeable are large language models (llms)? aka will llms replace knowledge graphs? In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 311–325, 2024.

Gemma Team, Morgane Riviere, Shreya Pathak, Pier Giuseppe Sessa, Cassidy Hardin, Surya Bhupatiraju, Léonard Hussenot, Thomas Mesnard, Bobak Shahriari, Alexandre Ramé, Johan Ferret, Peter Liu, Pouya Tafti, Abe Friesen, Michelle Casbon, Sabela Ramos, Ravin Kumar, Charline Le Lan, Sammy Jerome, Anton Tsitsulin, Nino Vieillard, Piotr Stanczyk, Sertan Girgin, Nikola Momchev, Matt Hoffman, Shantanu Thakoor, Jean-Bastien Grill, Behnam Neyshabur, Olivier Bachem, Alanna Walton, Aliaksei Severyn, Alicia Parrish, Aliya Ahmad, Allen Hutchison, Alvin Abdagic, Amanda Carl, Amy Shen, Andy Brock, Andy Coenen, Anthony Laforge, Antonia Paterson, Ben Bastian, Bilal Piot, Bo Wu, Brandon Royal, Charlie Chen, Chintu Kumar, Chris Perry, Chris Welty, Christopher A. Choquette-Choo, Danila Sinopalnikov, David Weinberger, Dimple Vijaykumar, Dominika Rogozińska, Dustin Herbison, Elisa Bandy, Emma Wang, Eric Noland, Erica Moreira, Evan Senter, Evgenii Eltyshev, Francesco Visin, Gabriel Rasskin, Gary Wei, Glenn Cameron, Gus Martins, Hadi Hashemi, Hanna Klimczak-Plucińska, Harleen Batra, Harsh Dhand, Ivan Nardini, Jacinda Mein, Jack Zhou, James Svensson, Jeff Stanway, Jetha Chan, Jin Peng Zhou, Joana Carrasqueira, Joana Iljazi, Jocelyn Becker, Joe Fernandez, Joost van Amersfoort, Josh Gordon, Josh Lipschultz, Josh Newlan, Ju yeong Ji, Kareem Mohamed, Kartikeya Badola, Kat Black, Katie Millican, Keelin McDonell, Kelvin Nguyen, Kiranbir Sodhia, Kish Greene, Lars Lowe Sjoesund, Lauren Usui, Laurent Sifre, Lena Heuermann, Leticia Lago, Lilly McNealus, Livio Baldini Soares, Logan Kilpatrick, Lucas Dixon, Luciano Martins, Machel Reid, Manvinder Singh, Mark Iverson, Martin Görner, Mat Velloso, Mateo Wirth, Matt Davidow, Matt Miller, Matthew Rahtz, Matthew Watson, Meg Risdal, Mehran Kazemi, Michael Moynihan, Ming Zhang, Minsuk Kahng, Minwoo Park, Mofi Rahman, Mohit Khatwani, Natalie Dao, Nenshad Bardoliwalla, Nesh Devanathan, Neta Dumai, Nilay Chauhan, Oscar Wahltinez, Pankil Botarda, Parker Barnes, Paul Barham, Paul Michel, Pengchong Jin, Petko Georgiev, Phil Culliton, Pradeep Kuppala, Ramona Comanescu, Ramona Merhej, Reena Jana, Reza Ardeshir Rokni, Rishabh Agarwal, Ryan Mullins, Samaneh Saadat, Sara Mc Carthy, Sarah Cogan, Sarah Perrin, Sébastien M. R. Arnold, Sebastian Krause, Shengyang Dai, Shruti Garg, Shruti Sheth, Sue Ronstrom, Susan Chan, Timothy Jordan, Ting Yu, Tom Eccles, Tom Hennigan, Tomas Kocisky, Tulsee Doshi, Vihan Jain, Vikas Yadav, Vilobh Meshram, Vishal Dharmadhikari, Warren Barkley, Wei Wei, Wenming Ye, Woohyun Han, Woosuk Kwon, Xiang Xu, Zhe Shen, Zhitao Gong, Zichuan Wei, Victor Cotruta, Phoebe Kirk, Anand Rao, Minh Giang, Ludovic Peran, Tris Warkentin, Eli Collins, Joelle Barral, Zoubin Ghahramani, Raia Hadsell, D. Sculley, Jeanine Banks, Anca Dragan, Slav Petrov, Oriol Vinyals, Jeff Dean, Demis Hassabis, Koray Kavukcuoglu, Clement Farabet, Elena Buchatskaya, Sebastian Borgeaud, Noah Fiedel, Armand Joulin, Kathleen Kenealy, Robert Dadashi, and Alek Andreev. Gemma 2: Improving open language models at a practical size, 2024.

Kushal Tirumala, Aram Markosyan, Luke Zettlemoyer, and Armen Aghajanyan. Memorization without overfitting: Analyzing the training dynamics of large language models. *Advances in Neural Information Processing Systems*, 35:38274–38290, 2022.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

Johannes Welbl, Nelson F Liu, and Matt Gardner. Crowdsourcing multiple choice science questions. *arXiv preprint arXiv:1707.06209*, 2017.

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

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. Hellaswag: Can a machine really finish your sentence? *arXiv preprint arXiv:1905.07830*, 2019.

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

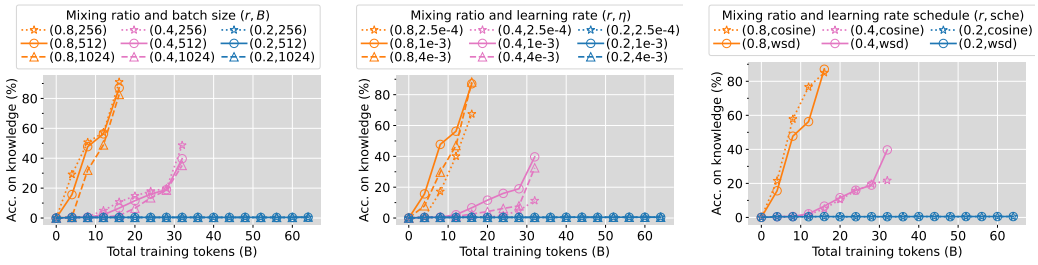
A Experimental Setup

The SynBio Dataset. We follow the approach of Allen-Zhu & Li (2024b) to curate a synthetic biography dataset with uniform data format and content. Specifically, each individual is characterized by five attributes: birth date, birth city, university, major, and employer. For each individual, the value of each attribute is randomly and independently sampled from a predefined domain. These (name, attribute, value) triplets are then converted into natural text descriptions using predefined sentence templates. For instance, the triplet (Gracie Tessa Howell, birth city, St. Louis, MO) is converted into the sentence: "Gracie Tessa Howell's birthplace is St. Louis, MO." Following (Allen-Zhu & Li, 2024b), each time the model encounters a biography, the five sentences is randomly shuffled, and a new sentence template is selected for each attribute from a set of five possible templates. We denote the dataset containing N biographies as SynBio-N. See Appendix H.2.1 for full details.

Evaluation. Denote a knowledge triplet (name, attribute, value) as $(\boldsymbol{n}, \boldsymbol{a}, \boldsymbol{v})$ and let $|\boldsymbol{v}|$ represent the number of tokens in the value $\boldsymbol{v}$. For evaluation, the model is prompted with the prefix of the templated sentence containing $\boldsymbol{n}$ and $\boldsymbol{a}$ and is asked to generate $|\boldsymbol{v}|$ tokens using greedy decoding. The triplet is successfully memorized if the generated text exactly matches $\boldsymbol{v}$. For example, to test whether the model has learned the triplet (Gracie Tessa Howell, birth city, St. Louis, MO), the prompt "Gracie Tessa Howell's birthplace is" is provided. The model is deemed to have memorized the fact if it generates "St. Louis, MO." We report the accuracy averaged over all individuals, attributes, and templates in the main text and defer the detailed results to Appendix G.3. In Appendix E, we go beyond simple memorization and experiment with real-world Wikipedia biographies, where the model is tasked with generalizing to evaluation prompts that rarely have exact matches in its training data.

Training Setup. Our experiments use the GPT-NeoX library (Andonian et al., 2023) and the Pythia model architecture (Biderman et al., 2023), with model sizes ranging from 14M to 1B and a sequence length of 2048. The default setup involves pre-training from scratch on a mixture of FineWeb-Edu and SynBio, using a batch size of 512 and the Warmup-Stable-Decay (WSD) learning rate schedule (Hu et al., 2024) with a peak learning rate of 10^{-3}. We also investigate the continual pre-training setup, which mimics data annealing phase where high-quality data are upweighted to improve model performance (Dubey et al., 2024; Blakeney et al., 2024; Feng et al., 2024; OLMo et al., 2025). Full details are provided in Appendix H.

B Ablation Studies

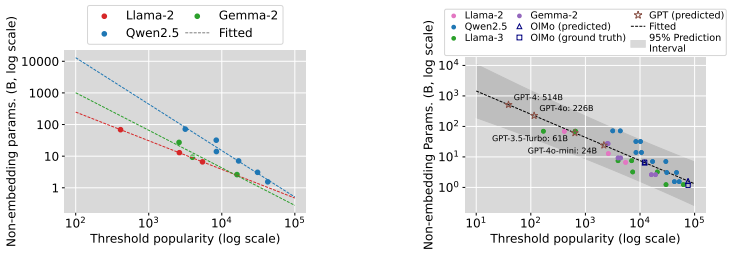


(a) Vary the batch size. (b) Vary the peak learning rate. (c) Vary the learning rate schedule. Both schedules use a peak learning rate of 10^{-3}.

Figure 4: Ablation studies on hyperparameters. The models exhibit consistent trends in knowledge acquisition across different batch sizes, learning rate values and schedules. All experiments are conducted by training 70M models on the mixture of FineWeb-Edu and SynBio-320k.

We now conduct ablation studies to demonstrate the robustness of our findings with respect to hyperparameters. We explore $r \in \{0.2, 0.4, 0.8\}$ and train 70M models for a total of 64B, 32B, and 16B tokens, respectively, ensuring each configuration passes SynBio the same number of times.

Consistent Trends Across Different Batch Sizes. As shown in Figure 4(a), we evaluate three batch sizes, $B \in \{256, 512, 1024\}$, for each r and observe consistent general trends across all batch sizes. For $r = 0.4$ and $r = 0.8$, smaller batch sizes yield slightly higher accuracies, likely due to the increased number of update steps. These experiments further distinguish between two types of



(a) Models likely trained on the same data mixture within each family.

(b) All models from five open source families.

Figure 5: Model size and threshold frequency exhibit a power-law relationship, which aligns with our theoretical prediction.

frequency at which the model encounters the knowledge dataset: per-token frequency and per-step frequency. For a fixed mixing ratio, doubling the batch size doubles the occurrences of each biography per step, while the occurrences per token remain unchanged. The results demonstrate that per-token frequency, rather than per-step frequency, determines training efficiency in knowledge acquisition.

Consistent trends across learning rate values and schedules. In Figure 4(b), we explore peak learning rates among $\{2.5 \times 10^{-4}, 10^{-3}, 4 \times 10^{-3}\}$ using the WSD scheduler. We observe that the trends are consistent across these values, although the learning process slows down at the lowest value 2.5×10^{-4}. In Figure 4(c), results for both cosine and WSD schedulers show similar trends.

