Scaling and Transferability of Annealing Strategies in Large Language Model Training

Siqi Wang[1], Zhengyu Chen[2], Teng Xiao[3], Zheqi Lv[2], Jinluan Yang[2], Xunliang Cai[2], Jingang Wang[2], Xiaomeng Li[1, 4]

[1]The Hong Kong University of Science and Technology, Hong Kong SAR
[2]Meituan Inc.
[3]Allen Institute for Artificial Intelligence, USA
[4]Shenzhen Loop Area Institute

swanggz@connect.ust.hk, chenzhengyu04@meituan.com, eexmli@ust.hk

Abstract

Learning rate scheduling is crucial for training large language models, yet understanding the optimal annealing strategies across different model configurations remains challenging. In this work, we investigate the transferability of annealing dynamics in large language model training and refine a generalized predictive framework for optimizing annealing strategies under the Warmup-Steady-Decay (WSD) scheduler. Our improved framework incorporates training steps, maximum learning rate, and annealing behavior, enabling more efficient optimization of learning rate schedules. Our work provides a practical guidance for selecting optimal annealing strategies without exhaustive hyperparameter searches, demonstrating that smaller models can serve as reliable proxies for optimizing the training dynamics of larger models. We validate our findings on extensive experiments using both Dense and Mixture-of-Experts (MoE) models, demonstrating that optimal annealing ratios follow consistent patterns and can be transferred across different training configurations.

Code — https://github.com/xmed-lab/fm-annealing

Introduction

Previous work (Kaplan et al. 2020; Hoffmann et al. 2022; Rae et al. 2021) on scaling laws has focused mainly on the general relationship of model performance and model size, compute budget or tokens. However, recent studies (Tissue, Wang, and Wang 2024; Hägele et al. 2024) have been focused on the discrepancies introduced by training dynamics with different hyperparameter settings.

When training a model, even if the amount of total tokens remains constant, different settings can lead to different loss trajectory over time (as shown in Figures 1). This suggests that the path the model follows to reach its irreducible loss varies depending on multiple hyperparameters in the training setup. Apart from batch sizes and maximum learning rates, a learning rate scheduler with warmup and annealing stages is crucial for achieving optimal performance and generalization in language models (He et al. 2015; Goyal et al. 2017; Popel and Bojar 2018; You et al. 2019a).

Firstly, our work aims to address the first question: beyond training tokens and model sizes, what factors influence the dynamics of the loss curve across different training settings? In our work, we find that training steps and annealing strategies play a crucial role, and we also observe that different batch sizes lead to distinct training loss curves (shown in Figure 1 *(Left)*). While large batch sizes are often recommended for various tasks (You et al. 2017; You, Gitman, and Ginsburg 2017; Smith, Kindermans, and Le 2017; Puri et al. 2018; Kaplan et al. 2020) to improve efficiency, we observe in Figure 1 that once the batch size exceeds a certain threshold, the training loss curves with respect to training steps tend to converge. McCandlish et al. (2018) proposed the concept of an optimal batch size, which is the smallest size that effectively approximates the true gradient. This suggests that when the batch size falls within a practical optimal range, the batch size can give a good estimation of true gradient and training steps can serve as a reliable metric for tracking loss curves, allowing us to focus on other influential factors. However, to save resources and ensure stable training, batch sizes should not be excessively large (Figure 1 *(Right)* when batch size is 2048).

Furthermore, our work addresses the second question: how do factors such as maximum learning rate, total training steps, and model size influence the annealing strategy? The optimal values of these factors are interdependent, with each influencing the selection of the others. For instance, previous studies (Goyal et al. 2017; Hoffer, Hubara, and Soudry 2017; You et al. 2019b; Granziol, Zohren, and Roberts 2020) have primarily focused on the selection and scaling of maximum learning rates with respect to batch sizes, introducing strategies such as square-root scaling and linear scaling for the SGD optimizer. Li et al. (2024) observed that for the Adam optimizer, the optimal learning rate initially increases and then decreases as the batch size grows. In this work, we go beyond individual factor selection and explore how these factors like the maximum learning rate influence the effectiveness of annealing strategies. We further propose a practical model-agnostic annealing strategy that adapts to different factors like maximum learning rate and model sizes.

Finally, we address the third question: what is the impact of the annealing strategy and LR scheduler on gener-

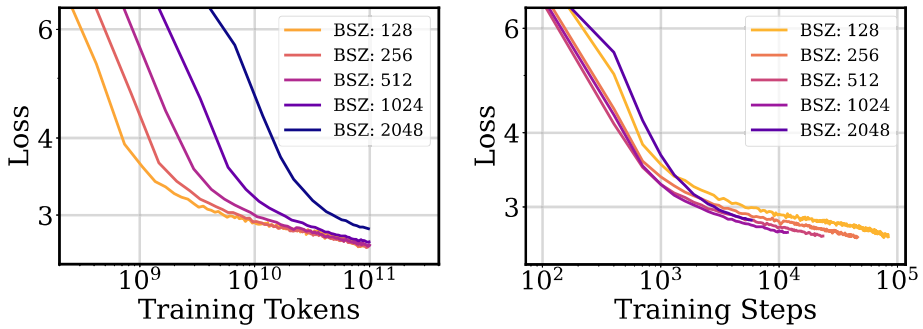

Figure 1: *(Left)* Loss vs Training tokens across different batch sizes with sequence length as 8192. *(Right)* Loss vs Training steps across different batch sizes with sequence length as 8192. The standard deviation of the loss values for batch sizes 256, 512, and 1024 is approximately 0.00847.

