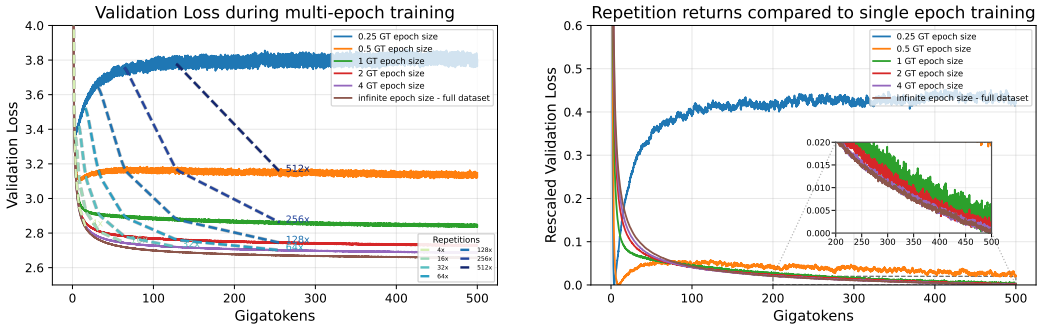
TRAIN SMARTER, NOT LONGER: MEMORIZATION-GUIDED DATA REUSE FOR EFFICIENT LLM TRAINING

Anonymous authors
Paper under double-blind review

ABSTRACT

The training paradigm of large language models has shifted from traditional one-pass training to multi-epoch training, as reasonable reuse of limited high-quality data can improve both model performance and sample efficiency. Meanwhile, excessive repetition introduces the risk of overfitting and diminishing returns. Determining when and how to reuse data effectively thus emerges as a natural but under-explored question. Through a novel observation of model's *Memorization Window* signals derived from loss retention dynamics and downstream evaluation scores, we propose *Memorization-guided Data Reuse*, a training paradigm that adaptively determines *when* and *how* data should be reused, enabling principled decisions on the number of training epochs and the scheduling of data replays. Our preliminary experiments reveal a consistent memorization-driven regime: performance continues to improve with repetition far beyond current practice (e.g., the commonly cited four-epoch limit). While a full scheduler remains future work, these insights provide a foundation for memorization-aware training schedules, helping to determine reuse budgets and move toward training LLMs *smarter rather than longer* with limited high-quality data.

1 INTRODUCTION



(a) Epoch size effect on validation loss

(b) Marginal Returns vs. Fresh Data

Figure 1: Epoch size determines the value of repetition. Under *memorization-aware* data scheduler (100M Transformer): (a) Validation loss for varying epoch sizes N. Small N leads to rapid overfitting, while larger N tracks the infinite-data baseline. (b) Returns from repetition vs. fresh data. When spacing is sufficient (e.g., $N = 4.0$GT), the loss decreases at the same rate as the full dataset (brown), implying that re-learning forgotten data yields the same marginal gain as injecting new data

Despite most LLMs were trained with a single epoch prior to 2024, recent advances in large language model (LLM) training rely heavily on multi-epoch training, in which training data are reused across epochs to improve model performance. This practice is driven by the limited availability of high-quality (HQ) data as LLMs continue to scale. Many works (Taylor et al., 2022; Lozhkov et al.; Olmo et al., 2025; Hao et al., 2025) have explored repetition strategies, typically including the reuse of high-quality (HQ) data uniformly across epochs.

Since prior work has explored the effect of data repetition on training (Muennighoff et al., 2023), showing that *training with up to four epochs of repeated data yields negligible changes to loss while further repetition leads to diminishing returns*, most existing repetition strategies adopt a relatively conservative approach. However, we argue that they either rely on fixed heuristics for data reuse (e.g., globally shuffling multi-epoch data) or ignore the dynamics of model memorization. We illustrate this dynamic in Figure 1a by varying different epoch size, the onset of overfitting is not fixed but shifts according to the epoch size N, effectively delaying the degradation point.

This leads to open questions about *given limited HQ data, how to optimally schedule or reuse them during training*. In practice, training with limited HQ data presents a trade-off: increasing the number of epochs on the HQ data risks overfitting through excessive repetition, while increasing HQ proportion with fixed epochs constrains the total training token budget. Addressing this trade-off requires a principled understanding of how memorization develops during training and how it interacts with data reuse. Regarding the reuse of limited HQ data, Figure 1b reveals a striking finding: under *memorization-aware* data scheduling, data repetition becomes mathematically equivalent to training on fresh data in terms of learning efficiency. When the spacing between repetitions is sufficient (e.g., $N = 4.0\,\text{GT}$[1]), the validation loss decreases at the same rate as a model trained on fresh, unique data. This implies that while the absolute performance remains bounded by the limited diversity of the finite subset (a diversity gap), the marginal utility of a repeated token is fully restored once the model has *forgotten* it, i.e., the repetition provides learning signal as if the sample were new.