C Warmup for Theoretical Analysis: Mixture of Facts

We start with a simple case where the data distribution $\mathcal{D}_\theta$ consists of K random facts. More formally, let $X_1, \ldots, X_K$ be K disjoint sets of input contexts and $y_1, \ldots, y_K$ be K target tokens. Each (X_i, y_i) represents a fact. For example, consider the fact "Donald Trump was born in the year 1946." The context x can be paraphrased in various ways, such as "Donald Trump's birth year is" or "Donald Trump came to this world in the year." However, all of these contexts lead to the same target token y, which, in this case, is "1946." We can group all possible paraphrases into a set X, and use the pair (X, y) to represent this fact. The universe samples $y_1, y_2, \ldots, y_K$ independently, where y_i is drawn from a fixed distribution $\mathcal{Y}_i$. Then the universe sets the latent variable θ to be $(y_1, y_2, \ldots, y_K)$ and $\mathcal{D}_\theta(y \mid x_i)$ to be a point mass at y_i for all $x_i \in X_i$. There could be other inputs x that can occur in $\mathcal{D}_\theta$, but the target tokens of such inputs are independent of θ.

Define the exposure frequency of each random fact, say the i-th, as the total probability that an input $x \in X_i$ occurs in $\mathcal{D}_\theta$. Formally, it is give by $\sum_{x' \in X_i} \mathbb{P}_\theta(x = x')$. Despite that the K facts have different entropies, the following theorem shows that if their exposure frequencies are the same, then the optimal bounded-capacity learner reduces the expected loss linearly with the capacity M, thus no phase transition in capacity.

Theorem C.1. *For all $M \geq 0$, if all the facts have the same exposure frequency p, then*

$$F_{\mathcal{P}}(M) = C + p \cdot \max\{H_{\text{tot}} - M, 0\}, \tag{3}$$

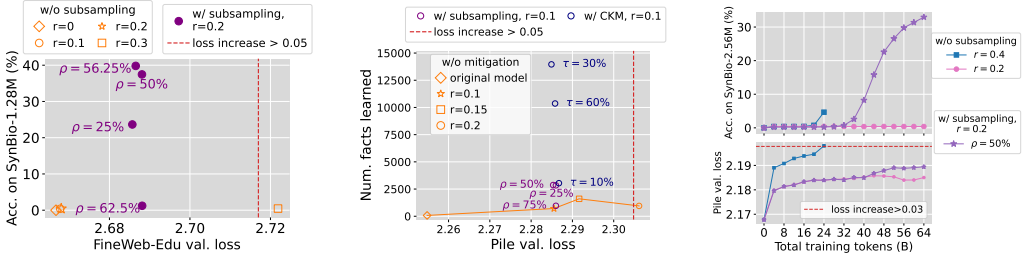
where $H_{\text{tot}} := \sum_{i=1}^{K} H(\mathcal{Y}_i)$ and $C := F_{\mathcal{P}}(\infty)$.

D Power-Law Relationship of Threshold Frequency and Model Size

Threshold Frequency for a Single Fact. For each fact in the first domain, its overall probability of being sampled is rp in the data mixture. Again arranging the terms in (2), we can further predict that for a single fact to be learned by the model, its frequency of appearing in the pre-training corpus should be larger than a threshold frequency f_{thres}, which scales with the model size following a power law:

$$f_{\text{thres}} \sim \frac{A}{M^{\alpha+1}}. \tag{4}$$

To validate our theoretical prediction of a power-law relationship between models size and threshold frequency, we examine the threshold frequency of a set of knowledge extracted from Wikipedia.



(a) 410M, trained from scratch on the mixture of FineWeb-Edu and SynBio-1.28M.

(b) 410M, continually pre-trained on the mixture of the Pile and WikiBio.

(c) 1B, continually pre-trainined on the mixture of the Pile and SynBio-2.56M.

Figure 6: Our mitigation strategies significantly boost knowledge acquisition while preserving models' general capability. For example, applying random subsampling and CKM to WikiBio improve the number of learned facts by 4 and 20 times, respectively. This is particularly surprising for subsampling, as it removes a significant proportion of the knowledge data but ends up with higher accuracy.

Specifically, we evaluate models on PopQA (Mallen et al., 2023), which contains 14k QA pairs derived from Wikidata triplets, along with monthly page view for corresponding Wikipedia articles. Since knowledge tested in PopQA can be structured as triplets, we consider them as homogeneous and expect them to exhibit similar threshold frequencies for a given model size.

Estimating the Threshold Frequency. Directly counting the frequency in the pre-training data can be challenging due to the sheer data volume (Kandpal et al., 2023). To address this, we follow Mallen et al. (2023) and use Wikipedia page views as a proxy for popularity, which is expected to be roughly proportional to how frequently the knowledge appears in web data. To estimate the threshold popularity P_{thres}, we determine the smallest popularity P such that the model's accuracy on data with popularity above P meets the target accuracy α_{target}. In our experiments, we set $\alpha_{\text{target}} = 60\%$. See Appendix H.4 for details.

Threshold frequency and model size follow a power-law relationship. We begin by examining base models from three families: Llama-2 (Touvron et al., 2023), Qwen-2.5 (Qwen et al., 2024) and Gemma-2 (Team et al., 2024). According to their technical reports, models from the same family are likely trained on the same data mixture. As shown in Figure 5(a), $\log P_{\text{thres}}$ generally decreases linearly as log model size increases within each family. The slope may vary across model families, as the exponent for model size in the loss scaling law can differ depending on model architecture and training data. Next, we relax the constraint of training on the same data mixture and investigate the overall trend between model size and P_{thres}. We add the Llama-3 (Dubey et al., 2024) family, and evaluate both base and instruction-tuned models for all families, totaling 30 models. Interestingly, in Figure 5(b), log model size and $\log P_{\text{thres}}$ also exhibit a linear relationship, with most models falling within the 95% confidence interval. We further use models from the OLMo (Groeneveld et al., 2024) family as a validation set, where predictions of the fitted power-law closely match the ground truth.

Potential Application: Inferring the Size of Proprietary Models. The identified power-law relationship offers a potential method for estimating the size of proprietary models, such GPTs. As a preliminary attempt, we estimate the threshold popularity for GPT-3.5-Turbo, GPT-4, GPT-4o, and GPT-4o-mini. Applying the fitted power law yields size predictions of 61B, 514B, 226B and 24B, respectively. The 95% confidence intervals are 12–314B, 80–3315B, 39–1313B, and 5–118B, respectively.

E Mitigation Strategies

While previous sections focus on how many facts the model memorizes, we now consider a more practical scenario where both factual accuracy and models' general capabilities are important. In this context, the model's extremely slow acquisition of low-frequency facts presents a dilemma: using a small r blows up the training steps to achieve the desired factual accuracy, while recklessly increasing r degrades the model's general capabilities. One could also imagine when multiple knowledge-dense datasets were mixed together to form a carefully balanced mixture, setting a large r could be even more detrimental. Inspired by our theory, we propose two simple yet effective mitigation strategies:

1. **Random Subsampling**: Randomly subsample the knowledge dataset to accelerate knowledge acquisition.
2. **Compact Knowledge Mixing (CKM)**: Rephrase the knowledge in a more compact form and add rephrased data to the original dataset while keeping the overall mixing ratio fixed.

We validate on both SynBio and a new real-world knowledge dataset, WikiBio, that these strategies significantly boost the accuracy on knowledge dataset. For example, applying subsampling and CKM to WikiBio improve the number of learned facts by 4 and 20 times, respectively. This is particularly surprising for subsampling, as it removes a significant proportion of the knowledge data but ends up with higher accuracy. Below, we first introduce the real-world dataset that complements SynBio.

E.1 Real-World Knowledge Data: WikiBio

The WikiBio Dataset. To extend our study to a more real-world scenario, we curate WikiBio, a dataset containing Wikipedia biographies along with ten paraphrased versions of the first paragraph for 275K individuals, totaling 453M tokens. We ensure that the key information—name, occupation, and birth date—is mentioned within the first paragraph. Experimenting with this dataset reflects the case where one aims to guarantee that the model can generate accurate answers to factual inquires about famous people. This task is more challenging as Wikipedia biographies comprise diverse texts lack of uniform formats, requiring the model to generalize to prompts that rarely have exact matches in the training data. See Appendix H.2.2 for full details.

Evaluation. We assess whether the model can recall a person's birth date, using this as a proxy for how well it memorizes the person's information. Specifically, for a (name, occupation, birth date) triplet, we prompt the model with "The {`occupation`} {`name`} was born on" and consider the response correct if it accurately includes the birth year and month in the generated text. The occupation is included not only to create out-of-distribution prompts but also to provide additional context and assist in disambiguation.

E.2 Strategy 1: Random Subsampling

The approach of random subsampling may seem counterintuitive at first glance, but it becomes reasonable if we consider how the threshold mixing ratio r_{thres} relates to the exposure frequency of each fact within the knowledge-dense dataset, denoted as p. For a dataset that contains only S facts with uniform probability, $p \propto 1/S$ and we can derive from (2) that the threshold mixing ratio is $r_{\text{thres}} \sim \frac{AS}{M^{\alpha+1}}$. Subsampling reduces S and thus lowers the threshold mixing ratio, allowing the model to achieve much higher accuracy on the subsampled dataset. An alternative view is to think of the frequency of occurrence for each single fact f, which follows $f \propto 1/S$ for fixed mixing ratio r. By subsampling, we increase the frequency of occurrence for each retained fact, pushing it above the threshold frequency f_{thres}. As a result, the model memorizes more facts within the same number of training steps. In the following text, we use ρ to represent the subsampling ratio.

Experimental Setup. We study both pre-training from scratch and continual pre-training setups. To evaluate the model's general capabilities, we use its validation loss on the web data (the Pile or FineWeb-Edu) and its zero-shot performance on downstream tasks. The selected downstream tasks include LAMBADA (Paperno et al., 2016), ARC-E (Clark et al., 2018), PIQA (Bisk et al., 2020), SciQ (Welbl et al., 2017), and HellaSwag (Zellers et al., 2019), covering core capabilities such as text understanding, commonsense reasoning, and question answering. We compare the validation loss and average downstream performance to the model trained with $r = 0$ (no knowledge-dense data) in the pre-training-from-scratch setup and to the original Pythia model in the continual pre-training setup. Downstream performance drop of more than 2% is considered unacceptable.

Subsampling enables faster fact memorization while maintaining general capability. We train 410M models from scratch on the mixture of FineWeb-Edu and SynBio-1.28M using mixing ratios $r \in \{0, 0.1, 0.2, 0.3\}$ for a total of 32B tokens. As shown in Figures 6(a) and 8(a), FineWeb-Edu validation loss and downstream performance worsen as r increases, with the degradation becoming unacceptable at $r = 0.3$, where downstream accuracy drops by 2.09% and loss increase exceeds 0.05. Despite this performance decline, SynBio accuracy remains near zero. Subsampling effectively mitigates this issue. Specifically, subsampling SynBio-1.28M to 25%, 50%, and 56.25% significantly improves SynBio accuracy from near 0% to 23.53%, 37.46%, and 39.81%, respectively, while maintaining downstream performance within the acceptable range. Note that further increasing ρ to 62.5% makes the frequency of each biography too low, resulting in SynBio accuracy dropping back to near zero. See training details in Appendix H.5 and detailed performance in Tables 1(b) and 2(a).