alization? The key factors are the accumulated *forward effect* of learning rates and the *annealing momentum*, which is a measure of the kinetic effect of learning rate decay. Annealing momentum captures both the rate and magnitude at which the LR decreases during annealing, thus characterizing the convergence dynamics in the decay phase. The annealing ratio (R) is defined as the fraction of total training steps spent in the decay phase of the learning rate schedule: $R = \frac{T_{\text{decay}}}{T_{\text{total}}}$. Tissue, Wang, and Wang (2024) proposed a scaling law to fit the loss curve across different schedulers throughout the training process, considering both forward effect and annealing degree. However, this formula assumes fixed batch sizes, and their forward term is not robust to variations in batch size. Moreover, their annealing momentum term relies on multiplicative accumulation, which can cause instability during calculation. In contrast, our approach uses integrals to compute both the forward and annealing momentum terms, which are more robust to batch size variations and irregular step counts. Additionally, we replace the multiplicative accumulation in the momentum term with an Adam-style momentum, mitigating the risk of instability and improving robustness to unstable training dynamics and hyperparameter tuning. We tested on WSD scheduler to investigate the relationship between optimal annealing ratio and factors such as total steps, then verified its transfer properties across different datasets.

Here are our key contributions:

1. *Development of a Model-Agnostic Predictive Model*: We developed a robust predictive model for training dynamics, considering forward and annealing effects along with model size. Validated on both Dense and MoE models, it demonstrates an agnostic relationship and can accurately predict the impact of learning rate schedules.

2. *Analysis of Key Factors Influencing Annealing Dynamics*: We have investigated how factors such as batch size, maximum learning rate, model architecture, and model size influence the annealing dynamics. We use training steps as a more reliable tracker for loss curves and explore the effects of factors like maximum learning rates and model characteristics on the annealing strategy.

3. *Transferability of Annealing Strategy*: We find that the optimal annealing ratio can be transferred via specific principles. This suggests the existence of a consistent and predictable principle governing the annealing dynamics across various settings.

Related Work

Large Language Models Recently, Large Language Models (LLMs) such as GPT-4 (Achiam et al. 2023), LLaMA (Touvron et al. 2023), Mistral (Jiang et al. 2023), DeepSeek (Bi et al. 2024; Guo et al. 2025), Longcat (Team et al. 2025; Chen et al. 2025b,c) and Qwen (Yang et al. 2025) have been significantly developed. Their performance in natural language understanding have gained great attention, with the number of training tokens, model sizes, and compute budgets scaling up to huge levels. However, there remain numerous aspects of their training dynamics and intermediate training loss curve that require further exploration, such as the impact of different annealing strategies on generalization scenarios.

Scaling Laws for LLMs Scaling laws (Kaplan et al. 2020; Hoffmann et al. 2022) have been extensively studied, which revealed the power-law relationship between pre-training cross-entropy loss and factors such as the amount of training tokens, model size, and compute budget. Additionally, optimal hyperparameters (McCandlish et al. 2018; Hu et al. 2024; Li et al. 2024) have been shown to scale with training loss according to specific patterns. Previous studies (Yang et al. 2022, 2023; Chen et al. 2025a) indicate that certain hyperparameters, like batch size and sequence length, can be scaled across model depth or width. However, there is a lack of investigation into the whole training dynamics for a systematic analysis.

Learning Rate Schedulers for LLMs Rae et al. (2021); Hoffmann et al. (2022) recommend the cosine scheduler (Loshchilov and Hutter 2017) for LLM pre-training. However, it is unsuitable for continued training because its period depends on the training length. Zhai et al. (2021); Raffel et al. (2019); Hu et al. (2024) proposed WSD scheduler featuring a flexible warmup-stable-decay LR scheduling. Previous works (Tissue, Wang, and Wang 2024; Hägele et al.

2024) have shown that its performance is comparable to the cosine scheduler and that scaling laws can also describe its final loss. Wen et al. (2024) introduced a variant of the WSD scheduler for continual learning to avoid performance drops after rewarming. Defazio et al. (2024); Schaipp et al. (2025) shows that the loss curves in large model training exhibit behavior similar to the performance bound derived from non-smooth convex optimization theory. In this work, we analyze the factors influencing the annealing strategy and explore its generalization across different configurations.

Forward-Momentum Scaling Law

Previous works (Kaplan et al. 2020; Hoffmann et al. 2022) have proposed empirical scaling laws to predict the final loss for large language models:

$$L = \frac{\lambda_N}{N^{\alpha_N}} + \frac{\lambda_D}{D^{\alpha_D}} + \sigma \tag{1}$$

where N is the model size, and D is the number of training tokens. The parameters α_N, α_D, λ_N, and λ_D are coefficients, and σ is the irreducible loss. However, this scaling law cannot capture differences in training loss curves when the same model is trained with the same number of tokens but different batch sizes or learning rate schedulers.

To better capture the discrepancies between different training settings, we tested training curves across various configurations with identical token counts and model sizes (Figure 1). The training curves vary significantly when different batch sizes are used, even with the same maximum learning rate and scheduler. Yet we found that when tracking the loss curve using training steps, the curves tend to converge and become more alike.

This observation indicates that the trajectory of model optimization is more stable with respect to steps than tokens, especially in a practical batch size range, where training efficiency does not vary drastically. This aligns with the intuition that, under similar optimization dynamics (like the same scheduler), the number of parameter updates is a more direct indicator of training progress than the total number of tokens processed. The steps-based normalization implicitly captures the optimization view of training, treating each gradient update as a unit of progress, in contrast to token-based views which are more aligned with data consumption rather than model convergence.

Previous works (Kaplan et al. 2020; Fedus, Zoph, and Shazeer 2021) have shown that tracking loss with respect to training steps is practical under the assumptions of infinite data and stable, Adam-optimized training. Their experiments demonstrate that loss follows a power-law relationship with training steps, providing a theoretical foundation for our choice of step-based modeling. Therefore, we adopt training steps as the primary unit to analyze and compare training dynamics, which contributes a forward effect on loss decreasing. Based on Equation 1, all training steps after the warm-up period are included:

$$L(N, T) = \frac{\lambda_N}{N^{\alpha_N}} + \frac{\lambda_T}{T^{\alpha_T}} + \sigma \tag{2}$$

where T is the training steps required to reach a targeted loss with fixed a batch size. And $\lambda_N, \alpha_N, \lambda_T, \alpha_T$ are all coefficients, σ is the irreducible loss.

