How to Set the Batch Size for Large-Scale Pre-training?

Anonymous ACL submission

Abstract

The concept of Critical Batch Size, as pioneered by OpenAI, has long served as a foundational principle for large-scale pre-training. However, with the paradigm shift towards the Warmup-Stable-Decay (WSD) learning rate scheduler, we observe that the original theoretical framework and its underlying mechanisms fail to align with new pre-training dynamics. To bridge this gap between theory and practice, this paper derives a revised $E(S)$ relationship tailored for WSD scheduler, characterizing the trade-off between training data consumption E and steps S during pre-training. Our theoretical analysis reveals two fundamental properties of WSD-based pre-training: 1) $B_{\min}$, the minimum batch size threshold required to achieve a target loss, and 2) B_{opt}, the optimal batch size that maximizes data efficiency by minimizing total tokens. Building upon these properties, we propose a dynamic Batch Size Scheduler. Extensive experiments demonstrate that our revised formula precisely captures the dynamics of large-scale pre-training, and the resulting scheduling strategy significantly enhances both training efficiency and final model quality.

1 Introduction

The continuous evolution of Large Language Models (LLMs) (Bi et al., 2024; Team et al., 2025) is perpetually expanding the frontiers of artificial intelligence, driven the large-scale pre-training. As the scale of pre-training continues to expand, the selection of optimal training strategies becomes paramount. A central challenge in this endeavor is the configuration of batch size to achieve trade-off between training efficiency and performance.

Foundational research on batch size for large-scale pre-training originates from OpenAI (McCandlish et al., 2018), which introduced the concept of **Critical Batch Size** to characterize the trade-off between token consumption E and steps S during pre-training. Building on this foundation, OpenAI further established the scaling laws for LLMs (Kaplan et al., 2020), a milestone that significantly catalyzed the revolution in generative artificial intelligence.

Concurrently, the pre-training paradigm has undergone significant evolution, most notably the transition in learning rate schedulers (LRS). The Warmup-Stable-Decay (WSD) LRS has increasingly replaced the traditional cosine LRS and gained widespread adoption in state-of-the-art models (Bi et al., 2024; Team et al., 2025; Hu et al., 2024). However, we discover that under the WSD LRS, the relationship between data consumption (E) and steps (S) during pre-traing no longer adheres to OpenAI's original $E(S)$ formula. This discrepancy implies that the underlying mechanism of critical batch size is no longer valid in current pre-training regimes, revealing a profound gap between theoretical foundations and engineering practice.

To bridge this theoretical divide, this paper derives a novel $E(S)$ relationship tailored for modern large-scale pre-training (i.e., adopting WSD LRS). Based on this new formulation, we reveal two intrinsic properties of the "Stable" phase in WSD pre-training: **Threshold Constraint** ($B_{\min}$): To achieve a specific target loss, the batch size must exceed a certain physical threshold. **Efficiency Optimality** (B_{opt}): There exists an optimal batch size that minimizes the total data consumption required to reach the target loss. Furthermore, Based on these insights of both $B_{\min}$ and B_{opt} exhibit an upward trend as training loss decreases, we introduce a novel batch size scheduler.

The core contributions of this paper are threefold:

Theoretical Reconstruction: We are the first to explicitly identify the limitations of existing batch size theories under the WSD paradigm and establish a new $E(S)$ formula that accurately describes the modern pre-training process.

Property Discovery and Methodological Inno-

vation: Based on the new $E(S)$ relationship, we reveal two essential properties of the large-scale pre-training—$B_{\min}$ and B_{opt}—and elucidate their evolution mechanisms, leading to a new Batch Size Scheduler for large-scale pre-training.

Experimental Validation: Extensive experimental results demonstrate that our proposed $E(S)$ formula precisely captures the dynamics between data consumption (E) and steps (S) during pre-traing, and the resulting Batch Size Scheduler significantly enhances the quality of pre-training.