In this work, we provide preliminary empirical evidence of a *Memorization Window* in LLM training and its interaction with training epochs. We observe that repeated exposure to high-quality (HQ) data initially improves performance but eventually leads to overfitting. By quantifying memorization through loss retention and downstream evaluation scores, we show that these signals can inform the design of adaptive training schedules, enabling safe repetition of HQ data beyond the typical four-epoch limit without incurring overfitting. Our analysis offers actionable insights for *memorization-aware data reuse*, laying the foundation for training LLMs *smarter rather than longer* with limited high-quality data. The main contributions of this paper are three-fold:

- We hypothesize the existence of a *Memorization Window* in large language models and validate it through both explicit and implicit measurements across multiple metrics.
- We demonstrate the correlation between the *Memorization Window* and the number of unique training tokens, and present preliminary results showing its impact on model performance and data scheduling strategies.
- We initiate a broader discussion on model memorization and its interaction with other factors influencing training efficiency, highlighting new research opportunities toward smarter and more efficient LLM training.

2 Related Work

2.1 Multi-epoch training with data repetition

From single-epoch training to inevitable reuse. Under the guidance of the Chinchilla scaling law (Hoffmann et al., 2022), early LLMs were predominantly trained for a single epoch over the pretraining corpus. Studies from this period largely cautioned against data reuse, reporting that repeated tokens could induce memorization and harm generalization (Hernandez et al., 2022; Lee et al., 2022). These findings motivated aggressive deduplication pipelines and the view that unique tokens should be prioritized over repeated exposure. However, the landscape has shifted as models continue to scale while the supply of high-quality human-generated data becomes increasingly limited. The central question has therefore evolved from *whether* to reuse data to *how* it should be reused effectively.

Empirical studies on repetition limits. Several works have begun to examine multi-epoch training more systematically. The scaling analysis of (Taylor et al., 2022; Muennighoff et al., 2023) shows that naively repeating data yields diminishing returns beyond roughly four epochs, establishing a widely cited practical limit. Similar observations are reported in (Xue et al., 2023), which

[1] Gigatokens (GT) correspond to billion tokens (B); the two notations are used interchangeably in the paper

highlights the interaction between unique token count, dataset quality, and degradation under token scarcity. (Parmar et al., 2024) further demonstrates diminishing marginal gains from continued pretraining and explores selecting reusable subsets rather than repeating the full corpus. Complementary theory from (Yan et al., 2025) suggests that the amount of unique data fundamentally governs how many repetitions remain beneficial and larger datasets can be repeated more. Recently, Phi-4 (Abdin et al., 2024) reports gains when training up to 12 epochs on synthetic data, contrasting with the four-epoch heuristic.

Data curation and controlled duplication. Parallel to these studies, data-centric efforts indicate that repetition can be beneficial when applied selectively. RefinedWeb (Penedo et al., 2023) and FineWeb (Penedo et al., 2024) show that controlled duplication of curated web sub-sources can improve performance, and OLMo-3 (Olmo et al., 2025) explicitly caps domain repetition to 4–7 passes. These results suggest that the restriction to single-epoch training is primarily a property of massive web corpora, and may not apply to carefully curated or high-quality subsets—precisely the regime where data scarcity is most acute.

2.2 Memorization in Large Language Models

Beyond a certain number of repetitions, models tend to memorize training data and overfit, degrading generalization. SmolLM2 (Allal et al., 2025) reports an early plateau when mathematics data is repeated for around ten epochs, illustrating the tension between capability gains and memorization. Memorization itself has been characterized in multiple ways. (Li et al., 2025) identify distinct phases of memorization, generalization, and grokking through validation loss dynamics, while (Zucchet et al., 2025) shows that repetition can accelerate the emergence of capabilities in grokking-like regimes. From a distributional perspective, (Wang et al., 2024) defines *distributional memorization* as the correlation between model outputs and the pretraining data distribution. An information-theory view is provided by (Morris et al., 2025), which distinguishes intended and unintended memorization and offers statistical tools to quantify them. Together, these works highlight that memorization is neither purely harmful nor fully understood, and that existing multi-epoch strategies lack mechanisms to monitor or exploit memorization signals—motivating our investigation of memorization-aware data reuse.

3 Memorization and the Memorization Window

We formally define *memorization* and the *memorization window* to characterize when data reuse remains beneficial during LLM pretraining. Our key insight is that the interval between repeated exposures to the same sample—measured in training tokens—determines whether reuse aids learning or causes overfitting.

3.1 Measuring Memorization

Consider a training corpus $D_{\text{train}} = (x_1, x_2, \ldots, x_n)$ processed sequentially. When the model first encounters sample x_i at token position t_i, it incurs a certain loss. After continued training on subsequent data, the loss on x_i reveals whether the model retained or forgot this sample.

Definition 3.1 (Loss Retention Gap)**.** *Let $\ell_i(\theta_{t_i})$ denote the loss on sample x_i when first encountered at token position t_i, and let $\ell_i(\theta_{t_i+\tau})$ denote the loss on x_i after τ additional tokens of continued training (**excluding** x_i). The **loss retention gap** is:*

$$\Delta_i(\tau) = \ell_i(\theta_{t_i}) - \ell_i(\theta_{t_i+\tau}), \tag{1}$$

where $\ell_i(\theta)$ is the per-sample negative log-likelihood:

$$\ell_i(\theta) = -\frac{1}{|x_i|} \sum_{j=1}^{|x_i|} \log p_\theta(x_{i,j} \mid x_{i,<j}). \tag{2}$$

A large $\Delta_i(\tau)$ indicates the model retains information about x_i after τ tokens of continual training; a small gap indicates forgetting.