Besides the forward effects brought by training steps, the annealing phase (cooldown phase or decay phase) also plays a crucial role in loss reduction (Hägele et al. 2024; Granziol, Zohren, and Roberts 2020). Learning rate annealing minimizes the oscillation of the parameter updates and brings the model closer to the minimum point. This behavior cannot be captured solely by the power-law term of training steps. Yet previous work (Tissue, Wang, and Wang 2024) proposed an annealing term to describe the behavior during the annealing stage. Inspired by the empirical observation and previous works, we decided to incorporate a momentum term M to describe the annealing behavior. The new formula is:

$$L = \lambda_S \cdot S^{-\alpha} + \lambda_M \cdot M + L_0 \tag{3}$$

where $L_0, \lambda_S, \lambda_M$ are coefficients. S is the integral of learning rate η with respect to steps (Equation 4) and M is the integral of momentum with respect to steps (Equation 5).

$$S = \int_0^T \eta(t)\, dt \tag{4}$$

where $\eta(t)$ represents the learning rate as a function of the training step t, and T is the total number of training steps.

$$M = \int_0^T \text{momentum}(t)\, dt \tag{5}$$

where $\text{momentum}(t)$ represents the momentum as a function of training step t, and T is the total number of training steps. Based on this definition, we can calculate M using the following steps. Firstly, the momentum m_t and second moment v_t are updated at each step using the formulas:

$$m_t = \beta_1 \cdot m_{t-1} + (1 - \beta_1) \cdot \Delta\eta_t \tag{6}$$

$$v_t = \beta_2 \cdot v_{t-1} + (1 - \beta_2) \cdot (\Delta\eta_t)^2 \tag{7}$$

Then bias correction is applied to both moments:

$$\hat{m}_t = \frac{m_t}{1 - \beta_1^t}, \quad \hat{v}_t = \frac{v_t}{1 - \beta_2^t} \tag{8}$$

The cumulative momentum M_t is updated as:

$$M_t = M_{t-1} + \frac{\hat{m}_t}{\sqrt{\hat{v}_t + \epsilon}} \tag{9}$$

This method adjusts the influence of the first and second moments over time, providing a refined estimate of the momentum, and stabilizing updates with small ϵ values. Thus, S captures the forward force to reduce the training loss, and M captures the loss reduction brought by the learning rate scheduler and annealing strategy.

Finally, we incorporate the model size term:

$$L = \lambda_S \cdot S^{-\alpha_S} + \lambda_N \cdot N^{-\alpha_N} + \lambda_M \cdot M + L_0 \tag{10}$$

where L is the training loss, M and S are the integrals of momentum and learning rate with respect to steps. L_0, λ_S, λ_M, λ_N, α_S, and α_N are coefficients.

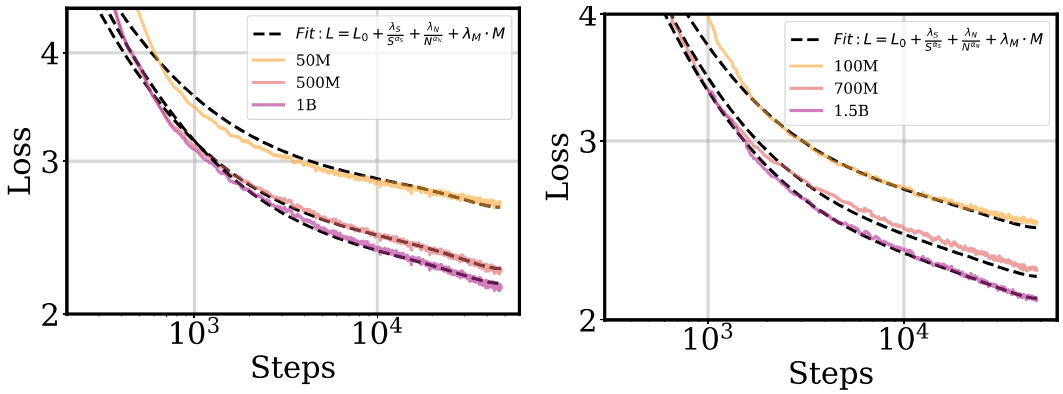

Figure 2: Fitting results across model sizes. *(Left)* Dense models (50M, 500M, 1B) with batch size 256 and learning rate 5e-5. *(Right)* MoE models (100M, 700M, 1.5B) with batch size 256 and learning rate 2e-4. All use cosine learning rate scheduler.

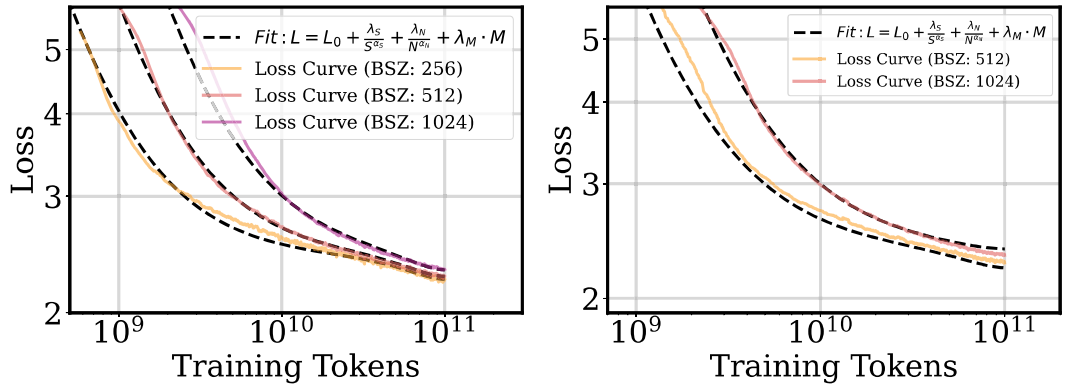

Figure 3: Fitting results across different batch sizes using cosine scheduler. *(Left)* Dense 500M models with batch sizes 256, 512, 1024 and max LR of 6e-4. *(Right)* MoE 700M models with batch sizes 512 and 1024 using max LR of 2e-4.