2 Related Work

2.1 The impact of batch size on model training dynamics

Batch size, a pivotal hyperparameter in model training, has garnered extensive attention from both academia and industry. Keskar et al. (2017) were among the first to investigate its impact on model generalization, observing that—unlike small batch sizes—training with large batch sizes tends to result in convergence to sharp minima, thereby degrading generalization performance. McCandlish et al. (2018) subsequently introduced a novel perspective by proposing the concept of Critical Batch Size to characterize the trade-off between training efficiency and batch size. Furthermore, they derived the renowned relationship between the total data consumption E and the number of optimization steps S required to reach a specific loss, known as the $E(S)$ formula:

$$(\frac{E}{E_{min}} - 1)(\frac{S}{S_{min}} - 1) = 1. \quad (1)$$

Extending the critical batch size framework, Kaplan et al. (2020) formalized the scaling laws governing neural language models. They demonstrated that model performance scales as a predictable power-law function of model size, data volume, and compute.

Distinct from Critical Batch Size, Optimal Batch Size characterizes the relationship between batch size and final model performance. However, although scaling laws have driven an exponential increase in model scale, the prohibitive experimental costs have severely limited research into optimal batch size. To address this, Bi et al. (2024) investigated the scaling properties of optimal batch size, revealing that it relates to the compute budget via a power law:

$$B_{opt} = 0.2920C^{0.3271}. \quad (2)$$

Crucially, this scaling law enables the extrapolation of optimal batch sizes for large-scale training from low-cost, small-scale experiments. Beyond compute, recent studies have further established power-law dependencies between optimal batch size and other key dimensions, specifically model size and data volume (Shuai et al., 2024; Li et al., 2025).

While dynamic batch size scheduling was briefly touched upon in large-scale model training (Bi et al., 2024; MiniMax et al., 2025), the theoretical principles guiding these schedules remain undisclosed. This paper aims to bridge this gap by providing a theoretical framework that elucidates the mechanisms underlying these empirical strategies.

2.2 Scaling relationship between batch size and learning rate

Given the interdependence of learning rate and batch size, a critical challenge lies in determining the optimal scaling strategy for the learning rate as batch size changes.

Krizhevsky (2014) initially proposed the square-root scaling rule for SGD, suggesting that the learning rate should scale by $\sqrt{k}$ when the batch size scales by k. However, this heuristic was subsequently challenged. Goyal et al. (2017) demonstrated that for SGD, the learning rate should instead scale linearly with batch size (i.e., by a factor of k). Smith et al. (2020) corroborated this linear scaling rule, emphasizing its validity specifically within the small-batch regime. Furthermore, while establishing the Critical Batch Size framework, McCandlish et al. (2018) formalized the relationship between optimal learning rate and batch size as follows:

$$\eta_{opt} = \frac{\eta_{max}}{1 + B_{noise}/B}, \quad (3)$$

where B_{noise} denotes the gradient noise scale, B presents the batch size, and η_{max} is a constant. Crucially, in the small-batch regime($B \ll B_{noise}$), the learning rate scales approximately linearly with the batch size.

The widespread adoption of the Adam optimizer (Kingma, 2014) has fundamentally altered the relationship between batch size and learning rate. You et al. (2019) empirically observed during BERT training that scaling the learning rate by the square root of the batch size ($\eta \propto \sqrt{B}$) yields superior performance. This heuristic was formalized by Liu et al. (2019), who demonstrated that under Adam, gradient noise variance scales with η^2/B; thus,

maintaining constant variance requires square-root scaling. Malladi et al. (2022) further substantiated this relationship theoretically via a stochastic differential equation (SDE) approximation of Adam. Recently, however, Li et al. (2024b) challenged this convention. By re-examining the optimization dynamics of Adam, they proposed a revised scaling law for the optimal learning rate:

$$\eta_{opt} = \frac{\eta_{max}}{\frac{1}{2}(\sqrt{\frac{B_{noise}}{B}} + \sqrt{\frac{B}{B_{noise}}})}. \quad (4)$$

In the small-batch regime ($B \ll B_{noise}$), the optimal learning rate scales approximately linearly with $\sqrt{B}$. However, once the batch size surpasses the gradient noise B_{noise}, the optimal learning rate begins to decay.