Definition 3.2 (Sample-level Memorization). *Sample x_i is **memorized** at horizon $\tau_{\max}$ if the loss retention gap remains large:*

$$x_i \text{ is memorized} \iff \Delta_i(\tau) \geq \epsilon, \quad \forall\, \tau \in (0, \tau_{\max}], \tag{3}$$

where $\epsilon > 0$ is a threshold. Equivalently, x_i is memorized if its loss remains significantly lower than when first encountered, indicating the model still retains the learned information.

3.2 The Memorization Window

The memorization window characterizes the token interval over which data reuse remains beneficial. We define it from two complementary perspectives—retention and generalization—that correspond to our experimental measurements.

Definition 3.3 (Memorization Window). *We define the memorization window from two complementary perspectives:*

***(i) Retention-based.** The **retention window** τ_{ret} is the maximum token interval over which a sample remains memorized:*

$$\tau_{\text{ret}} = \sup\{\tau : \Delta(\tau) \geq \epsilon\}, \tag{4}$$

where $\Delta(\tau)$ is the loss retention gap (Definition 3.1) after τ tokens of continued training. For $\tau > \tau_{\text{ret}}$, the gap falls below ϵ, indicating forgetting.

***(ii) Generalization-based.** The **generalization window** τ_{gen} is the training horizon at which downstream performance peaks:*

$$\tau_{\text{gen}} = \arg\max_{\tau \in (0,T]} \mathcal{G}(\theta_\tau), \tag{5}$$

where T is the total training budget and $\mathcal{G}(\theta_\tau)$ denotes generalization performance after τ tokens of training. For $\tau > \tau_{\text{gen}}$, continued training leads to overfitting.

***(iii) Effective memorization window.** The **memorization window** τ^* is the safe reuse interval:*

$$\tau^* = \min(\tau_{\text{ret}}, \tau_{\text{gen}}). \tag{6}$$

Intuitively, τ_{ret} marks when the model *forgets*, while τ_{gen} marks when *overfitting* begins. The effective window τ^* is constrained by whichever limit is reached first.

3.3 Implications for Multi-Epoch Training

We consider two approaches to multi-epoch data scheduling:

Global shuffling (Traditional). The standard approach shuffles all data globally across epochs. Given a corpus D_{train} repeated for E epochs, all $E \times |D_{\text{train}||}|$ samples are randomly permuted before training. This treats each sample independently, ignoring when it was last seen. The interval between consecutive exposures to the same sample x_i varies randomly from 1 to $E \times |D_{\text{train}}|$ tokens, providing no control over memorization dynamics.

Spaced repetition (Memorization-aware). We propose scheduling data reuse based on the memorization window. Consider a training corpus $D_{\text{train}} = (x_1, x_2, \ldots, x_n)$ comprising N tokens, processed sequentially and repeated over multiple epochs in the same order. Each sample x_i is re-encountered after exactly N tokens—a fixed, controllable interval. The relationship between epoch length N and memorization window τ^* determines the effectiveness of spaced repetition:

Case 1: $N < \tau^*$ (Over-memorization risk). Samples are re-exposed *before* being forgotten. Since the model still retains information about x_i, the repeated exposure provides redundant learning signal. Continued reuse in this regime risks over-memorization and eventual overfitting.

Case 2: $N \approx \tau^*$ (Optimal reuse). Samples are re-exposed *just as* they begin to be forgotten. This maximizes the utility of each repetition: the model refreshes decaying information precisely when needed, balancing retention and learning efficiency.

Case 3: $N > \tau^*$ (Forgetting before reuse). Samples are partially forgotten before re-exposure, requiring the model to re-learn information about x_i. While this regime avoids over-memorization, its impact on final performance relative to $N \approx \tau^*$ remains an empirical question.

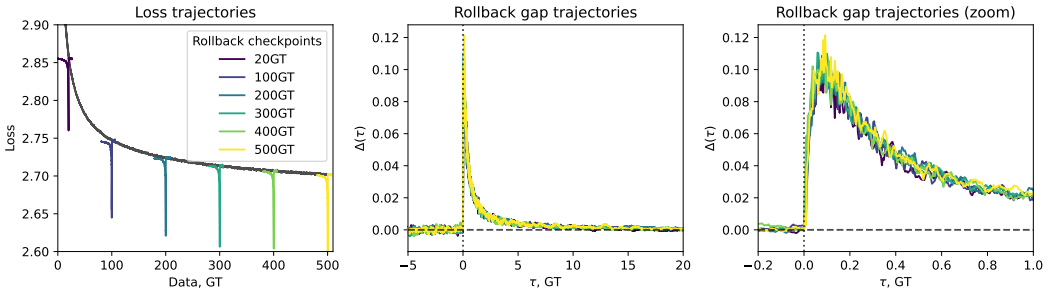


Figure 2: *(Left)* The loss trajectories for the original training without repetitions and for the rollbacks at the specified checkpoints from 20 to 500 Billion tokens. *(Middle and right)* Rollback gap $\Delta^R(\tau)$ vs. data age τ (in Billion tokens).