Training Loss Curve Fitting

Loss curves vary significantly depending on various hyperparameters. In this section, we investigate whether the loss curve can be accurately modeled across different training configurations within acceptable error margins, such as model sizes, batch sizes, and learning rate schedulers. We compare our Adam-style momentum formulation with multiplicative accumulations (CMMT) (Tissue, Wang, and Wang 2024), indicating Adam-style updates achieve the best stability and generalizability in transfer scenarios (more details shown in supplementary materials).

Across Model Sizes Fitting

Equation 1 can be used to model final loss across different model sizes. However, to capture the full training dynamics, we propose a more comprehensive formulation: Equation 10, and we apply it to both MoE and Dense models.

As shown in Figure 2, we first control for batch size, total training steps, maximum learning rate, and scheduler type to ensure consistency. For Dense models (50M, 500M, and 1B), we use a fixed batch size of 256 and a cosine scheduler with a maximum learning rate of 5e-5. Our fitted loss curves achieve an average MAPE (Mean Absolute Percentage Error) below 2%. Similarly, for MoE models (100M, 700M, and 1.5B), using the same batch size and a maximum learning rate of 2e-4, the MAPE loss remains consistently below 2%. The sequence length is 8192. These results confirm that Equation 10 effectively captures loss trends across model sizes, reinforcing its generalizability.

Observation 1: Model Size Dependence The loss curves exhibit a power-law dependence on model size, as described in Equation 10, enabling consistent extrapolation across different model scales.

Across Batch Sizes Fitting

Training the same model with different batch sizes results in different training curves when using token-based loss tracking. However, our experiments show that when the batch size exceeds a certain threshold, the step-based training loss curves become close to each other.

The threshold corresponds to the optimal batch size (B_{opt}). B_{opt} is the minimum size that can approximate the

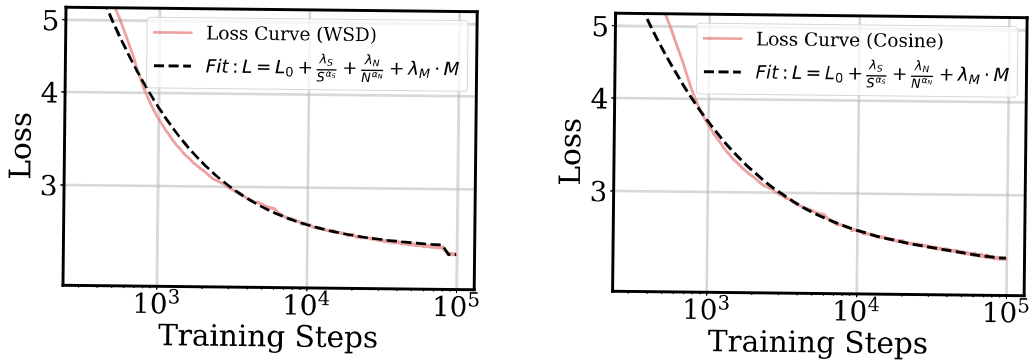

Figure 4: Fitting results for 100M Dense models using different learning rate schedulers (Cosine and WSD) with the same total training steps and max LR. *(Left)* Cosine scheduler loss curves used to predict those of WSD. *(Right)* Vice versa.

true gradient, offering a good approximation while minimizing computational cost (McCandlish et al. 2018; Li et al. 2024). We assume that a wide range of batch sizes (greater than B_{opt}) with the same learning rate schedule produce similar loss v.s.steps curves. To verify our assumption, we fit the training curves using the same model size, maximum learning rate, and schedulers, varying only the batch size values (greater than B_{opt}). To ensure its generalization, we test this on both MoE and Dense models.

Firstly, we estimate the scaling laws between loss values and the optimal batch sizes. By examining the training curves across different batch sizes, we select and estimate the batch sizes that allow the model to use the minimum number of training tokens to reach a specific loss value (Hu et al. 2024). We estimate that the optimal batch size for 500M Dense Model is about 293 (with sequence length as 8192), while that for 700M MoE is about 366 (with sequence length as 8192). After determining the corresponding batch size interval, we fit the selected training curves. The results show that step-based loss tracking is more effective for both (More details are shown in Figure 3).

Observation 2: Batch Size Dependence For a fixed model, a wide range of batch sizes larger than B_{opt}, when paired with the same learning-rate schedule, produce nearly identical training curves. The optimal batch size B_{opt} follows a power-law relationship with the training loss:

$$B_{\text{opt}} = \frac{\lambda_B}{L^{\alpha_B}} \tag{11}$$

where L is the training loss and B_{opt} is the optimal batch size. The parameters λ_B and α_B are coefficients.

For MoE Models, $\alpha_B \approx 3.430 \times 10^0$ and $\lambda_B \approx 4.390 \times 10^7$, while for Dense Models, $\alpha_B \approx 2.710 \times 10^0$ and $\lambda_B \approx 2.711 \times 10^7$.

Across Schedulers Fitting

Based on previous work (Tissue, Wang, and Wang 2024), Equation 3 focuses on capturing the impact of the learning rate schedule and annealing strategy. The power-law dependencies in S and N are consistent across different schedulers, while the momentum term M captures the specific behavior for each. It is optimistically expected that Equation 3 can be transferable across different schedulers.