Summary and Connection Prior work falls into two main paradigms: empirically fitting optimal batch size scaling laws (often theory-light) or theoretically deriving learning rate adjustments (often impractical for large-scale training). We address the limitations of both approaches by:

1. Diverging from prior studies that rely exclusively on empirical fitting to determine batch size scaling laws, our work provides a formal theoretical characterization of pre-training dynamics under the WSD schedule. By deriving a novel $E(S)$ relationship for the Stable phase, we establish a robust framework grounded in first principles that elucidates these underlying dynamics;

2. Distinguished from purely theoretical studies on hyperparameters like learning rate and batch size, our work translates theoretical insights into a practical batch size schedule tailored for WSD large-scale pre-training. Validated across diverse scenarios, our approach demonstrates significant practical utility and robustness.

3 Approach

3.1 Rethinking the Critical Batch Size

To characterize the optimal trade-off between data consumption E and optimization steps S, McCandlish et al. (2018) introduced the concept of *Critical Batch Size*. This framework is grounded in the empirical observation that, when training a model to a fixed performance target, E and S satisfy the relationship in Eq.1. Here, S_{min} represents the minimum steps required to achieve the target loss, while E_{min} denotes the minimum data volume needed. The Critical Batch Size is formally defined as the ratio $B_{crit} = E_{min}/S_{min}$.

Existing research on Critical Batch Size, including the seminal work by McCandlish et al. (2018), has predominantly focused on the Cosine learning rate schedule. Crucially, however, the behavior of Critical Batch Size under the Warmup-Stable-Decay (WSD) learning rate schedule (Hu et al., 2024) remains significantly underexplored. This represents a critical gap, particularly given the widespread adoption of WSD in modern large-scale pre-training tasks, such as those by DeepSeek (Bi et al., 2024), Kimi (Team et al., 2025), and Qwen (Yang et al., 2025).

To analyze this discrepancy, we first reformulate Eq.1 to examine the data consumption required to reach a specific target loss across varying batch sizes. The reformulated equation is given by:

$$E = E_{min} + BS_{min}. \quad (5)$$

This equation indicates that achieving a fixed target loss with a larger batch size typically necessitates greater data consumption. Specifically, assuming a model is trained to the same loss level using batch sizes B_1 and B_2 (where $B_1 < B_2$), the corresponding data consumption E_1 and E_2 must satisfy the inequality $E_1 < E_2$.

However, under the WSD learning rate schedule, we observe that the training curves $L(D)$ for varying batch sizes intersect during practice. Specifically, while the relationship $E_1 < E_2$ holds at relatively higher target losses, this relationship inverts once the target loss drops below a specific threshold, resulting in $E_1 > E_2$ (as illustrated in Figure 1). This observation stands in direct contradiction to the monotonicity implied by the standard $E(S)$ formula. Consequently, these experimental results demonstrate that the fundamental principles of Critical Batch Size do not hold during the Stable phase of the WSD paradigm.

3.2 A New E(S) Formula Adapted to Large-scale Pre-training

Experimental analysis reveals that the prerequisites for the existing Critical Batch Size theory are violated during the Stable phase of the WSD learning rate schedule. Fundamentally, the standard $E(S)$ relationship renders itself inapplicable in this regime. To address this, we draw upon the analytical methodology of McCandlish et al. (2018) regarding SGD optimization dynamics to construct a novel $E(S)$ theoretical framework tailored specifically for the WSD learning rate schedule. This framework models the data consumption