Consistent Results for Continual Pre-training. We extend our analysis to the continual pre-training setup, where we continually pre-train the 410M and 1B Pythia models from their 100k-step checkpoints by mixing Pile with WikiBio and SynBio-2.56M, respectively. We train 410M models for a total of 32B tokens and 1B models for 64B tokens. Since mixing in the knowledge-dense data introduces a distribution shift, the Pile validation loss may increase with training due to catastrophic forgetting (Ibrahim et al., 2024). To keep the model's general capabilities in the acceptable range, we apply early stopping when Pile validation loss increases by 0.05 for 410M models and 0.03 for 1B models, both corresponding to approximately 2% drop in downstream performance. As shown in Figures 6(b) and 8(b), without subsampling, setting $r = 0.1$ and $r = 0.15$ results in slow learning of WikiBio. On the other hand, increasing r to 0.2 causes the Pile validation loss to grow during training, leading to early stopping after 20B tokens, resulting in poor WikiBio performance. By contrast, subsampling WikiBio to 25% or 50% significantly accelerates knowledge acquisition while keeping Pile validation loss within the acceptable range. For example, when fixing $r = 0.1$, setting ρ to 50% improves the number of learned facts by 4 times. Further increasing ρ to 75% proves to be too high, resulting in poor performance. Similar conclusions can be drawn from experiments with the 1B models, where subsampling SynBio to 50% at $r = 0.2$ significantly outperforms both $r = 0.2$ and the early-stopped $r = 0.4$ without subsampling, achieving a margin of approximately 30% (see Figure 6(c)). We defer the training details to Appendix H.5 and detailed performance to Tables 1(c), 2(b) and 3.

E.3 Strategy 2: Compact Knowledge Mixing (CKM)

The second strategy involves rephrasing knowledge in compact forms (e.g., tuples) and adding these rephrased forms to the original dataset. To explain why CKM helps, consider the frequency of occurrence f for each fact, which, for fixed r, is inversely proportional to the average number of tokens required to represent each fact. By adding compact representations, the average token count for each fact decreases, pushing f above the threshold f_{thres}. For our specific WikiBio dataset, we compress the key information—name, birth date, and occupation—into a tuple format represented as "Bio: N {`name`} B {`birth date`} O {`occupation`}". We keep adding these tuple-form data points to WikiBio until their token count reaches τ times the total token count of the original dataset. We name this ratio τ as Compact Knowledge Mixing (CKM) ratio.

Experimental Setup. We apply CKM to WikiBio to validate its effectiveness, with the same continual pre-training setup as in Appendix E.2. Each time models encounter the tuple-form data point, the order of birth date and occupation is randomly flipped. We apply early stopping when Pile validation loss increases by 0.05.

CKM significantly improves knowledge acquisition efficiency while preserving general capability. We explore CKM ratios $\tau \in \{0.1, 0.3, 0.6\}$, fixing the overall mixing ratio $r = 0.1$. As shown in Figure 6(b) and Figure 8(c), CKM keeps the general capability within the acceptable range and consistently boosts knowledge acquisition. Notably, performance on WikiBio improves fourfold even when the token count of the added tuple-form data points constitutes only 10% of the original dataset. This aligns with the phase transition in frequency predicted by our theory. Increasing τ to 30% further boosts the number of learned facts to 20 times compared with training without CKM. Notice that WikiBio performance saturates as τ reaches 90%, indicating that τ should be carefully chosen to balance memorization and generalization. Detailed downstream performance is provided in Table 4.

F Additional Related Works

Knowledge Capacity Scaling Law. LLMs are typically trained on a vast amount of data that are rich in knowledge, and extensive studies have investigated how much knowledge LLMs can acquire from the training data. Pioneering studies (Petroni et al., 2019; Roberts et al., 2020; Da et al., 2021) demonstrate that LLMs can capture a substantial amount of knowledge, suggesting their potential as knowledge bases. To quantify the relationship between model size and knowledge storage, Allen-Zhu & Li (2024a) and Lu et al. (2024) discover a linear relationship between models' knowledge capacity and their parameter count by training LLMs on data only containing fixed-format knowledge for sufficiently long horizons. Later, Nichani et al. (2025) formally proved this linear relationship. In contrast, this paper examines the data mixing scenario and demonstrates that this linear scaling can

be disrupted when the knowledge-dense dataset is mixed with vast amounts of web-scraped data. Another important factor is the frequency of occurrence for knowledge.

Impact of Frequency on Knowledge Acquisition. This paper identifies phase transitions in knowledge acquisition within data mixtures with respect to model size and mixing ratio. Some relevant observations can be found in previous papers, but we takes a more direct and systematic approach. Kandpal et al. (2023); Mallen et al. (2023); Sun et al. (2024) find that LLMs can perform poorly on low-frequency knowledge. Ghosal et al. (2024) show that frequency of knowledge in the pre-training data determines how well the model encodes the knowledge, which influences its extractability after QA fine-tuning. Taking a more microscopic view, Chang et al. (2024) insert a few pieces of new knowledge during training and track their loss. By fitting a forgetting curve, they conjecture that the model may fail to learn the knowledge if its frequency is lower than some threshold.

Memorization and Forgetting. Our findings also relate to prior observations on the memorization and forgetting behaviors of LLMs, but we explicitly characterize phase transitions in the context of data mixing. Carlini et al. (2023) show that memorization of training data follows a log-linear relationship with model size, the number of repetitions, and prompt length. Biderman et al. (2024) take a data point-level perspective and demonstrate that it is difficult to predict whether a given data point will be memorized using a smaller or partially trained model. By injecting a few new sequences into the training data, Huang et al. (2024) find that a sequence must be repeated a non-trivial number of times to be memorized. By examining training dynamics, Tirumala et al. (2022) observe that memorization can occur before overfitting and that larger models memorize faster while forgetting more slowly. From a theoretical perspective, Feldman (2020) prove that memorization of training labels is necessary to achieve near-optimal generalization error for long-tailed data distributions.

Scaling laws for Data Mixing. LLM performance is significantly influenced by the mixing proportions of the training data from different domains. Our paper is related to a line of studies that optimize the mixing proportions by modeling LLM performance as a function of the mixing proportions (Liu et al., 2024; Kang et al., 2024; Ye et al., 2024; Ge et al., 2024). However, their datasets can be highly heterogeneous even within a single domain (e.g., OpenWebText, Pile-CC) while we focus on mixing a uniform, knowledge-dense dataset into web-scraped data.

G Additional Experimental Results

G.1 Additional Plot for Phase Transition in Mixing Ratio

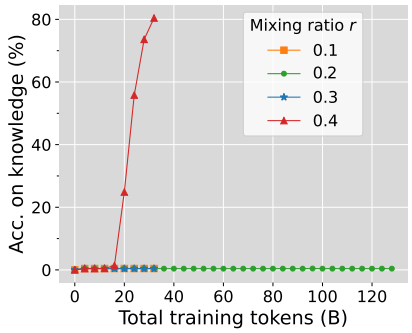
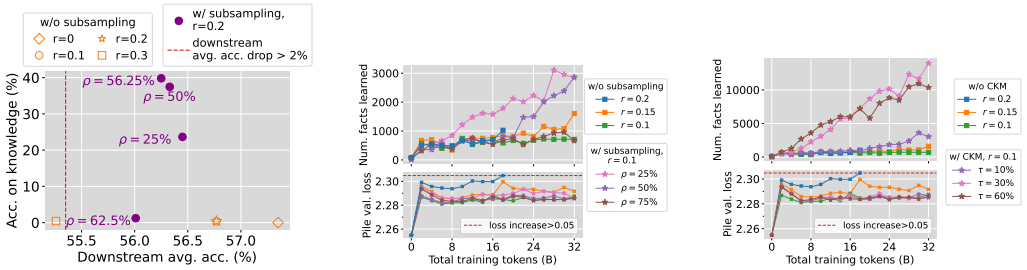


Figure 7: For 410M models trained on the mixture of FineWeb-Edu and SynBio-1.28M, accuracy for $r = 0.2$ remains near zero even when we extend the training by 4 times.

G.2 Additional Plots for Mitigation Strategies



(a) 410M, trained from scratch on the mixture of FineWeb-Edu and SynBio-1.28M.

(b) Training trajectory for applying subsampling to WikiBio.

(c) Training trajectory for applying CKM to WikiBio.

Figure 8: Additional plots for mitigation strategies.

G.3 Detailed Performance on SynBio

In Table 1(a), we detail the accuracy of each attribute for 70M models trained on the mixture of FineWeb-Edu and SynBio-320k with $r \in \{0.2, 0.4, 0.8\}$, trained for 64B, 32B, and 16B tokens respectively. We notice that the accuracy for birth date is lower than other attributes. This can be attributed to the complexity of precisely recalling the combined elements of day, month, and year information, which together form a much larger domain than other attributes. To maintain clarity and conciseness, we omit the detailed performance in other 70M experiments, as this pattern persists across them.

Furthermore, we present the detailed performance of 410M models on SynBio-1.28M corresponding to Figure 6(a) in Table 1(b). We also provide the detailed performance of 1B models on SynBio-2.56M corresponding to Figure 6(c) in Table 1(c).

Table 1: Detailed performance on SynBio. We report the accuracy (%) for each attribute averaged over five templates.

(a) 70M model, pre-trained from scratch on the mixture of FineWeb-Edu and SynBio-320k.

r	Birth date	Birth city	University	Major	Employer	Avg.
Random guess	0.00	0.50	0.33	1.00	0.38	0.44
0.2	0.00	0.63	0.43	1.12	0.38	0.51
0.4	16.96	45.67	41.03	50.78	43.93	39.68
0.8	79.76	88.64	88.55	90.10	88.30	87.07

(b) 410M model, pre-trained from scratch on the mixture of FineWeb-Edu and SynBio-1.28M.

N	ρ (%)	r	Birth date	Birth city	University	Major	Employer	Avg.
Random guess			0.00	0.50	0.33	1.00	0.38	0.44
-	-	0.00	0.00	0.00	0.00	0.00	0.00	0.00
1.28M	100	0.1	0.00	0.42	0.33	1.01	0.21	0.39
1.28M	100	0.2	0.00	0.45	0.34	1.09	0.22	0.42
1.28M	100	0.3	0.00	0.49	0.35	1.14	0.25	0.45
320k	25	0.2	22.34	23.98	23.64	24.03	23.65	23.53
640k	50	0.2	27.97	39.66	38.51	41.50	39.68	37.46
720k	56.25	0.2	28.02	42.94	42.15	44.07	41.88	39.81
800k	62.5	0.2	0.01	1.16	0.85	3.19	0.89	1.22

(c) 1B model, continually pre-trained on the mixture of the Pile and SynBio-2.56M. Note that $r = 0.4$ is early stopped due to its Pile validation loss increasing beyond the acceptable range.