Model Size	Cos Scheduler	WSD Scheduler
50M	0.456 ± 0.061	0.720 ± 0.132
100M	0.232 ± 0.020	0.410 ± 0.019
500M	0.453 ± 0.038	0.798 ± 0.014

Table 1: MAPE (%) loss prediction. "Cos Scheduler" refers to predicting the cosine scheduler's loss curve using the WSD-fitted model, and vice versa. Results are reported as mean $\pm$ std over 3 random seeds.

In our work, we primarily focus on two types of schedulers: the cosine scheduler and the WSD scheduler. For the cosine scheduler, the total number of steps is equal to the period of the scheduler. In contrast, the WSD scheduler consists of a warmup, constant, and linear decay phase. Its training loss curve typically ends with a sharp drop during the annealing stage. The formula for the WSD scheduler is:

$$\eta(t) = \begin{cases} \eta_{\max} \cdot \frac{t}{T_{\text{warmup}}}, & 0 \le t < T_{\text{warmup}} \\ \eta_{\max}, & T_{\text{warmup}} \le t < T_{\text{constant}} \\ \eta_{\max} \cdot \left(1 - \frac{t - T_{\text{constant}}}{T_{\text{decay}}}\right)^{\beta}, & T_{\text{constant}} \le t < T_{\text{total}} \end{cases} \tag{12}$$

where $\beta = 1$ (Details shown in supplementary materials).

We verified that the fitting results for the cosine scheduler's loss curve can predict those of the WSD scheduler, and vice versa (Figure 4 and Table 1). More detailed comparison is shown in supplementary materials.

Observation 3: Scheduler Dependence Different learning-rate schedulers influence the loss curve in a predictable manner, as captured by Equation 3.

Transfer of Optimal Annealing Ratio

It is crucial for schedulers to balance the forward effect of large learning rate with the convergence effect of annealing.

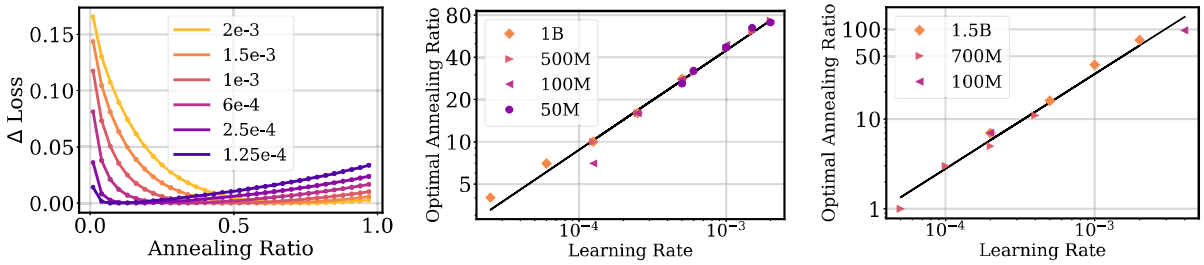

Figure 5: Optimal annealing ratio across schedulers and maximum learning rates. *(Left)* Δ_{Loss} vs. annealing ratio under different maximum learning rates for the 500M Dense model. *(Middle)* Optimal annealing ratio vs. maximum learning rate across model sizes for Dense models. *(Right)* Same plot for MoE models. All experiments use batch size 512, sequence length 8192, and 24k steps. Δ_{Loss} is defined as gap between the final loss at each annealing ratio and the lowest final loss observed at R_{opt}.

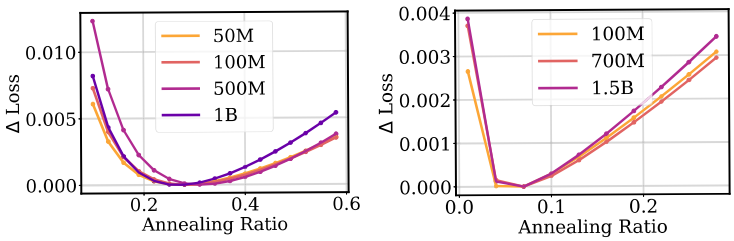

Figure 6: Transferability of optimal annealing ratio across model sizes. *(Left)* Dense models with batch size 512, sequence length 8192, 24k steps, and max LR of 4e-5. *(Right)* MoE models with the same batch size and steps, and max LR of 2e-4.

A larger initial learning rate helps prevent the model from getting stuck in local minima. As training progresses and the model begins to converge, annealing stage allows model to refine its understanding of the data and capture more subtle, complex patterns. For the cosine scheduler, the optimal annealing strategy is to set the period of the scheduler equal to the total number of steps. However, for the WSD scheduler, selecting the optimal annealing ratio is more complex, which should consider the total steps, maximum learning rate, model sizes, model architectures and so on.

However, the annealing behavior across different configurations has not always been well understood yet. In this section, we will validate whether the optimal annealing ratio (R_{opt}) can be transferable across different training settings (maximum learning rates, model sizes, datasets, and training steps) to enable efficient and robust training.

Across LR_{max} Transferability

The maximum learning rate after warmup phase is crucial for training language models. Kaplan et al. (2020) suggests that smaller models can tolerate larger learning rates, while larger models require smaller learning rates to avoid divergence. And its impact on training becomes more significant when model size becomes larger. In our work, we fit the training curves for different maximum learning rates across each model size (as shown in Figure 5 *(Left)*). We then estimate the optimal annealing ratio for each corresponding maximum learning rate (Equation 13).

Observation 1: R_{opt} Scaling Across $\eta_{\max}$ Empirically, the optimal annealing ratio follows a power-law relationship with the maximum learning rate:

$$R_{\text{opt}} = \lambda_\eta \cdot \eta_{\max}^{\alpha_\eta} \tag{13}$$

where R_{opt} is the optimal annealing ratio and $\eta_{\max}$ is the maximum learning rate. λ_η and α_η are coefficients.