Epoch budget constraint. Even when $N \geq \tau^*$, excessive repetition eventually causes overfitting due to cumulative over-exposure. We define the *safe training regime* as:

$$N \geq \tau^* \quad \text{and} \quad E \leq E_{\max}, \tag{7}$$

where E is the number of epochs and $E_{\max}$ is the maximum epochs before generalization degrades:

$$E_{\max} = \arg \max_{E \in \{1,\ldots,E_{\text{budget}}\}} \mathcal{G}(\theta^{(E)}), \tag{8}$$

with $\theta^{(E)}$ denoting the checkpoint after E epochs, $E_{\text{budget}||}$ the maximum epochs considered and $\mathcal{G}(\cdot)$ the downstream generalization performance.

Summary. Our framework yields a principled criterion for multi-epoch training: *the epoch length N should satisfy $N \geq \tau^*$ to avoid over-memorization, and the total number of epochs E should not exceed $E_{\max}$*. Together, these constraints define the safe reuse budget for high-quality data. We empirically validate these predictions in Section 4.

Remark. Our definition of memorization is *functional* rather than *extractive*. Extractive memorization (Carlini et al., 2022) concerns whether a model can verbatim reproduce training data—a privacy concern. Functional memorization concerns whether a model retains learned information, measured by loss stability under continued training. This aligns with our goal of informing data scheduling, where the key question is whether re-exposure provides learning signal.

4 Experiments

We validate the memorization window framework through two experiments: (i) measuring the retention window τ_{ret} via rollback loss, and (ii) measuring the generalization window τ_{gen} via downstream evaluation. We aim to answer: Do τ_{ret} and τ_{gen} exist? How does epoch size N affect performance? What role does LQ data play beyond spacing?

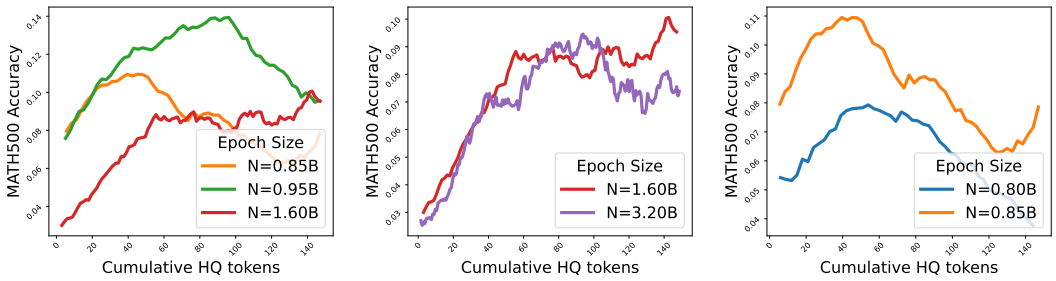
4.1 Retention Window

We measure the retention window τ_{ret} by evaluating how quickly the model forgets previously seen data.

Setup. Definition 3.1 measures forgetting by comparing current loss to the loss when data was originally trained. In practice, tracking per-sample losses throughout training is computationally prohibitive. We adopt a simpler *rollback* protocol to estimate original memorization profile $\Delta(\tau)$. Specifically, we fix model checkpoint θ_t, and compute the loss of this checkpoint on the past and future tokens relative to checkpoint step t. Formally, we compute:

$$\Delta^R(\tau) = \ell_{\text{base}}(\theta_t) - \ell(\theta_t; x_{t-\tau}), \tag{9}$$

where $\ell_{\text{base}}(\theta_t)$ is the estimation of the population loss of checkpoint θ_t via averaging the loss on a large enough collection of unseen token. By comparing equation 9 with definition equation 3.1 we see that $\Delta^R(\tau)$ reasonably approximates $\Delta(\tau)$ up to slight differences.



(a) Epoch size effect (b) Saturation regime (c) LQ diversity effect

Figure 3: MATH500 accuracy vs. cumulative HQ tokens. (a) Larger epoch size delays the peak: orange ($N = 0.85$B) peaks earliest, green ($N = 0.95$B) later, red ($N = 1.6$B) latest among the three. (b) Red ($N = 1.6$B) and purple ($N = 3.2$B) show similar behavior, both peaking at 100–150B HQ tokens—indicating saturation beyond τ^*. (c) Minimal LQ addition (6%, orange) dramatically outperforms pure HQ (blue), demonstrating diversity matters beyond spacing.

- We can think of the retention gap $\Delta(\tau)$ as consisting of two components: the memorization of sample x_i after training for τ tokens; and overall improvement of model capability from θ_{t_i} to $\theta_{t_i+\tau}$ that uniformly decreases the loss on all samples, including x_i. Similarly, the rollback gap $\Delta^R(\tau)$ consist of the same memorization and model evolution components, but the evolution component is different as τ is varied. Instead of the change in model capabilities when the loss is measured, rollback gap is potentially affected by a change of capabilities when the sample $x_{t-\tau}$ was memorized by the model in the state $\theta_{t-\tau}$.
- Rollback gap is a noisy estimation of underlying population characteristic due to variance of per-sample loss values $\ell(\theta; x)$ with respect to x. We have already mentioned that $\ell_{\text{base}}(\theta_t)$ is a noisy estimation of population loss. Likewise, $\ell(\theta_t; x_{t-\tau})$ can be treated as a stochastic process with respect to τ, and we want to access the mean of this process. This can be done via smoothing, but choosing the smoothing window presents the classical bias-variance tradeoff.