N	ρ (%)	r	Training tokens (B)	Birth date	Birth city	University	Major	Employer	Avg.
Random guess				0.00	0.50	0.33	100	0.38	0.44
Pythia-1B-100k-ckpt				0.00	0.00	0.00	0.00	0.00	0.00
2.56M	100	0.2	64	0.01	0.46	0.33	0.98	0.21	0.39
2.56M	100	0.4	24	0.05	10.95	3.90	4.74	3.64	4.66
1.28M	50	0.2	64	23.95	34.55	35.05	35.96	35.19	32.94

G.4 Detailed Downstream Performance

We employ the `lm-eval-harness` (Gao et al., 2024) codebase to evaluate the zero-shot performance on five downstream tasks, including LAMBADA, ARC-E, Sciq, PIQA, and HellaSwag. We compute the validation loss on about 50M tokens on a holdout set from the Pile or FineWeb-Edu. The detailed downstream performance and validation loss for applying the random subsampling strategy to SynBio and WikiBio are presented in Tables 2 and 3, respectively. Additionally, we report the detailed downstream results for applying CKM to WikiBio in Table 4.

Table 2: Detailed downstream performance and validation loss for applying the random subsampling strategy to SynBio. We report the accuracy (%) and standard deviation (%) in the format $\text{acc.}_{(\text{std. dev.})}$ for each downstream task.

(a) 410M model, train from scratch.

N	ρ (%)	r	LAMBADA	ARC-E	Sciq	PIQA	HellaSwag	Avg.	FineWeb-Edu val. loss
-	-	0	$38.25_{(0.68)}$	$61.83_{(1.00)}$	$83.60_{(1.17)}$	$68.01_{(1.09)}$	$35.04_{(0.48)}$	57.35	2.667
1.28M	100	0.1	$34.56_{(0.66)}$	$62.33_{(0.99)}$	$83.50_{(1.17)}$	$68.34_{(1.09)}$	$35.13_{(0.48)}$	56.77(↓ 0.58)	2.668(↑ 0.001)
1.28M	100	0.2	$34.43_{(0.67)}$	$62.13_{(0.99)}$	$83.80_{(1.17)}$	$68.12_{(1.09)}$	$35.39_{(0.48)}$	56.77(↓ 0.58)	2.668(↑ 0.001)
1.28M	100	0.3	$33.94_{(0.66)}$	$60.77_{(1.00)}$	$80.80_{(1.25)}$	$66.54_{(1.10)}$	$34.23_{(0.47)}$	55.26(↓ 2.09)	2.722(↑ 0.054)
320k	25	0.2	$36.70_{(0.67)}$	$60.35_{(1.00)}$	$(82.70_{1.20})$	$67.74_{(1.09)}$	$34.76_{(0.48)}$	56.45(↓ 0.90)	2.686(↑ 0.019)
640k	50	0.2	$36.58_{(0.67)}$	$60.61_{(1.00)}$	$83.30_{(1.18)}$	$66.65_{(1.10)}$	$34.53_{(0.47)}$	56.33(↓ 1.02)	2.688(↑ 0.021)
720k	56.25	0.2	$35.61_{(0.67)}$	$60.94_{(1.00)}$	$83.00_{(1.19)}$	$67.14_{(1.10)}$	$34.54_{(0.47)}$	56.25(↓ 1.10)	2.687(↑ 0.020)
800k	62.5	0.2	$35.20_{(0.67)}$	$60.48_{(1.00)}$	$83.40_{(1.20)}$	$66.54_{(1.10)}$	$34.45_{(0.47)}$	56.01(↓ 1.34)	2.688(↑ 0.021)

(b) 1B model, continually pre-trained. Note that $r = 0.4$ is early stopped due to its Pile validation loss increasing beyond the acceptable range.

N	ρ (%)	r	Training tokens (B)	LAMBADA	ARC-E	Sciq	PIQA	HellaSwag	Avg.	Pile val. loss
Pythia-1B-100k-ckpt				$55.66_{(0.69)}$	$54.50_{(1.02)}$	$83.00_{(1.19)}$	$70.78_{(1.06)}$	$36.97_{(0.48)}$	60.18	2.168
2.56M	100	0.2	64	$53.68_{(0.69)}$	$51.47_{(1.03)}$	$81.00_{(1.24)}$	$68.77_{(1.08)}$	$35.91_{(0.48)}$	58.17(↓ 2.01)	2.184(↑ 0.016)
2.56M	100	0.4	24	$52.38_{(0.70)}$	$51.47_{(1.03)}$	$80.70_{(1.25)}$	$68.17_{(1.09)}$	$34.95_{(0.48)}$	57.53(↓ 2.65)	2.198(↑ 0.030)
1.28M	50	0.2	64	$54.71_{(0.69)}$	$52.86_{(1.02)}$	$81.30_{(1.23)}$	$68.99_{(1.08)}$	$35.48_{(0.48)}$	58.67(↓ 1.51)	2.189(↑ 0.022)

Table 3: Detailed downstream performance for applying the random subsampling strategy to WikiBio. We use ρ to denote the subsampling ratio. We report the accuracy (%) and standard deviation (%) in the format $\text{acc.}_{(\text{std. dev.})}$ for each downstream task. Note that $r = 0.2$ is early stopped due to its Pile validation loss increasing beyond the acceptable range.

N	ρ (%)	r	Training tokens (B)	LAMBADA	ARC-E	Sciq	PIQA	HellaSwag	Avg.	Pile val. loss
Pythia-1B-100k-ckpt				$50.86_{(0.70)}$	$52.10_{(1.03)}$	$83.70_{(1.17)}$	$67.14_{(1.10)}$	$34.09_{(0.47)}$	57.58	2.255
277k	100	0.1	32	$50.77_{(0.70)}$	$48.95_{(1.03)}$	$80.80_{(1.25)}$	$66.43_{(1.10)}$	$33.16_{(0.47)}$	56.02(↓ 1.56)	2.286(↑ 0.031)
277k	100	0.15	32	$49.12_{(0.70)}$	$49.66_{(1.03)}$	$81.80_{(1.22)}$	$66.38_{(1.10)}$	$32.84_{(0.47)}$	55.96(↓ 1.62)	2.292(↑ 0.037)
277k	100	0.2	20	$49.37_{(0.70)}$	$49.87_{(1.03)}$	$79.70_{(1.27)}$	$65.40_{(1.11)}$	$33.08_{(0.47)}$	55.48(↓ 2.10)	2.306(↑ 0.051)
69k	25	0.1	32	$48.63_{(0.70)}$	$50.59_{(1.03)}$	$81.00_{(1.24)}$	$66.49_{(1.10)}$	$33.16_{(0.47)}$	55.97(↓ 1.54)	2.286(↑ 0.031)
137k	50	0.1	32	$50.30_{(0.70)}$	$50.38_{(1.03)}$	$78.80_{(1.29)}$	$66.27_{(1.10)}$	$33.16_{(0.47)}$	55.78(↓ 1.80)	2.285(↑ 0.030)
208k	75	0.1	32	$50.34_{(0.70)}$	$49.20_{(1.03)}$	$80.10_{(1.26)}$	$66.97_{(1.10)}$	$33.19_{(0.47)}$	55.96(↓ 1.62)	2.286(↑ 0.031)

Table 4: Detailed downstream performance for applying the compact knowledge mixing strategy on WikiBio. We use τ to denote the CKM ratio. We report the accuracy (%) and standard deviation (%) in the format $\text{acc.}_{(\text{std. dev.})}$ for each downstream task. Note that $r = 0.2$ is early stopped due to its Pile validation loss increasing beyond the acceptable range.

r	τ (%)	Training tokens (B)	LAMBADA	ARC-E	Sciq	PIQA	HellaSwag	Avg.	Pile val. loss
Pythia-1B-100k-ckpt			$50.86_{(0.70)}$	$52.10_{(1.03)}$	$83.70_{(1.17)}$	$67.14_{(1.10)}$	$34.09_{(0.47)}$	57.58	2.255
0.1	0	32	$50.77_{(0.70)}$	$48.95_{(1.03)}$	$80.80_{(1.25)}$	$66.43_{(1.10)}$	$33.16_{(0.47)}$	56.02(↓ 1.56)	2.286(↑ 0.031)
0.15	0	32	$49.12_{(0.70)}$	$49.66_{(1.03)}$	$81.80_{(1.22)}$	$66.38_{(1.10)}$	$32.84_{(0.47)}$	55.96(↓ 1.62)	2.292(↑ 0.037)
0.2	0	20	$49.37_{(0.70)}$	$49.87_{(1.03)}$	$79.70_{(1.27)}$	$65.40_{(1.11)}$	$33.08_{(0.47)}$	55.48(↓ 2.10)	2.306(↑ 0.051)
0.1	10	32	$49.70_{(0.70)}$	$49.54_{(1.03)}$	$80.40_{(1.26)}$	$66.32_{(1.10)}$	$33.11_{(0.47)}$	55.81(↓ 1.77)	2.287(↑ 0.032)
0.1	30	32	$50.11_{(0.70)}$	$49.12_{(1.03)}$	$80.20_{(1.26)}$	$66.54_{(1.10)}$	$33.11_{(0.47)}$	55.82(↓ 1.76)	2.285(↑ 0.030)
0.1	60	32	$49.99_{(0.70)}$	$49.41_{(1.03)}$	$80.00_{(1.27)}$	$65.78_{(1.11)}$	$32.99_{(0.47)}$	55.63(↓ 1.76)	2.286(↑ 0.031)

H Experimental Details

H.1 General Setup

Hyperparameters. For each training setup, we specify the batch size and learning rate in the respective sections, while maintaining other hyperparameters consistent with those used in Pythia. For both WSD and cosine schedulers, we use 160 steps for linear warmup. When employing the WSD scheduler, we allocate the final 10% steps for cooldown.

Hardware. We train models of sizes 70M and 160M using 8 NVIDIA RTX 6000 Ada GPUs, while models of sizes 410M and 1B are trained using either 16 NVIDIA RTX 6000 Ada GPUs or 8 NVIDIA A100 GPUs. The estimated runtime required to train each model size on 1B tokens is detailed in Table 5. Consequently, a typical run training a 410M model on 32B tokens takes approximately 32 hours, whereas the longest run, which trains a 1B model on 64B tokens, exceeds five days.

Table 5: Estimated runtime required to train each model size on 1B tokens on our hardware.

Model size	Hardware	Runtime (h) per billion tokens.
70M	8xNVIDIA RTX 6000 Ada	0.25
160M		0.70
410M	16xNVIDIA RTX 6000 Ada	1.0
1B	or 8xNVIDIA A100	2.0

Implementation of data mixing. During training, when loading each sample, the model flips a biased coin: with probability r, it loads from the SynBio dataset, and with probability $1 - r$, it loads from the web-scraped data (FineWeb-Edu or the Pile).

H.2 Details of Dataset Construction

H.2.1 Constructing the SynBio Dataset

To generate names, we collect a list of 400 common first names, 400 common middle names, and 1000 common last names, resulting in 1.6×10^8 unique names. To generate SynBio-N, we sample N names from this set without replacement. For each individual, the value for each attribute is randomly assigned as follows: birth date (1–28 days × 12 months × 100 years spanning 1900–2099), birth city (from 200 U.S. cities), university (from 300 institutions), major (from 100 fields of study), and employer (from 263 companies). Each attribute is paired with five sentence templates, which are used to convert (name, attribute, value) triplets into natural text descriptions. A complete list of sentence templates is provided in Table 6, and an example of a synthetic biography can be found in Table 7.

Table 6: Sentence templates to generate the SynBio Dataset.