This power-law behavior is consistent across different model sizes, regardless of whether the models are Dense or MoE (as shown in Figure 5 *(Middle)* and *(Right)*). For Dense Models, $\lambda_\eta \approx 5.996 \times 10^3$ and $\alpha_\eta \approx 7.09 \times 10^{-1}$. While for MoE Models, $\lambda_\eta \approx 4.675 \times 10^4$ and $\alpha_\eta \approx 1.056$.

Across Model Transferability

Yang et al. (2022, 2023) propose that the optimal hyperparameters for smaller models can be scaled and transferred to their larger counterparts. Thus, smaller models can serve as proxies for larger ones. Therefore, we assume that the optimal annealing ratios R_{opt} between small and large models can be transferred according to a certain principle.

We fit the coefficients across different model sizes with fixed maximum learning rate, total steps and batch size. Then the forward term and momentum term values are calculated to estimate the final losses at the end of training. We plot the gap (Δ_{Loss}) between the final loss at each annealing ratio and the lowest final loss observed at R_{opt}, to quantify the sensitivity of convergence to annealing choices across model sizes. As shown in Figure 6, the results show that R_{opt} can be transferred across different model sizes for both Dense and MoE Models. This strengthens the assumption that certain aspects of training dynamics are stable across architectures and sizes when hyperparameters are matched.

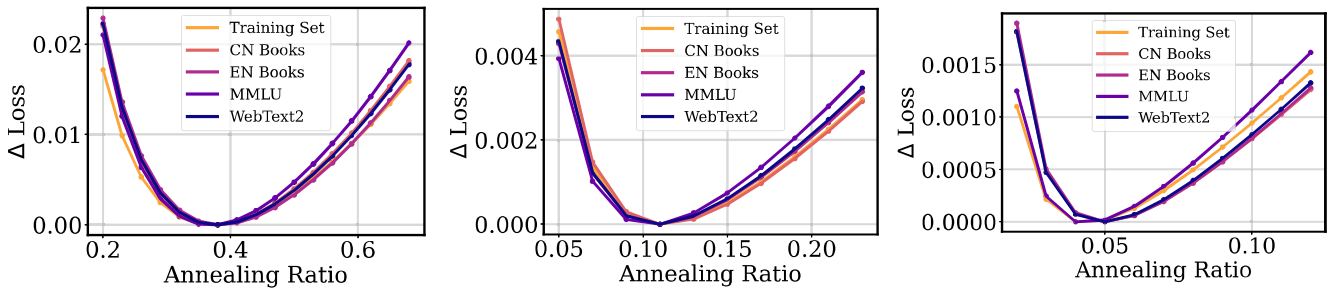

Figure 7: Optimal annealing ratio transferability for the 1B MoE model across different validation sets, under a fixed max learning rate of 2e-4 and batch size of 640. Each subplot shows Δ_{Loss} vs. annealing ratio at different total training steps. *(Left)* 28.9k steps, *(Middle)* 111k steps, *(Right)* 227k steps.

Observation 2: R_{opt} Scaling Across Model Sizes Empirically, the optimal annealing ratio remains stable and transferable across different model sizes for Dense and MoE.

Noticeably, the performance gaps Δ_{Loss} between different annealing ratios become larger when increasing the model size. Yet the optimal annealing ratio across different model sizes converges at the same value.

Across Dataset Transferability

We further verify that the optimal annealing ratio (R_{opt}) predicted from training loss also generalizes across validation datasets with minimal degradation, suggesting that training dynamics capture generalization trends to a significant degree. We demonstrate the details and our findings using 1B MoE Models with WSD scheduler when maximum learning rate is 2e-4 and batch size is 640 with different steps. The absolute final loss error is within 0.003.

Figure 7 shows that R_{opt} remains consistent across training and validation sets. Yet it is shown that the loss gaps (Δ_{Loss}) can be more significant when we transfer the annealing ratio to validation sets. Besides, it is indicated that with the increasing of the total training steps, the optimal annealing ratio is decreasing. If we enter the annealing stage too much early, we risk overfitting. Moreover, the performance discrepancies (Δ_{Loss}) between different ratios becomes smaller when increasing training steps. That means that longer training steps generally allow the model to explore the parameter space more thoroughly, leading to better generalization. It indicates a convergence when training a model for a large number of steps.

Observation 3: R_{opt} Scaling Across Datasets Empirically, the optimal annealing ratio remains consistent across training and validation sets.

Across Steps Transferability

From previous section, we observed that the optimal annealing ratio (R_{opt}) decreases as the number of total training steps (T) increases. To quantify this relationship , we computed the optimal annealing ratio across different training steps and visualized the results (Equation 14) in Figure 8.

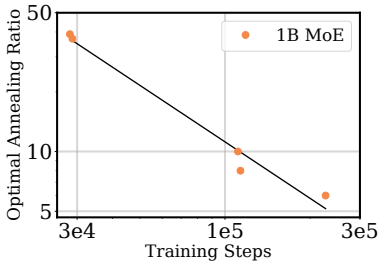

Figure 8: Optimal annealing ratio vs. training steps for 1B MoE models. Total steps are 30k, 100k, 300k, all with fixed batch size 640 and maximum learning rate 2e-4.

Observation 4: R_{opt} Scaling Across Steps T Empirically, R_{opt} and T follows a power-law form:

$$R_{\text{opt}} = \lambda_T \cdot T^{\alpha_T} \tag{14}$$

where R_{opt} represents the optimal annealing ratio, and T denotes the number of training steps. The parameters λ_T and α_T are data-dependent coefficients.

Specifically, for 1B MoE models, we estimate $\lambda_T \approx 5.987 \times 10^5$ and $\alpha_T \approx -9.46 \times 10^{-1}$. This implies that when using the WSD scheduler with a batch size within the optimal range, the annealing strategy can be directly inferred from the total training steps or maximum learning rate.