Results. We train a 100M model on Fineweb (Penedo et al., 2024) without repetitions, and then perform rollbacks at different positions during training. The training is done over 500GT with constant learning rate and the other optimization hyperparameters. The results are depicted on figure 2. We make two observations:

- The rollback gap $\Delta^R(\tau)$ peaks at around $\tau = 0.15$GT, and then quickly decays to 0. While we don't fix the threshold ϵ at this moment, we see that the gap decays substantially between 1 and 5 billion tokens, putting the retention window in the range $\tau_{\text{ret}} \in (1\text{GT}, 5\text{GT})$.
- Perhaps counterintuitively, the rollbacks trajectories fully overlap even though the respective checkpoints span large range of training stage from 20 to 500 GT. From this observation, we cautiously conclude that the degree of memorization of individual tokens in independent from model capabilities if model architecture and optimizer hyperparameter are fixed.

4.2 Generalization Window

Setup. We train a 100M parameter decoder-only Transformer from scratch. We use OpenMathInstruct2 (Toshniwal et al., 2024) as high-quality (HQ) data (samping `0.8B unique tokens`) and FineWeb (Penedo et al., 2024) as low-quality (LQ) data. We vary epoch size $N \in \{0.8, 0.85, 0.95, 1.6, 3.2\}$B tokens by mixing HQ with different amounts of LQ data while keeping unique HQ tokens fixed. We evaluate on MATH500 (Lightman et al., 2023) and report accuracy vs. cumulative HQ tokens.

We present results in three parts: (i) the effect of epoch size on peak location, (ii) saturation beyond the generalization window, and (iii) the role of LQ data diversity.

Table 1: Memorization window estimates from experiments.

Parameter	Estimate	Evidence
$\tau_{\text{ret}\|\|}$	1B - 5B tokens	Retention loss (Fig. 2)
Optimal N^*	≈ 0.95B tokens	Highest peak (Fig. 3a)
Saturation threshold	≥ 1.6B tokens	Red $\approx$ Purple (Fig. 3b)
τ_{gen}	100–150B tokens HQ tokens	Peak location
$E_{\max}$	125–190 epochs	$\tau_{\text{gen}}/0.8$B tokens

Epoch size determines peak location. Figure 3a compares configurations with $N \in \{0.85, 0.95, 1.6\}$B tokens (orange, green, red). All exhibit rise-then-fall patterns, confirming τ_{gen} exists. Larger epoch sizes delay the peak: orange peaks at $\sim$30 epochs, green at $\sim$125 epochs, and red at $\sim$175 epochs. This validates our framework—larger N provides more spacing between HQ re-exposures, allowing the model to train longer before over-memorization.

Saturation beyond the memorization window. Figure 3b compares red ($N = 1.6$B tokens) and purple ($N = 3.2$B tokens). Despite a $2\times$ difference in epoch size, both curves exhibit remarkably similar behavior—peaking at 100–150B tokens cumulative HQ tokens with comparable accuracy. This suggests *saturation*: once N exceeds τ^*, further increases provide no additional benefit. The model already has sufficient spacing to avoid over-memorization; the limiting factor becomes the data mixture or the forgetting between exposures rather than reuse interval.

LQ diversity is essential. Figure 3c compares pure HQ (blue, $N = 0.8$B tokens) with minimal LQ mixing (orange, $N = 0.85$B tokens, 6% FineWeb). Despite nearly identical epoch sizes, orange dramatically outperforms blue. This gap cannot be explained by spacing alone—the 0.05B difference is negligible. FineWeb provides distribution diversity that regularizes learning, preventing overfitting to the narrow HQ distribution. This extends our framework:

$$\text{Safe regime:} \quad N \geq \tau^*, \quad E \leq E_{\max}, \quad \rho_{\text{LQ}} > 0. \tag{10}$$

Table 1 summarizes our estimates from both experiments.

Limitations. Our study has several limitations. First, we evaluate one domain (math), one model scale (100M), one HQ budget (0.8B tokens), and one LQ source (FineWeb)—the memorization window likely varies across these factors. Second, identifying the optimal N^* by the highest peak is not fully rigorous: benchmark performance depends not only on memorization dynamics but also on the data mixture itself. For instance, smaller N configurations (e.g., green) have higher HQ ratios than larger N configurations (e.g., red, purple), confounding the comparison. Disentangling these effects requires further investigation. Finally, some configurations (e.g., $N = 0.85B$) exhibit secondary performance rises after extended training, potentially related to grokking (Li et al., 2025); we focus on the first peak and leave multi-phase dynamics to future work.

5 Discussions and Perspectives

Our work introduces the memorization window as a framework for understanding data reuse in LLM training. While our experiments provide initial validation, they also open numerous research directions. We discuss the broader implications and outline key questions for future investigation.