Attribute	Template
Birth date	{name} was born on {birth date}. {name} came into this world on {birth date}. {name}'s birth date is {birth date}. {name}'s date of birth is {birth date}. {name} celebrates {possessive pronoun} birthday on {birth date}.
Birth city	{name} spent {possessive pronoun} early years in {birth city}. {name} was brought up in {birth city}. {name}'s birthplace is {birth city}. {name} originates from {birth city}. {name} was born in {birth city}.
University	{name} received mentorship and guidance from faculty members at {university}. {name} graduated from {university}. {name} spent {possessive pronoun} college years at {university}. {name} completed {possessive pronoun} degree at {university}. {name} completed {possessive pronoun} academic journey at {university}.
Major	{name} completed {possessive pronoun} education with a focus on {major}. {name} devoted {possessive pronoun} academic focus to {major}. {name} has a degree in {major}. {name} focused {possessive pronoun} academic pursuits on {major}. {name} specialized in the field of {major}.
Employer	{name} is employed at {employer}. {name} a staff member at {employer}. {name} is associated with {employer}. {name} is engaged in work at {employer}. {name} is part of the team at {employer}.

Table 7: An example of a synthetic biography. The values that we expect the model to recall during evaluation are underlined.

Gracie Tessa Howell's birth date is August 09, 1992. Gracie Tessa Howell's birthplace is St. Louis, MO. Gracie Tessa Howell received mentorship and guidance from faculty members at Santa Clara University. Gracie Tessa Howell has a degree in Robotics. Gracie Tessa Howell is engaged in work at Truist Financial.

H.2.2 CONSTRUCTING THE WIKIBIO DATASET

To create the WikiBio dataset, we query Wikidata to gather names and birth dates of individuals from 16 common occupations, identify each person's biography on Wikipedia by matching the name with the Wikipedia document title, and retain only those biographies where the person's occupation and birth date are mentioned within the first paragraph. The detailed composition is listed in Table 8. Inspired by Allen-Zhu & Li (2023), we employ Llama-3.1-70B-Instruct to paraphrase the first paragraph of each biography ten times as a form of data augmentation. See Table 9 for the prompt for paraphrasing. An example of the original text and the paraphrased versions are presented below.

- **Original text**: Rebecca Jo Budig (born June 26, 1973) is an American actress and television presenter. Her career began in 1993, and in 1995, she was cast in the role of Michelle Bauer on the CBS soap opera Guiding Light. In 1999, she was cast as Greenlee Smythe on the ABC soap opera All My Children; she held the role off-and-on until the network series finale in 2011. In 2015, she was cast in the role of Hayden Barnes on General Hospital. In 2019, Budig was cast on L.A.'s Finest, as Carlene Hart, the drug trafficking-soccer mom of two young school-age children, a boy and a girl.
- **Paraphrased versions**:
  1. Born on June 26, 1973, Rebecca Jo Budig is an American television presenter and actress with a career spanning nearly three decades. Her professional journey began in 1993 and has included notable roles such as Michelle Bauer on Guiding Light and Greenlee Smythe on All My Children. After playing the latter role on-and-off until 2011, she went on to portray Hayden Barnes in General Hospital in 2015 and Carlene Hart in the 2019 television series L.A.'s Finest.
  2. With a diverse career in television, Rebecca Jo Budig, born June 26, 1973, has established herself as a talented actress and presenter. Her career milestones include her roles as Michelle Bauer in the CBS soap opera Guiding Light, and Greenlee Smythe in All My Children. Her portrayal of Greenlee spanned several years, concluding with the show's finale in 2011. Budig continued to expand her repertoire with roles in General Hospital and L.A.'s Finest.
  3. Rebecca Jo Budig is a versatile American actress and television host, born on June 26, 1973. Since her career began in 1993, she has landed prominent roles in several television series. One of her earliest notable roles was Michelle Bauer in Guiding Light, followed by her portrayal of Greenlee Smythe in All My Children, a character she played until the series' conclusion in 2011. In the years that followed, she appeared in General Hospital and L.A.'s Finest.
  4. Rebecca Jo Budig, an American actress and television presenter, was born on June 26, 1973. She began her career two decades later, securing the role of Michelle Bauer on Guiding Light. Budig's subsequent roles have included Greenlee Smythe on All My Children, a part she played intermittently until the series ended in 2011. Her later appearances include a role in General Hospital and as Carlene Hart in the series L.A.'s Finest.
  5. American actress Rebecca Jo Budig was born on June 26, 1973. Her television career, which began in 1993, encompasses multiple notable roles, such as Michelle Bauer on the soap opera Guiding Light and Greenlee Smythe on All My Children. She portrayed the latter character until the series finale in 2011. Budig later appeared as Hayden Barnes in General Hospital and took on the role of Carlene Hart in L.A.'s Finest.
  6. Since launching her career in 1993, Rebecca Jo Budig has established herself as a talented actress and television presenter in the United States. Born on June 26, 1973, she has appeared in a range of notable roles, including Michelle Bauer on Guiding Light and Greenlee Smythe on All My Children. The latter role spanned several years, concluding with the show's finale in 2011. Her subsequent appearances include General Hospital and L.A.'s Finest.
  7. Rebecca Jo Budig, born on June 26, 1973, has enjoyed a successful career in American television as an actress and presenter. Her breakout role came in 1995 when she was cast as Michelle Bauer on Guiding Light. Later, she played the character Greenlee Smythe on All My Children, a part she held intermittently until the show's finale in 2011. Her more recent roles include appearances in General Hospital and as Carlene Hart in L.A.'s Finest.
  8. Born on June 26, 1973, Rebecca Jo Budig is a talented American actress and television presenter. Since her career began in 1993, she has appeared in various television series. Notable roles include her portrayal of Michelle Bauer on the soap opera Guiding Light, as

well as Greenlee Smythe on All My Children. Budig continued to expand her acting repertoire with roles in General Hospital and L.A.'s Finest, including her portrayal of Carlene Hart.

9. As an American actress and television host, Rebecca Jo Budig has had a diverse career spanning nearly three decades. Born on June 26, 1973, she began her professional journey in 1993. Her notable roles include Michelle Bauer on Guiding Light and Greenlee Smythe on All My Children, a character she played until the series finale in 2011. Her subsequent appearances include General Hospital and the series L.A.'s Finest, where she portrayed Carlene Hart.
10. With a career in television that began in 1993, Rebecca Jo Budig, born June 26, 1973, has established herself as a versatile actress and presenter. Her early roles include Michelle Bauer on Guiding Light, while her breakout role came as Greenlee Smythe on All My Children. She continued to portray Greenlee intermittently until the show's finale in 2011. Her later roles include appearances in General Hospital and L.A.'s Finest, where she took on the role of Carlene Hart.

Table 8: Detailed Composition of WikiBio.

Occupation	Num. Wikipedia biographies
Singer	18,482
Actor	31,846
Politician	38,653
Businessperson	8,068
Mathematician	5,093
Physicist	4,296
Writer	26,746
Football player	56,547
Basketball player	16,956
Sport shooter	3,156
Tennis plater	7,602
Swimmer	9,108
Painter	12,927
Volleyball player	3,556
Composer	13,719
Athlete	18,013
Total	274,768

Table 9: The prompt for paraphrasing the first paragraph of Wikipedia documents.

```
I am creating the training data for an LLM. I would like
↪  to teach it to flexibly extract knowledge from a
↪ Wikipedia paragraph. Therefore, I want to diversify
↪ the Wikipedia paragraphs as much as possible so that
↪ the model can learn the actual relationships between
↪ entities, rather than just memorizing the text. Please
↪  assist with the paraphrasing task. Paraphrase the
↪ following Wikipedia paragraph about {Wikipedia
document title} 10 times. Aim to make the paraphrased
↪ versions as varied as possible. Ensure all essential
↪ information is retained, particularly the information
↪ about the birthday and the occupuation.
```

H.3 Details of the Fitting Process

We use T to denote the required training steps to reach 40% accuracy and r to denote the mixing ratio.

Fitting the exponential function. We fit T with respect to r for all $r \geq 0.3$ using the function $T(r) = A\exp(B/r)$, where A and B are coefficients to be fitted. Taking logarithmic on both sides, we obtain a linear function $\log T = \log A + B/r$. By fitting $\log T$ against $1/r$ with linear regression, we obtain $\log A \approx -0.25512$, $B \approx 1.5137$ with goodness-of-fit $R^2 = 0.9980$.

Fitting the power-law function. We fit T with respect to r for all $r \in \{0.3, 0.4, 0.45, 0.5, 0.55\}$ using the function $T(r) = Cr^{-D}$, where C and D are coefficients to be fitted. Taking logarithmic on both sides, we obtain a linear function $\log T = \log C - D\log r$. By fitting $\log T$ against $\log r$ with linear regression, we obtain $C \approx 0.098158$, $D \approx 3.83878$ with goodness-of-fit $R^2 = 0.9853$.

Table 10: The prompt for evaluating models on PopQA.

```
You are a helpful assistant. I want to test your
↪ knowledge level. Here are a few examples.

{few shot examples text with templates}

Now, I have a question for you. Please respond in just a
↪  few words, following the style of the examples
↪ provided above.
```

H.4 Details of Estimating the Threshold Popularity

Following Mallen et al. (2023), we evaluate models using 15-shot prompting. We use the prompt presented in Table 10 for evaluatioin and allow models to generate up to 128 tokens with greedy decoding. To assess answer correctness, we employ Llama-3.1-8B-Instruct as a judge. Specifically, we instruct the Llama-3.1-8B-Instruct model to evaluate the semantic similarity between the model-generated answer and the reference answer provided in PopQA. The prompt used for the Llama judge is detailed in Table 11.

After judging the correctness of each answer, we use Algorithm 1 to estimate the popularity threshold. In our experiments, we set the target accuracy $\alpha_{\text{target}} = 60\%$ and the fault tolerance level $N_{\text{fail}} = 5$.

Algorithm 1: Estimate Threshold Popularity

Input:
- x: A list of popularity values for each data point, where x_i represents the popularity of the i-th data point.
- y: A list of binary values indicating the correctness of the model's response, where $y_i = 1$ if the model answers the i-the question correctly, and $y_i = 0$ otherwise.
- α_{target}: The target accuracy.
- N_{fail}: The maximum number of failures before termination, denoting the fault tolerance level.

Output:
- P_{thres}: The threshold popularity.

Initialize correct count: sum_correct $\leftarrow 0$
Initialize error count: $e \leftarrow 0$
Sort (x, y) by x in ascending order and store the indices in a list I.
Initialize loop variable $j \leftarrow \text{len}(x) - 1$
Initialize flag counter flag $\leftarrow 0$
while $j \geq 0$ **do**
  $k \leftarrow j$
  while $k \geq 0$ **and** $x_{I_k} = x_{I_j}$ **do**
    $k \leftarrow k - 1$
  end while
  for $l = k + 1$ **to** j **do**
    $i \leftarrow I_l$
    sum_correct $\leftarrow$ sum_correct $+ y_i$
  end for
  if $\frac{\text{sum\_correct}}{\text{len}(x)-k-1} <$ set_threshold **then**
    $e \leftarrow e + 1$
  end if
  if $e = N_{\text{fail}}$ **then**
    Return: x_{I_j} {Return the threshold popularity}
  end if
  $j \leftarrow k$
end while
Return: x_{I_0} {If no such point is found, return the smallest popularity value}

Table 11: The prompt for testing synonym.

```
<|begin\_of\_text|><|start_header_id|>system<|
↪ end_header_id|>
Cutting Knowledge Date: December 2023
Today Date: 19 Dec 2024

You are a linguistic expert specializing in synonyms.
↪ Your task is to determine whether two given English
↪ words are synonyms or not. A synonym is a word that
↪ has a very similar meaning to another word and can
↪ often replace it in sentences without significantly
↪ changing the meaning.

For each pair of words provided:
1. Analyze their meanings and typical usage.
2. Decide whether they are synonyms (Yes/No).
3. Provide a brief explanation for your decision.

Here are some examples to guide you:

Words: "happy" and "joyful"
Yes
Explanation: Both words describe a state of being
↪ pleased or content and are often interchangeable in
↪ most contexts.

Words: "run" and "jog"
No
Explanation: While both refer to forms of movement, "run"
↪  typically implies a faster pace than "jog."

Words: "angry" and "frustrated"
No
Explanation: Although both express negative emotions, "
↪ angry" implies strong displeasure or rage, while "
↪ frustrated" conveys annoyance due to obstacles or
↪ failure.

<|eot_id|><|start_header_id|>user<|end_header_id|>

Words: {} and {}
<|eot_id|><|start_header_id|>assistant<|end_header_id|>
```

H.5 Experimental Details of the Mitigation Strategy

This subsection presents the details for the experiments in Appendix E.