Conclusion

In this paper, we demonstrate that the training steps could be a better tracker for loss curves than total training tokens, especially when the batch size is larger than the threshold (the optimal batch size). Then we investigate three key factors influencing training loss curves: the cumulative forward effect, the annealing momentum over training steps and model size. Then we validate the prediction accuracy across various scenarios. Finally, through estimating the accumulated momentum value, we examine its transferability across varying total training steps, model sizes, maximum learning rates, and validation datasets. These results show that annealing policies are transferable and predictable across configurations, enabling efficient LLM training.

Acknowledgments

This work was partially supported by research grants from the Research Grants Council (RGC) of the Hong Kong Special Administrative Region, China (Project Nos. R6005-24 and AoE/E-601/24-N), and the Hong Kong Joint Research Scheme (JRS) of the National Natural Science Foundation of China (NSFC)/RGC (Project No. N_HKUST654/24). We would like to thank Ziqing Xu for his insightful advice on the implementation details. This work was also supported by the Hong Kong University of Science and Technology (HKUST) CSE PhD Fellowship.

References

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

Bi, X.; Chen, D.; Chen, G.; Chen, S.; Dai, D.; Deng, C.; Ding, H.; Dong, K.; Du, Q.; Fu, Z.; et al. 2024. Deepseek llm: Scaling open-source language models with longtermism. *arXiv preprint arXiv:2401.02954*.

Chen, Z.; Wang, S.; Xiao, T.; Wang, Y.; Chen, S.; Cai, X.; He, J.; and Wang, J. 2025a. Revisiting scaling laws for language models: The role of data quality and training strategies. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 23881–23899.

Chen, Z.; Wang, Y.; Xiao, T.; Zhou, R.; Yang, X.; Wang, W.; Sui, Z.; and Wang, J. 2025b. From Mathematical Reasoning to Code: Generalization of Process Reward Models in Test-Time Scaling. *arXiv preprint arXiv:2506.00027*.

Chen, Z.; Yang, J.; Xiao, T.; Zhou, R.; Zhang, L.; Xi, X.; Shi, X.; Wang, W.; and Wang, J. 2025c. Can Tool-Integrated Reinforcement Learning Generalize Across Diverse Domains? *arXiv preprint arXiv:2510.11184*.

Defazio, A.; Cutkosky, A.; Mehta, H.; and Mishchenko, K. 2024. Optimal Linear Decay Learning Rate Schedules and Further Refinements. arXiv:2310.07831.

Fedus, W.; Zoph, B.; and Shazeer, N. M. 2021. Switch Transformers: Scaling to Trillion Parameter Models with Simple and Efficient Sparsity. *ArXiv*, abs/2101.03961.

Goyal, P.; Dollár, P.; Girshick, R. B.; Noordhuis, P.; Wesolowski, L.; Kyrola, A.; Tulloch, A.; Jia, Y.; and He, K. 2017. Accurate, Large Minibatch SGD: Training ImageNet in 1 Hour. *ArXiv*, abs/1706.02677.

Granziol, D.; Zohren, S.; and Roberts, S. J. 2020. Learning Rates as a Function of Batch Size: A Random Matrix Theory Approach to Neural Network Training. *J. Mach. Learn. Res.*, 23: 173:1–173:65.

Guo, D.; Yang, D.; Zhang, H.; Song, J.; Zhang, R.; Xu, R.; Zhu, Q.; Ma, S.; Wang, P.; Bi, X.; et al. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.

Hägele, A.; Bakouch, E.; Kosson, A.; Allal, L. B.; Von Werra, L.; and Jaggi, M. 2024. Scaling Laws and Compute-Optimal Training Beyond Fixed Training Durations. *arXiv preprint arXiv:2405.18392*.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2015. Deep Residual Learning for Image Recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 770–778.

Hoffer, E.; Hubara, I.; and Soudry, D. 2017. Train longer, generalize better: closing the generalization gap in large batch training of neural networks. *ArXiv*, abs/1705.08741.

Hoffmann, J.; Borgeaud, S.; Mensch, A.; Buchatskaya, E.; Cai, T.; Rutherford, E.; de Las Casas, D.; Hendricks, L. A.; Welbl, J.; Clark, A.; Hennigan, T.; Noland, E.; Millican, K.; van den Driessche, G.; Damoc, B.; Guy, A.; Osindero, S.; Simonyan, K.; Elsen, E.; Rae, J. W.; Vinyals, O.; and Sifre, L. 2022. Training Compute-Optimal Large Language Models. *ArXiv*, abs/2203.15556.

Hu, S.; Tu, Y.; Han, X.; He, C.; Cui, G.; Long, X.; Zheng, Z.; Fang, Y.; Huang, Y.; Zhao, W.; Zhang, X.; Thai, Z. L.; Zhang, K.; Wang, C.; Yao, Y.; Zhao, C.; Zhou, J.; Cai, J.; Zhai, Z.; Ding, N.; Jia, C.; Zeng, G.; Li, D.; Liu, Z.; and Sun, M. 2024. MiniCPM: Unveiling the Potential of Small Language Models with Scalable Training Strategies. *ArXiv*, abs/2404.06395.

Jiang, A. Q.; Sablayrolles, A.; Mensch, A.; Bamford, C.; Chaplot, D. S.; Casas, D. d. l.; Bressand, F.; Lengyel, G.; Lample, G.; Saulnier, L.; et al. 2023. Mistral 7B. *arXiv preprint arXiv:2310.06825*.

Kaplan, J.; McCandlish, S.; Henighan, T.; Brown, T. B.; Chess, B.; Child, R.; Gray, S.; Radford, A.; Wu, J.; and Amodei, D. 2020. Scaling Laws for Neural Language Models. *ArXiv*, abs/2001.08361.