5.1 Implications for Data Scheduling

Beyond global shuffling. Current multi-epoch training practices typically shuffle data globally across epochs, treating all samples uniformly regardless of when they were last seen (Muennighoff et al., 2023). Our framework suggests a more principled approach: *schedule data reuse based on the memorization window*. Rather than random shuffling, samples should be re-exposed when they approach the forgetting boundary ($\tau \approx \tau^*$), maximizing the learning signal from each repetition.

Deterministic data loading. Recent work on deterministic data loading (Zuo et al., 2025) demonstrates the benefits of sequential sample processing: reproducible training, flexible checkpointing,

and dynamic mixture updates. Our framework complements this approach by providing a principled criterion for *when* to revisit data sources. Specifically, if a data source has epoch size $N < \tau^*$, the scheduler should interleave other data to increase the effective reuse interval, preventing over-memorization.

Adaptive scheduling. An exciting direction is *memorization-aware adaptive scheduling*: monitoring rollback loss during training and dynamically adjusting data replay based on per-sample or per-source memorization signals. Samples approaching the forgetting boundary could be prioritized for re-exposure, while well-memorized samples could be delayed. This would enable efficient use of limited high-quality data without manual tuning of epoch sizes.

5.2 Scaling Properties of the Memorization Window

A central open question is how the memorization window scales with model and data characteristics.

Model size. Larger models have greater capacity to retain information. We hypothesize that $\tau_{\text{ret}||}$ increases with model size—larger models forget more slowly. Counterintuitively, this implies that larger models are *more susceptible* to over-memorization: since they retain information longer, repeated exposure within the same interval provides redundant signal. To avoid overfitting, larger models may require *larger epoch sizes* N (more spacing between repetitions) or *fewer total epochs* E. Conversely, smaller models forget faster and may tolerate more frequent data reuse without over-memorization. This has practical implications: scaling up model size may necessitate scaling up unique data or extending reuse intervals accordingly.

Model architecture. Different architectures may exhibit different memorization dynamics. Depth versus width could affect how information is stored and forgotten—deeper models may retain more through compositional representations (Zuo et al., 2025). Across architecture families, standard Transformers, linear attention models, state-space models (e.g., Mamba (Gu & Dao, 2024)), and hybrid architectures (Qwen Team) likely have distinct retention properties due to their different mechanisms for information routing. Models with explicit memory or retrieval-augmented mechanisms (Huang et al., 2024; Cheng et al., 2026) may further decouple memorization from weights. Comparing τ_{ret} across architectures could reveal which designs are most efficient for learning from limited data.

Training stage. Our experiments show consistent τ_{ret} across early (5B tokens) and late (500B tokens) training. However, one might expect the memorization window to *expand* as training progresses—more capable models may retain information longer. If confirmed, the optimal epoch size N^* should increase during training: early stages could tolerate frequent reuse, while later stages require more spacing. This connects to adaptive scheduling—dynamically adjusting N based on the evolving memorization window rather than using a fixed schedule throughout training.

5.3 Data-Dependent Memorization

The memorization window likely varies across data characteristics and sources, with implications for scheduling in complex training setups.

Data complexity and domain. Not all data is equally memorable. High-entropy, complex samples (e.g., diverse reasoning chains) may be harder to memorize than low-entropy, repetitive patterns (e.g., templated text), suggesting complex data may need more frequent re-exposure. Similarly, different domains (code, natural language, scientific text) likely exhibit different memorization patterns due to varying structure and redundancy. We evaluate only mathematical reasoning; characterizing domain-specific memorization windows is an important direction. This suggests that the memorization window is an inherent property of an LLM-data pair, not just the model alone.

Complex data mixtures. Real-world pretraining involves mixtures of diverse data sources, each potentially with its own $\tau_{\text{ret}}^{(k)}$ and $\tau_{\text{gen}}^{(k)}$. Optimal scheduling would respect these heterogeneous windows rather than applying a uniform epoch size. Furthermore, training on one source may affect

retention of another—sources sharing common patterns, tokens, or skills may interfere or reinforce each other's memorization. Since sources also contribute differently to downstream performance, integrating memorization windows with data contribution estimation could enable mixture strategies that balance retention, interference, and contribution weighting.

5.4 Connection to Grokking

Our experiments reveal secondary performance rises after initial decline—a pattern reminiscent of *grokking* (Li et al., 2025), where models suddenly generalize long after memorizing training data. This raises intriguing questions:

Grokking as delayed generalization beyond the memorization window. The initial performance peak may correspond to the memorization window boundary, while the secondary rise reflects a distinct learning phase where the model discovers generalizable patterns. Understanding this multi-phase dynamics could enable training strategies that intentionally push through the initial overfitting regime to reach grokking.

Data repetition and grokking. Recent work suggests repetition can accelerate grokking (Zucchet et al., 2025). Our framework provides a lens to study this: does grokking require pushing past τ_{gen} with continued exposure? Characterizing the relationship between memorization window, repetition, and grokking could yield new training paradigms.