Experimental details of random sampling. In Figure 6(a), we train all models from scratch on the mixture of FineWeb-Edu and SynBio-1.28M using the cosine learning rate schedule with a peak value of 10^{-3}. In Figures 6(b) and 6(c), following Zhu et al. (2024), we continually pre-train intermediate checkpoints of Pythia models. This strategy allows us to use a larger learning rate without experiencing extreme loss spikes. Specifically, we continually pre-train 410M and 1B Pythia models from their respective 100k-step checkpoint with a constant learning rate of 8.7×10^{-5}, which corresponds to the original learning rate used at step 100k in the Pythia model training.

I PROOFS OF THEORETICAL RESULTS

We follow the notations in Section 3. We use $H(\cdot)$ to denote the entropy and $I(\cdot\,;\cdot)$ to denote the mutual information.

We define a data distribution $\mathcal{D}$ as a distribution over (x, y), where x is an input and y is a token. A data universe $\mathcal{U} = (\mathcal{P}, \mathcal{D}_\theta)$ is defined by a prior $\mathcal{P}$ over a latent variable θ and a family of data distributions $\mathcal{D}_\theta$ indexed by θ.

A predictor h is a function that maps x to a distribution over y. A learning algorithm $\mathcal{A}$ is a procedure that takes samples from a data distribution $\mathcal{D}$ of (x, y) and outputs a predictor $h \sim \mathcal{A}(\mathcal{D})$ in the end. For a given predictor h, we measure its performance by the expected cross-entropy loss

$$\mathcal{L}(h; \mathcal{D}) := \mathbb{E}_{(x,y)\sim\mathcal{D}}[-\log p(y \mid h, x)], \tag{5}$$

where $p(y \mid h, x)$ denotes the predicted distribution of y given x by the predictor h, and $\log$ is in base 2 for convenience. For a data universe $\mathcal{U} = (\mathcal{P}, \mathcal{D}_\theta)$, we measure the performance of a learning algorithm $\mathcal{A}$ by its expected loss over all data distributions $\mathcal{D}_\theta$ with respect to the prior $\mathcal{P}$:

$$\bar{\mathcal{L}}_{\mathcal{P}}(\mathcal{A}) := \mathbb{E}_{\theta\sim\mathcal{P}}\mathbb{E}_{h\sim\mathcal{A}(\mathcal{D}_\theta)}[\mathcal{L}(h; \mathcal{D}_\theta)]. \tag{6}$$

We use the mutual information $I(\mathcal{A}(\mathcal{D}_\theta); \mathcal{D}_\theta)$ as a measure of the *effective model capacity* for the predictor picked by $\mathcal{A}$ on $\mathcal{D}_\theta$, where θ is sampled from $\mathcal{Q}$.

Same as Definition 3.1, for a data universe $\mathcal{U} = (\mathcal{P}, \mathcal{D}_\theta)$ and $M > 0$, we define the best achievable loss under the capacity constraint M as

$$F_{\mathcal{P}}(M) := \inf_{\mathcal{A}} \left\{ \bar{\mathcal{L}}_{\mathcal{P}}(\mathcal{A}) : I(\mathcal{A}(\mathcal{D}_\theta); \mathcal{D}_\theta) \le M \right\}, \tag{7}$$

where the infimum is taken over all learning algorithms. An optimal M-bounded-capacity learner is a learning algorithm $\mathcal{A}$ such that $I(\mathcal{A}(\mathcal{D}_\theta); \mathcal{D}_\theta) \le M$ and $\bar{\mathcal{L}}_{\mathcal{P}}(\mathcal{A}) = F_{\mathcal{P}}(M)$.

I.1 CONVEXITY OF THE BEST ACHIEVABLE LOSS

It is easy to see that $F_{\mathcal{P}}(M)$ is non-negative and non-increasing in M. A classic result in rate-distortion theory is that the rate-distortion function is convex. This further implies that $F_{\mathcal{P}}(M)$ is convex in M. Here we present it as a lemma for completeness.

Lemma I.1. *For any data universe $\mathcal{U} = (\mathcal{P}, \mathcal{D}_\theta)$, $F_{\mathcal{P}}(M)$ is convex in M.*

Proof. Let $\epsilon > 0$ be any positive number. Let $\mathcal{A}_1$ be a learning algorithm that achieves a loss $\le F_{\mathcal{P}}(M_1) + \epsilon$ with mutual information $I_1(\mathcal{A}(\mathcal{D}_\theta); \mathcal{D}_\theta) \le M_1$ and $\mathcal{A}_2$ be a learning algorithm that achieves a loss $F_{\mathcal{P}}(M_2) + \epsilon$ with mutual information $I_2(\mathcal{A}(\mathcal{D}_\theta); \mathcal{D}_\theta) \le M_2$.

Let $\mathcal{A}$ be a new learning algorithm that outputs the same as $\mathcal{A}_1$ with probability $1 - p$ and the same as $\mathcal{A}_2$ with probability p. Then the mutual information between $\mathcal{A}(\mathcal{D}_\theta)$ and $\mathcal{D}_\theta$ is

$$\begin{aligned} I(\mathcal{A}(\mathcal{D}_\theta); \mathcal{D}_\theta) &= (1-p)I(\mathcal{A}_1(\mathcal{D}_\theta); \mathcal{D}_\theta) + pI(\mathcal{A}_2(\mathcal{D}_\theta); \mathcal{D}_\theta) \\ &\le (1-p)M_1 + pM_2. \end{aligned}$$

By linearity of expectation, the expected loss of $\mathcal{A}$ can be bounded as

$$\begin{aligned} \mathbb{E}_{\theta\sim\mathcal{P}(\theta)}[\mathcal{L}(\mathcal{A}(\mathcal{D}_{\boldsymbol{\theta}}); \mathcal{D}_{\boldsymbol{\theta}})] &= (1-p)\mathbb{E}_{\theta\sim\mathcal{P}(\theta)}[\mathcal{L}(\mathcal{A}_1(\mathcal{D}_{\boldsymbol{\theta}}); \mathcal{D}_{\boldsymbol{\theta}})] + p\mathbb{E}_{\theta\sim\mathcal{P}(\theta)}[\mathcal{L}(\mathcal{A}_2(\mathcal{D}_{\boldsymbol{\theta}}); \mathcal{D}_{\boldsymbol{\theta}})] \\ &\le (1-p)F_{\mathcal{P}}(M_1) + pF_{\mathcal{P}}(M_2) + 2\epsilon. \end{aligned}$$

Therefore, we have

$$F_{\mathcal{P}}((1-p)M_1 + pM_2) \le \mathbb{E}_{\theta\sim\mathcal{P}(\theta)}[\mathcal{L}(\mathcal{A}(\mathcal{D}_{\boldsymbol{\theta}}); \mathcal{D}_{\boldsymbol{\theta}})] \le (1-p)F_{\mathcal{P}}(M_1) + pF_{\mathcal{P}}(M_2) + 2\epsilon,$$

taking $\epsilon \to 0$ finishes the proof. □

I.2 PROOFS FOR THE WARMUP CASE

Definition I.2 (Factual Data Universe)**.** We define a fact as a pair (X, y), where X is a set of inputs and y is a target token. A factual data universe is a data universe $\mathcal{U} = (\mathcal{P}, \mathcal{D}_\theta)$ containing K random facts $(X_1, y_1), \ldots, (X_K, y_K)$ in the following way:

1. $X_1, \ldots, X_K$ are K disjoint sets of inputs, and $y_1, \ldots, y_K$ are random tokens;
2. θ is structured as $(y_1, \ldots, y_K)$. Given $\theta = (y_1, \ldots, y_K)$, the data distribution $\mathcal{D}_\theta$ satisfies that for all $x \in X_i$, $\mathcal{D}_\theta(y \mid x_i)$ is a point mass at y_i;

3. For all θ, the input distribution $\mathcal{D}_\theta(x)$ is the same;
4. For all θ, the target distribution $\mathcal{D}_\theta(y \mid x)$ is the same for all $x \notin \bigcup_{i=1}^K X_i$;
5. The prior distribution $\mathcal{P}$ over θ is given by the product distribution $\mathcal{P}(y_1, y_2, \ldots, y_K) = \prod_{k=1}^K \mathcal{Y}_k(y_k)$, where $\mathcal{Y}_k$ is a fixed prior distribution over y_k.

The exposure frequency of each random fact is defined as the total probability that an input $x \in X_i$ occurs in $\mathcal{D}_\theta$.

Theorem I.3 (Theorem C.1, restated)**.** *For a factual data universe $\mathcal{U} = (\mathcal{P}, \mathcal{D}_\theta)$ with K random facts, if all the facts have the same exposure frequency p, then*

$$F_{\mathcal{P}}(M) = C + p \cdot \max\{H_{\text{tot}} - M, 0\}, \tag{8}$$

where $H_{\text{tot}} := \sum_{i=1}^K H(\mathcal{Y}_i)$ and $C := F_{\mathcal{P}}(\infty)$.