Li, S.; Zhao, P.; Zhang, H.; Sun, X.; Wu, H.; Jiao, D.; Wang, W.; Liu, C.; Fang, Z.; Xue, J.; Tao, Y.; Cui, B.; and Wang, D. 2024. Surge Phenomenon in Optimal Learning Rate and Batch Size Scaling. *ArXiv*, abs/2405.14578.

Loshchilov, I.; and Hutter, F. 2017. SGDR: Stochastic Gradient Descent with Warm Restarts. arXiv:1608.03983.

McCandlish, S.; Kaplan, J.; Amodei, D.; and Team, O. D. 2018. An empirical model of large-batch training. *arXiv preprint arXiv:1812.06162*.

Popel, M.; and Bojar, O. 2018. Training Tips for the Transformer Model. *The Prague Bulletin of Mathematical Linguistics*, 110: 43 – 70.

Puri, R.; Kirby, R.; Yakovenko, N.; and Catanzaro, B. 2018. Large Scale Language Modeling: Converging on 40GB of Text in Four Hours. *2018 30th International Symposium on Computer Architecture and High Performance Computing (SBAC-PAD)*, 290–297.

Rae, J. W.; Borgeaud, S.; Cai, T.; Millican, K.; Hoffmann, J.; Song, F.; Aslanides, J.; Henderson, S.; Ring, R.; Young, S.; et al. 2021. Scaling language models: Methods, analysis & insights from training gopher. *arXiv preprint arXiv:2112.11446*.

Raffel, C.; Shazeer, N. M.; Roberts, A.; Lee, K.; Narang, S.; Matena, M.; Zhou, Y.; Li, W.; and Liu, P. J. 2019. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. *J. Mach. Learn. Res.*, 21: 140:1–140:67.

Schaipp, F.; Hägele, A.; Taylor, A.; Simsekli, U.; and Bach, F. 2025. The Surprising Agreement Between Convex Optimization Theory and Learning-Rate Scheduling for Large Model Training. arXiv:2501.18965.

Smith, S. L.; Kindermans, P.-J.; and Le, Q. V. 2017. Don't Decay the Learning Rate, Increase the Batch Size. *ArXiv*, abs/1711.00489.

Team, M. L.; Li, B.; Lei, B.; Wang, B.; Rong, B.; Wang, C.; Zhang, C.; Gao, C.; Zhang, C.; Sun, C.; et al. 2025. Longcat-flash technical report. *arXiv preprint arXiv:2509.01322*.

Tissue, H.; Wang, V.; and Wang, L. 2024. Scaling Law with Learning Rate Annealing. *arXiv preprint arXiv:2408.11029*.

Touvron, H.; Martin, L.; Stone, K.; Albert, P.; Almahairi, A.; Babaei, Y.; Bashlykov, N.; Batra, S.; Bhargava, P.; Bhosale, S.; et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Wen, K.; Li, Z.; Wang, J.; Hall, D.; Liang, P.; and Ma, T. 2024. Understanding Warmup-Stable-Decay Learning Rates: A River Valley Loss Landscape Perspective. *ArXiv*, abs/2410.05192.

Yang, A.; Li, A.; Yang, B.; Zhang, B.; Hui, B.; Zheng, B.; Yu, B.; Gao, C.; Huang, C.; Lv, C.; Zheng, C.; Liu, D.; Zhou, F.; Huang, F.; Hu, F.; Ge, H.; Wei, H.; Lin, H.; Tang, J.; Yang, J.; Tu, J.; Zhang, J.; Yang, J.; Yang, J.; Zhou, J.; Zhou, J.; Lin, J.; Dang, K.; Bao, K.; Yang, K.; Yu, L.; Deng, L.; Li, M.; Xue, M.; Li, M.; Zhang, P.; Wang, P.; Zhu, Q.; Men, R.; Gao, R.; Liu, S.; Luo, S.; Li, T.; Tang, T.; Yin, W.; Ren, X.; Wang, X.; Zhang, X.; Ren, X.; Fan, Y.; Su, Y.; Zhang, Y.; Zhang, Y.; Wan, Y.; Liu, Y.; Wang, Z.; Cui, Z.; Zhang, Z.; Zhou, Z.; and Qiu, Z. 2025. Qwen3 Technical Report. arXiv:2505.09388.

Yang, G.; Hu, J. E.; Babuschkin, I.; Sidor, S.; Liu, X.; Farhi, D.; Ryder, N.; Pachocki, J. W.; Chen, W.; and Gao, J. 2022. Tensor Programs V: Tuning Large Neural Networks via Zero-Shot Hyperparameter Transfer. *ArXiv*, abs/2203.03466.

Yang, G.; Yu, D.; Zhu, C.; and Hayou, S. 2023. Tensor Programs VI: Feature Learning in Infinite-Depth Neural Networks. *ArXiv*, abs/2310.02244.

You, K.; Long, M.; Jordan, M. I.; and Wang, J. 2019a. Learning Stages: Phenomenon, Root Cause, Mechanism Hypothesis, and Implications. *ArXiv*, abs/1908.01878.

You, Y.; Gitman, I.; and Ginsburg, B. 2017. Large Batch Training of Convolutional Networks. *arXiv: Computer Vision and Pattern Recognition*.

You, Y.; Li, J.; Reddi, S. J.; Hseu, J.; Kumar, S.; Bhojanapalli, S.; Song, X.; Demmel, J.; Keutzer, K.; and Hsieh, C.-J. 2019b. Large Batch Optimization for Deep Learning: Training BERT in 76 minutes. *arXiv: Learning*.

You, Y.; Zhang, Z.; Hsieh, C.-J.; and Demmel, J. 2017. 100-epoch ImageNet Training with AlexNet in 24 Minutes. *ArXiv*, abs/1709.05011.

Zhai, X.; Kolesnikov, A.; Houlsby, N.; and Beyer, L. 2021. Scaling Vision Transformers. *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 1204–1213.