5.5 Broader Implications

Hyperparameter interactions. Training hyperparameters may influence memorization dynamics. Higher learning rates could accelerate both learning and forgetting, potentially shrinking the memorization window. Batch size and optimizer choice may also affect how sharply the model memorizes individual samples. However, these interactions remain speculative without controlled experiments. We use fixed hyperparameters throughout our study; systematically investigating their effect on τ_{ret} and τ_{gen} is an avenue for future work.

Efficient use of limited high-quality data. As high-quality human-generated data becomes increasingly scarce, efficient data reuse becomes critical. Current practice often extends training duration with repeated data, hoping more exposure improves performance—but this risks over-memorization and wasted compute. The memorization window framework provides a principled alternative: rather than training *longer* with blind repetition, we can train *smarter* by timing data reuse to maximize learning signal. This means reusing data when the model begins to forget ($N \approx \tau^*$), stopping before overfitting ($E \leq E_{\max}$), and allocating compute to samples that still provide learning signal rather than those already memorized.

Curriculum learning. The memorization window connects to curriculum learning (Bengio et al., 2009). Traditional curriculum learning orders samples by difficulty—easy to hard. Our framework suggests an alternative: order by memorization dynamics, presenting samples when they approach forgetting rather than when freshly memorized. This *forgetting-aware curriculum* could complement existing approaches. Notably, recent pretraining practices (Zuo et al., 2025; Team et al., 2025) have moved away from staged curricula toward uniform data mixtures throughout pretraining; our framework provides a principled basis for revisiting data scheduling strategies.

Continual learning. The retention window τ_{ret} directly relates to catastrophic forgetting in continual learning (Parmar et al., 2024). When learning new tasks sequentially, models forget previous knowledge—the same phenomenon our rollback loss measures. Our framework could inform replay buffer strategies: rather than replaying old data at fixed intervals or random sampling, replay could be timed based on $\tau_{\text{ret}||}$—revisiting data as it approaches the forgetting boundary, preventing forgetting without over-rehearsal.

6 CONCLUSION

We introduce the *Memorization Window* framework to characterize when data reuse remains beneficial during LLM pretraining. Through the loss retention gap, we formally define sample-level memorization and derived two complementary windows: the retention window τ_{ret} (when forgetting begins) and the generalization window τ_{gen} (when overfitting begins). The effective memorization window $\tau^* = \min(\tau_{\text{ret}}, \tau_{\text{gen}})$ provides a principled criterion for data reuse.

Our experiments on a 100M parameter Transformer validated the framework: larger epoch sizes delay overfitting, excessive spacing yields diminishing returns, and low-quality data provides essential diversity beyond temporal spacing. These findings yield practical guidelines for multi-epoch training—reuse data at intervals $N \geq \tau^*$ and stop before $E_{\max}$ epochs—enabling practitioners to train LLMs *smarter rather than longer* with limited high-quality data.

Limitations remain: we evaluate one domain (mathematical reasoning), one model scale (100M), and limited data configurations. More comprehensive studies are needed, e.g., investigating scaling laws across model sizes, diverse domains, data mixtures, and hyperparameter interactions. Additionally, developing efficient online methods to estimate τ^* during training would enable practical adoption. We hope this framework stimulates further research toward principled data reuse in LLM pretraining.

REFERENCES

Marah Abdin, Jyoti Aneja, Harkirat Behl, Sébastien Bubeck, Ronen Eldan, Suriya Gunasekar, Michael Harrison, Russell J Hewett, Mojan Javaheripi, Piero Kauffmann, et al. Phi-4 technical report. *arXiv preprint arXiv:2412.08905*, 2024.

Loubna Ben Allal, Anton Lozhkov, Elie Bakouch, Gabriel Martín Blázquez, Guilherme Penedo, Lewis Tunstall, Andrés Marafioti, Hynek Kydlíček, Agustín Piqueres Lajarín, Vaibhav Srivastav, et al. Smollm2: When smol goes big–data-centric training of a small language model. *arXiv preprint arXiv:2502.02737*, 2025.

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *Proceedings of the 26th annual international conference on machine learning*, pp. 41–48, 2009.

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*, 2022.

Xin Cheng, Wangding Zeng, Damai Dai, Qinyu Chen, Bingxuan Wang, Zhenda Xie, Kezhao Huang, Xingkai Yu, Zhewen Hao, Yukun Li, et al. Conditional memory via scalable lookup: A new axis of sparsity for large language models. *arXiv preprint arXiv:2601.07372*, 2026.

Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces. In *First conference on language modeling*, 2024.

Xintong Hao, Ruijie Zhu, Ge Zhang, Ke Shen, and Chenggang Li. Reformulation for pretraining data augmentation. *arXiv preprint arXiv:2502.04235*, 2025.

Danny Hernandez, Tom Brown, Tom Conerly, Nova DasSarma, Dawn Drain, Sheer El-Showk, Nelson Elhage, Zac Hatfield-Dodds, Tom Henighan, Tristan Hume, et al. Scaling laws and interpretability of learning from repeated data. *arXiv preprint arXiv:2205.10487*, 2022.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, et al. Training compute-optimal large language models. *arXiv preprint arXiv:2203.15556*, 2022.

Zihao Huang, Qiyang Min, Hongzhi Huang, Defa Zhu, Yutao Zeng, Ran Guo, and Xun Zhou. Ultra-sparse memory network. *arXiv preprint arXiv:2411.12364*, 2024.