Proof. First, we prove a lower bound for $F_{\mathcal{P}}(M)$. For any learning algorithm $\mathcal{A}$ with $I(\mathcal{A}(\mathcal{D}_\theta); \mathcal{D}_\theta) \le M$,

$$\begin{aligned}
\bar{\mathcal{L}}_{\mathcal{P}}(\mathcal{A}(\mathcal{D}_\theta)) &= \mathbb{E}_{\theta\sim\mathcal{P}}\mathbb{E}_{(x,y)\sim\mathcal{D}_\theta}\mathbb{E}_{h\sim\mathcal{A}(\mathcal{D}_\theta)}[-\log p(y \mid h, x)] \\
&= \mathbb{E}_x\mathbb{E}_{\theta\sim\mathcal{P}}\mathbb{E}_{y\sim\mathcal{D}_\theta(\cdot|x)}\mathbb{E}_{h\sim\mathcal{A}(\mathcal{D}_\theta)}\mathbb{E}[-\log p(y \mid h, x)] \\
&\ge \underbrace{\mathbb{E}_x\left[\mathbb{1}_{\{x\in\bigcup_{i=1}^K X_i\}} H_{\theta\sim\mathcal{P}}(\mathcal{D}_\theta(\cdot \mid x))\right]}_{=:C_0} + p\left[\sum_{i=1}^K \left(H(\mathcal{Y}_i) - I(\mathcal{A}(\mathcal{D}_\theta); y_i)\right)\right]_+ \\
&\ge C_0 + p\left[H_{\text{tot}} - I(\mathcal{A}(\mathcal{D}_\theta); \mathcal{D}_\theta)\right]_+ \\
&\ge C_0 + p\left[H_{\text{tot}} - M\right]_+.
\end{aligned}$$

For upper bounds, we first show that $F_{\mathcal{P}}(M) \le C_0$ for all $M \ge H_{\text{tot}}$. Let $\mathcal{A}_1$ be the learning algorithm that inputs $\mathcal{D}_\theta$ and outputs the predictor h that always outputs the token y_i for the input $x \in X_i$. For all the other inputs x, the predictor just outputs $h(y \mid h, x) = \mathbb{E}_{\theta\sim\mathcal{P}}[\mathcal{D}_\theta(y \mid x)]$. Both $\mathcal{A}_1(\mathcal{D}_\theta)$ and $\mathcal{D}_\theta$ can be transformed from θ with a reversible function, so

$$I(\mathcal{A}_1(\mathcal{D}_\theta); \mathcal{D}_\theta) = H(\theta) = \sum_{i=1}^K H(\mathcal{Y}_i) = H_{\text{tot}}.$$

It is easy to see that $\bar{\mathcal{L}}_{\mathcal{P}}(\mathcal{A}_1) = C_0$. This implies that $F_{\mathcal{P}}(M) \le C_0$ for all $M \ge H_{\text{tot}}$.

Now, if $M < H_{\text{tot}}$, we construct a learning algorithm $\mathcal{A}_q$ that outputs the same as $\mathcal{A}_1$ with probability q and outputs $h(y \mid h, x) = \mathbb{E}_{\theta\sim\mathcal{P}}[\mathcal{D}_\theta(y \mid x)]$ with probability $1 - q$. Setting $q = \frac{M}{H_{\text{tot}}}$, we have

$$I(\mathcal{A}_q(\mathcal{D}_\theta); \mathcal{D}_\theta) = q \cdot H_{\text{tot}} = M.$$

By linearity of expectation, we also have $\bar{\mathcal{L}}_{\mathcal{P}}(\mathcal{A}_q(\mathcal{D}_\theta)) = \bar{\mathcal{L}}_{\mathcal{P}}(\mathcal{A}_1) + (1 - q) \cdot p \sum_{i=1}^K H(\mathcal{Y}_i)$. This implies that $F_{\mathcal{P}}(M) \le \bar{\mathcal{L}}_{\mathcal{P}}(\mathcal{A}_1) + p \cdot \max\{H_{\text{tot}} - M, 0\}$ for all $M < H_{\text{tot}}$.

Putting all the pieces together finishes the proof. □

I.3 Proofs for the Data Mixing Case

Definition I.4 (Mixture of Data Universes)**.** Let $\mathcal{U}_1 = (\mathcal{P}_1, \mathcal{D}^{(1)}_{\theta_1})$ and $\mathcal{U}_2 = (\mathcal{P}_2, \mathcal{D}^{(2)}_{\theta_2})$ be two data universes. We mix them together to form a new data universe $\mathcal{U} = (\mathcal{P}, \mathcal{D}_\theta)$:

1. θ is structured as (θ_1, θ_2). Given $\theta = (\theta_1, \theta_2)$, the data distribution $\mathcal{D}_\theta$ is formed as $\mathcal{D}_\theta = r\mathcal{D}^{(1)}_{\theta_1} + (1 - r)\mathcal{D}^{(2)}_{\theta_2}$, where r is called *the mixing ratio*;
2. The prior distribution $\mathcal{P}$ over θ is a joint distribution of $\mathcal{P}_1$ and $\mathcal{P}_2$.

In reality, mixing two datasets can be seen as mixing two data universes first and then sampling a data distribution from the mixed data universe. Here we consider the simplified case where the two data universes are so different from each other that they convey orthogonal information.

Definition I.5 (Orthogonal Mixture of Data Universes)**.** We say that $\mathcal{U}$ is an orthogonal mixture of $\mathcal{U}_1$ and $\mathcal{U}_2$ if

1. For any x that is in both supports of $\mathcal{D}_{\theta_1}^{(1)}$ and $\mathcal{D}_{\theta_2}^{(2)}$, we have $\mathcal{D}_{\theta_1}^{(1)}(y \mid x) = \mathcal{D}_{\theta_2}^{(2)}(y \mid x)$ for all θ_1 and θ_2. In other words, the conditional distribution of the next token y given the context x remains consistent across both domains and is unaffected by variations in values of θ_1 and θ_2.
2. $\mathcal{P}(\theta_1, \theta_2) = \mathcal{P}_1(\theta_1) \cdot \mathcal{P}_2(\theta_2)$, i.e., θ_1 and θ_2 are independent.

Below, we first establish two lemmas that provide conditions for when the loss on the first domain will be very low or very high for an optimal M-bounded-capacity learner given an orthogonal mixture of two data universes. Then, we use these lemmas to prove Theorem 3.2.

We use $\mathrm{D}^{-}F(t)$ and $\mathrm{D}^{+}F(t)$ to denote the left and right derivatives of a function F at a point t, respectively.

Lemma I.6. *Let $\mathcal{U} = (\mathcal{P}, \mathcal{D}_\theta)$ be an orthogonal mixture of $\mathcal{U}_1 = (\mathcal{P}_1, \mathcal{D}_{\theta_1}^{(1)})$ and $\mathcal{U}_2 = (\mathcal{P}_2, \mathcal{D}_{\theta_2}^{(2)})$ with mixing ratio r. For all $r \in (0,1)$ and $M \geq 0$, if the following inequality holds,*

$$\frac{r}{1-r} < \frac{\mathrm{D}^{-}F_{\mathcal{P}_2}(M)}{\mathrm{D}^{+}F_{\mathcal{P}_1}(0)}, \tag{9}$$

then for any optimal M-bounded-capacity learner $\mathcal{A}$ on $\mathcal{U}$, $\mathbb{E}_{\theta\sim\mathcal{P}}\mathbb{E}_{h\sim\mathcal{A}(\mathcal{D}_\theta)}[\mathcal{L}(h; \mathcal{D}_{\theta_1}^{(1)})] = F_{\mathcal{P}_1}(0)$.

Intuitive Explanation. We define $\bar{\mathcal{L}}_2(\mathcal{A}) := \mathbb{E}_{\theta\sim\mathcal{P}_2}[\mathcal{L}(\mathcal{A}(\mathcal{D}_\theta); \mathcal{D}_{\theta_1}^{(2)})]$, similar to $\bar{\mathcal{L}}_1(\mathcal{A})$. The overall test loss is given by $\bar{\mathcal{L}}_{\mathcal{P}}(\mathcal{A}) = r\bar{\mathcal{L}}_1(\mathcal{A}) + (1-r)\bar{\mathcal{L}}_2(\mathcal{A})$. Since $\mathrm{D}^{+}F_{\mathcal{P}_1(0)} < 0$, we can rearrange (9) to obtain $r\mathrm{D}^{+}F_{\mathcal{P}_1}(0) - (1-r)\mathrm{D}^{-}F_{\mathcal{P}_2}(M) > 0$. Intuitively, this means that increasing the capacity assigned to learn $\mathcal{U}_1$ by one unit and reducing the capacity for $\mathcal{U}_2$ by one unit will increase the overall test loss, compared to fully assigning capacity to $\mathcal{U}_2$ and none to $\mathcal{U}_1$. Alternatively, we can view $\frac{r\mathrm{D}^{+}F_{\mathcal{P}_1}(0)}{(1-r)\mathrm{D}^{-}F_{\mathcal{P}_2}(M)}$ as the Competitive Ratio (CR) of the knowledge-dense dataset relative to web data. Hence, the model should prioritize web data and not learn from the knowledge-dense dataset when $CR < 1$.

Proof. Let h be the predictor picked by $\mathcal{A}$ on $\mathcal{D}_\theta$. Let $\mathcal{X}_1$ and $\mathcal{X}_2$ be the supports of x in $\mathcal{D}_{\theta_1}^{(1)}$ and $\mathcal{D}_{\theta_2}^{(2)}$, respectively. Let $m_1 := I(h|_{\mathcal{X}_1}; \mathcal{D}_{\theta_1})$ and $m_2 := I(h|_{\mathcal{X}_2}; \mathcal{D}_{\theta_2})$. By data processing inequality, we have

$$m_1 = I(h|_{\mathcal{X}_1}; \mathcal{D}_{\theta_1}) \leq I(h; \mathcal{D}_{\theta_1}),$$
$$m_2 = I(h|_{\mathcal{X}_2}; \mathcal{D}_{\theta_2}) \leq I(h; \mathcal{D}_{\theta_2}),$$

Further noticing that $I(h; \mathcal{D}_\theta) = I(h; \mathcal{D}_{\theta_1}; \mathcal{D}_{\theta_2}) \geq I(h; \mathcal{D}_{\theta_1}) + I(h; \mathcal{D}_{\theta_2})$, we have

$$m_1 + m_2 \leq I(h; \mathcal{D}_\theta) \leq M.$$

Since $h|_{\mathcal{X}_1}$ and $h|_{\mathcal{X}_2}$ are valid predictors on $\mathcal{D}_{\theta_1}$ and $\mathcal{D}_{\theta_2}$, respectively, we have

$$\mathbb{E}[\mathcal{L}(h; \mathcal{D}_{\theta_1})] = \mathbb{E}[\mathcal{L}(h|_{\mathcal{X}_1}; \mathcal{D}_{\theta_1})] \geq F_{\mathcal{P}_1}(m_1),$$
$$\mathbb{E}[\mathcal{L}(h; \mathcal{D}_{\theta_2})] = \mathbb{E}[\mathcal{L}(h|_{\mathcal{X}_2}; \mathcal{D}_{\theta_2})] \geq F_{\mathcal{P}_2}(m_2) \geq F_{\mathcal{P}_2}(M - m_1).$$

Adding the two inequalities with weights r and $1-r$, we have

$$\bar{\mathcal{L}}_{\mathcal{P}}(\mathcal{A}) = \mathbb{E}[\mathcal{L}(h; \mathcal{D}_\theta)] \geq rF_{\mathcal{P}_1}(m_1) + (1-r)F_{\mathcal{P}_2}(M - m_1).$$

By convexity (Lemma I.1), we have

$$F_{\mathcal{P}_1}(m_1) \geq F_{\mathcal{P}_1}(0) + \mathrm{D}^{+}F_{\mathcal{P}_1}(0)m_1, \qquad F_{\mathcal{P}_2}(M - m_1) \geq F_{\mathcal{P}_2}(M) - \mathrm{D}^{-}F_{\mathcal{P}_2}(M)m_1.$$

Plugging these into the previous inequality, we have

$$\mathbb{E}[\mathcal{L}(h; \mathcal{D}_\theta)] \geq rF_{\mathcal{P}_1}(0) + (1-r)F_{\mathcal{P}_2}(M) + \left(r\mathrm{D}^{+}F_{\mathcal{P}_1}(0) - (1-r)\mathrm{D}^{-}F_{\mathcal{P}_2}(M)\right)m_1.$$

By (9) and the fact that $\mathrm{D}^{+}F_{\mathcal{P}_1}(0) \leq 0$, we have $r\mathrm{D}^{+}F'_{\mathcal{P}_1}(0) > (1-r)\mathrm{D}^{-}F'_{\mathcal{P}_2}(M)$. So the right-hand side is strictly increasing in m_1.

Now we claim that $m_1 = 0$. If not, then the following learning algorithm $\mathcal{A}'$ is better than $\mathcal{A}$. Let $\mathcal{A}_1$ be an optimal 0-bounded-capacity learner on $\mathcal{U}_1$ and $\mathcal{A}_2$ be an optimal M-bounded-capacity learner on $\mathcal{U}_2$. Run the algorithms to obtain $h_1 \sim \mathcal{A}_1(\mathcal{D}_\theta|_{\mathcal{X}_1})$ and $h_2 \sim \mathcal{A}_2(\mathcal{D}_\theta|_{\mathcal{X}_2})$. Then, whenever seeing an input x from $\mathcal{X}_1$, output $h_1(x)$; otherwise output $h_2(x)$. This algorithm achieves the expected loss $rF_{\mathcal{P}_1}(0) + (1-r)F_{\mathcal{P}_2}(M)$, which is strictly less than $\bar{\mathcal{L}}_{\mathcal{P}}(\mathcal{A})$ and contradicts the optimality of $\mathcal{A}$.

Therefore, for the optimal algorithm $\mathcal{A}$, $\bar{\mathcal{L}}_1(\mathcal{A}) = F_{\mathcal{P}_1}(0)$. □

Lemma I.7. *Let $\mathcal{U} = (\mathcal{P}, \mathcal{D}_\theta)$ be an orthogonal mixture of $\mathcal{U}_1 = (\mathcal{P}_1, \mathcal{D}_{\theta_1})$ and $\mathcal{U}_2 = (\mathcal{P}_2, \mathcal{D}_{\theta_2})$ with mixing ratio r. For all $r \in (0,1)$, $M \geq 0$ and $\beta \geq 0$, if the following inequality holds,*

$$\frac{r}{1-r} > \frac{\mathrm{D}^+F_{\mathcal{P}_2}(M-\beta)}{\mathrm{D}^-F_{\mathcal{P}_1}(\beta)}, \tag{10}$$

then for any optimal M-bounded-capacity learner $\mathcal{A}$ on $\mathcal{U}$, $\mathbb{E}_{\theta\sim\mathcal{P}}\mathbb{E}_{h\sim\mathcal{A}(\mathcal{D}_\theta)}[\mathcal{L}(h;\mathcal{D}_{\theta_1})] \leq F_{\mathcal{P}_1}(\beta)$.

Intuitive Explanation. Similar to the explanation for Lemma I.6, we can rearrange Equation (10) into $-r\mathrm{D}^-F_{\mathcal{P}_1}(\beta) + (1-r)\mathrm{D}^+F_{\mathcal{P}_1}(M-\beta) < 0$. Intuitively, this means that reducing the capacity assigned to learn $\mathcal{U}_1$ by one unit and increasing the capacity for $\mathcal{U}_2$ by one unit will increase the overall test loss, compared to assigning capacity β to $\mathcal{U}_1$ and $M-\beta$ to $\mathcal{U}_2$. Therefore, the optimal M-bounded-capacity learner $\mathcal{A}$ will assign at least capacity β to learn $\mathcal{U}_1$, resulting in a test loss on $\mathcal{U}_1$ that is lower than $F_{\mathcal{P}_1}(\beta)$. Alternatively, we can view $\frac{r\mathrm{D}^-F_{\mathcal{P}_1}(\beta)}{(1-r)\mathrm{D}^+F_{\mathcal{P}_2}(M-\beta)}$ as the Competitive Ratio (CR) of the knowledge-dense dataset relative to web data. Hence, the model should do its best to learn the knowledge-dense dataset when $CR > 1$.

Proof. Similar to the previous proof, letting $m_1 := I(h|_{\mathcal{X}_1}; \mathcal{D}_{\theta_1})$ and $m_2 := I(h|_{\mathcal{X}_2}; \mathcal{D}_{\theta_2})$, we have

$$\begin{aligned}
m_1 + m_2 &\leq I(h;\mathcal{D}_\theta) \leq M,\\
\mathbb{E}[\mathcal{L}(h;\mathcal{D}_{\theta_1})] &= \mathbb{E}[\mathcal{L}(h|_{\mathcal{X}_1};\mathcal{D}_{\theta_1})] \geq F_{\mathcal{P}_1}(m_1),\\
\mathbb{E}[\mathcal{L}(h;\mathcal{D}_{\theta_2})] &= \mathbb{E}[\mathcal{L}(h|_{\mathcal{X}_2};\mathcal{D}_{\theta_2})] \geq F_{\mathcal{P}_2}(m_2) \geq F_{\mathcal{P}_2}(M-m_1),\\
\mathbb{E}[\mathcal{L}(h;\mathcal{D}_\theta)] &\geq rF_{\mathcal{P}_1}(m_1) + (1-r)F_{\mathcal{P}_2}(M-m_1).
\end{aligned}$$

First, we show that $m_1 \geq \beta$. If not, then by convexity (Lemma I.1), we have

$$\begin{aligned}
F_{\mathcal{P}_1}(m_1) &\geq F_{\mathcal{P}_1}(\beta) - \mathrm{D}^-F_{\mathcal{P}_1}(\beta)\cdot(\beta-m_1),\\
F_{\mathcal{P}_2}(M-m_1) &\geq F_{\mathcal{P}_2}(M-\beta) + \mathrm{D}^+F_{\mathcal{P}_2}(M-\beta)\cdot(\beta-m_1).
\end{aligned}$$

Plugging these into the previous inequality, we have
$\mathbb{E}[\mathcal{L}(h;\mathcal{D}_\theta)] \geq rF_{\mathcal{P}_1}(\beta) + (1-r)F_{\mathcal{P}_2}(M-\beta) + \left(-r\mathrm{D}^-F_{\mathcal{P}_1}(\beta) + (1-r)\mathrm{D}^+F_{\mathcal{P}_2}(M-\beta)\right)(\beta-m_1)$.
By (10) and the fact that $\mathrm{D}^-F_{\mathcal{P}_1}(\beta) \leq 0$, we have $r\mathrm{D}^-F_{\mathcal{P}_1}(\beta) < (1-r)\mathrm{D}^+F_{\mathcal{P}_2}(M-\beta)$. So the right-hand side is strictly decreasing in m_1.

Next, we prove by contradiction that $m_1 \geq \beta$. If $m_1 < \beta$, the following learning algorithm $\mathcal{A}'$ is better than $\mathcal{A}$. Let $\mathcal{A}_1$ be an optimal β-bounded-capacity learner on $\mathcal{U}_1$ and $\mathcal{A}_2$ be an optimal $(M-\beta)$-bounded-capacity learner on $\mathcal{U}_2$. Run the algorithms to obtain $h_1 \sim \mathcal{A}_1(\mathcal{D}_\theta|_{\mathcal{X}_1})$ and $h_2 \sim \mathcal{A}_2(\mathcal{D}_\theta|_{\mathcal{X}_2})$. Then, whenever seeing an input x from $\mathcal{X}_1$, output $h_1(x)$; otherwise output $h_2(x)$. This algorithm achieves the expected loss $rF_{\mathcal{P}_1}(\beta) + (1-r)F_{\mathcal{P}_2}(M-\beta)$, which is strictly less than $\bar{\mathcal{L}}_{\mathcal{P}}(\mathcal{A})$.

Therefore, we have $m_1 \geq \beta$ for the algorithm $\mathcal{A}$. Now we prove that $\mathbb{E}[\mathcal{L}(h;\mathcal{D}_{\theta_1})] \leq F_{\mathcal{P}_1}(\beta)$. If not, then the following learning algorithm $\mathcal{A}''$ is better than $\mathcal{A}$. Construct $\mathcal{A}''$ similarly as $\mathcal{A}'$, but with $\mathcal{A}_1$ and $\mathcal{A}_2$ replaced by the optimal m_1-bounded-capacity learner on $\mathcal{U}_1$ and the optimal m_2-bounded-capacity learner on $\mathcal{U}_2$, respectively. If $\mathbb{E}[\mathcal{L}(h;\mathcal{D}_{\theta_1})] > F_{\mathcal{P}_1}(\beta)$, then $\mathcal{A}''$ achieves a lower expected loss than $\mathcal{A}$, which contradicts the optimality of $\mathcal{A}$. □

Now we consider the case where $\mathcal{U}_1$ is a factual data universe, and $\mathcal{U}_2$ is an arbitrary data universe.

Theorem I.8. *Let $\mathcal{U}_1$ be a factual data universe with K random facts, each with the same exposure frequency p, and the entropies of their target tokens sum to $H_{\mathrm{tot}} := \sum_{i=1}^K H(\mathcal{Y}_i)$. Let $\mathcal{U}_2$ be an arbitrary data universe. Let $\mathcal{U} = (\mathcal{P}, \mathcal{D}_\theta)$ be an orthogonal mixture of $\mathcal{U}_1$ and $\mathcal{U}_2$ with mixing ratio r. For all $r \in (0,1)$ and $M \geq 0$,*

1. *if $\frac{r}{1-r}\cdot p < -\mathrm{D}^-F_{\mathcal{P}_2}(M)$, then $\bar{\mathcal{L}}_1(\mathcal{A}) = F_{\mathcal{P}_1}(0)$;*
2. *if $\frac{r}{1-r}\cdot p > -\mathrm{D}^+F_{\mathcal{P}_2}(M - H_{\mathrm{tot}})$, then $\bar{\mathcal{L}}_1(\mathcal{A}) = F_{\mathcal{P}_1}(\infty)$.*

Proof. By Theorem I.3, $\mathrm{D}^+F_{\mathcal{P}_1}(0) = \mathrm{D}^-F_{\mathcal{P}_1}(H_{\mathrm{tot}}) = p$. Plugging this into Lemma I.6 and Lemma I.7 with $\beta = H_{\mathrm{tot}}$ finishes the proof. □

Now we are ready to prove the main theorem we stated in Section 3. Recall that

$$M_0^-(t) := \sup\{M \geq 0 : -F'_{\mathcal{P}_2}(M) > t\},$$
$$M_0^+(t) := \inf\{M \geq 0 : -F'_{\mathcal{P}_2}(M) < t\},$$

Theorem I.9 (Theorem 3.2, restated)**.** *For any optimal M-bounded-capacity learner $\mathcal{A}$,*

1. *if $M \leq M_0^-(\frac{r}{1-r} \cdot p)$, then $\bar{\mathcal{L}}_1(\mathcal{A}) = F_{\mathcal{P}_1}(0)$;*
2. *if $M \geq M_0^+(\frac{r}{1-r} \cdot p) + H_{\mathrm{tot}}$, then $\bar{\mathcal{L}}_1(\mathcal{A}) = F_{\mathcal{P}_1}(\infty)$.*

Proof. This is a direct consequence of Theorem I.8 by noting that (1) $-\mathrm{D}^- F_{\mathcal{P}_2}(M)$ is left continuous and non-increasing in M; (2) $-\mathrm{D}^+ F_{\mathcal{P}_2}(M)$ is right continuous and non-increasing in M; (3) $F_{\mathcal{P}_2}(M)$ is almost everywhere differentiable. □