Katherine Lee, Daphne Ippolito, Andrew Nystrom, Chiyuan Zhang, Douglas Eck, Chris Callison-Burch, and Nicholas Carlini. Deduplicating training data makes language models better. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 8424–8445, 2022.

Ziyue Li, Chenrui Fan, and Tianyi Zhou. Where to find grokking in llm pretraining? monitor memorization-to-generalization without test. *arXiv preprint arXiv:2506.21551*, 2025.

Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. *arXiv preprint arXiv:2305.20050*, 2023.

Anton Lozhkov, Elie Bakouch, Gabriel Martin Blazquez, Guilherme Penedo, Lewis Tunstall, Andrés Marafioti, Agustín Piqueres Lajarín, Hynek Kydlíček, Vaibhav Srivastav, Joshua Lochner, et al. Smollm2: When smol goes big—data-centric training of a fully open small language model. In *Second Conference on Language Modeling*.

John X Morris, Chawin Sitawarin, Chuan Guo, Narine Kokhlikyan, G Edward Suh, Alexander M Rush, Kamalika Chaudhuri, and Saeed Mahloujifar. How much do language models memorize? *arXiv preprint arXiv:2505.24832*, 2025.

Niklas Muennighoff, Alexander Rush, Boaz Barak, Teven Le Scao, Nouamane Tazi, Aleksandra Piktus, Sampo Pyysalo, Thomas Wolf, and Colin A Raffel. Scaling data-constrained language models. *Advances in Neural Information Processing Systems*, 36:50358–50376, 2023.

Team Olmo, Allyson Ettinger, Amanda Bertsch, Bailey Kuehl, David Graham, David Heineman, Dirk Groeneveld, Faeze Brahman, Finbarr Timbers, Hamish Ivison, et al. Olmo 3. *arXiv preprint arXiv:2512.13961*, 2025.

Jupinder Parmar, Sanjev Satheesh, Mostofa Patwary, Mohammad Shoeybi, and Bryan Catanzaro. Reuse, don't retrain: A recipe for continued pretraining of language models. *arXiv preprint arXiv:2407.07263*, 2024.

Guilherme Penedo, Quentin Malartic, Daniel Hesslow, Ruxandra Cojocaru, Alessandro Cappelli, Hamza Alobeidli, Baptiste Pannier, Ebtesam Almazrouei, and Julien Launay. The refinedweb dataset for falcon llm: outperforming curated corpora with web data, and web data only. *arXiv preprint arXiv:2306.01116*, 2023.

Guilherme Penedo, Hynek Kydlíček, Anton Lozhkov, Margaret Mitchell, Colin A Raffel, Leandro Von Werra, Thomas Wolf, et al. The fineweb datasets: Decanting the web for the finest text data at scale. *Advances in Neural Information Processing Systems*, 37:30811–30849, 2024.

Qwen Team. Qwen3-coder-next technical report. Technical report. URL `https://github.com/QwenLM/Qwen3-Coder/blob/main/qwen3_coder_next_tech_report.pdf`. Accessed: 2026-02-03.

Ross Taylor, Marcin Kardas, Guillem Cucurull, Thomas Scialom, Anthony Hartshorn, Elvis Saravia, Andrew Poulton, Viktor Kerkez, and Robert Stojnic. Galactica: A large language model for science. *arXiv preprint arXiv:2211.09085*, 2022.

Kimi Team, Yifan Bai, Yiping Bao, Guanduo Chen, Jiahao Chen, Ningxin Chen, Ruijue Chen, Yanru Chen, Yuankun Chen, Yutian Chen, et al. Kimi k2: Open agentic intelligence. *arXiv preprint arXiv:2507.20534*, 2025.

Shubham Toshniwal, Wei Du, Ivan Moshkov, Branislav Kisacanin, Alexan Ayrapetyan, and Igor Gitman. Openmathinstruct-2: Accelerating ai for math with massive open-source instruction data. *arXiv preprint arXiv:2410.01560*, 2024.

Xinyi Wang, Antonis Antoniades, Yanai Elazar, Alfonso Amayuelas, Alon Albalak, Kexun Zhang, and William Yang Wang. Generalization vs memorization: Tracing language models' capabilities back to pretraining data. *arXiv preprint arXiv:2407.14985*, 2024.

Fuzhao Xue, Yao Fu, Wangchunshu Zhou, Zangwei Zheng, and Yang You. To repeat or not to repeat: Insights from scaling llm under token-crisis. *Advances in Neural Information Processing Systems*, 36:59304–59322, 2023.

Tingkai Yan, Haodong Wen, Binghui Li, Kairong Luo, Wenguang Chen, and Kaifeng Lyu. Larger datasets can be repeated more: A theoretical analysis of multi-epoch scaling in linear regression. *arXiv preprint arXiv:2511.13421*, 2025.

Nicolas Zucchet, Francesco d'Angelo, Andrew K Lampinen, and Stephanie CY Chan. The emergence of sparse attention: impact of data distribution and benefits of repetition. *arXiv preprint arXiv:2505.17863*, 2025.

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