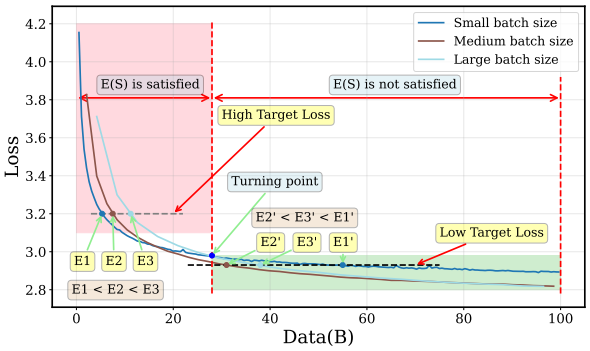

Figure 1: Loss curves for models trained with different batch sizes (Stable phase under WSD schedule). The red region denotes the regime where the $E(S)$ formula and Critical Batch Size theory remain valid. In the green region, the $E(S)$ relationship no longer holds, leading to a failure of the Critical Batch Size framework. Post-intersection, the partial ordering of data consumption among the various batch sizes is inverted.

E required to reach a target loss as a function of optimization steps S, meticulously decomposing the evolution process into three distinct stages:
Initial stage:E fluctuates inversely with $S - S_{min}$(inverse linear stage in Figure 2);
Transition stage: E is expressed as a quadratic function of S(transition stage in Figure 2);
Asymptotic stage: E increases linearly with S(linear stage in Figure 2).

The corresponding piecewise function expression is as follows:

$$E(S) = \begin{cases} B_{-1}/(S - S_{min}) + B_0, S_{min} < S < S_1, \\ C(S - S_{opt})^2 + E_{min}, S_1 < S < S_2, \\ A_1 S + A_0, S > S_2. \end{cases} \tag{6}$$

For a detailed derivation of this formula, please refer to the Appendix A.2.

3.3 Fitting of the New E(S) formula

From the piecewise form of the function $E(S)$, we obtain 10 parameters to be fitted. First, we impose constraints on these parameters. By requiring the $E(S)$ curve to be continuous, smooth and differentiable, we derive the following equality constraints:

$$\frac{B_{-1}}{S_1 - S_{min}} + B_0 = C(S_1 - S_{opt})^2 + E_{min}, \tag{7}$$

$$C(S_2 - S_{opt})^2 + E_{min} = A_1 S_2 + A_0, \tag{8}$$

$$-\frac{B_{-1}}{(S_1 - S_{min})^2} = 2C(S_1 - S_{opt}), \tag{9}$$

$$2C(S_2 - S_{opt}) = A_1. \tag{10}$$

Meanwhile, the following inequality constraints are given:

$$S_{min} < S_1 < S_{opt} < S_2. \tag{11}$$

Thereby, we establish the parameter search space for fitting. The $E(S)$ curve is then fitted by minimizing the Huber loss function (Huber, 1992). Assume the dataset for fitting is $\{(S_i, E_i)\}_{i=1}^n$ and the parameters to be fitted are denoted by θ. The fitting process can be described by the following equation:

$$\theta^* = arg\min_{\theta} \sum_{i=1}^{n} Huber_\delta(E_i, E(S_i, \theta)), \tag{12}$$

here, the Huber loss is defined as following:

$$Huber_\delta(x, y) = \begin{cases} \frac{1}{2}(x - y)^2, |x - y| \le \delta, \\ \delta|x - y| - \frac{1}{2}\delta^2, |x - y| > \delta. \end{cases} \tag{13}$$

In order to improve fitting efficiency, we utilize scaling laws to expedite the generation of Loss-Step pairs. According to Luo et al., in the regime of a constant learning rate, the loss is governed by the following scaling relationship with respect to steps:

$$L(S) = L_0 + AS^{-\alpha}. \tag{14}$$

Given a target loss, the above formula enables straightforward calculation of the steps needed for the model to descend to that loss. This yields data points for fitting $E(S)$ at the given loss.

Figure 2 presents our new $E(S)$ fitting results for the 1B model. As evident from the plot, the derived $E(S)$ exhibits excellent fitting performance, further substantiating the correctness of our analysis of model training dynamics in the Stable phase.

Under stable learning rate schedule, the Critical Batch Size no longer holds. Instead, it is replaced by two metrics:B_{min} and B_{opt}, as given by the following formulas:

$$B_{min} = A_1, B_{opt} = \frac{E_{min}}{S_{opt}}. \tag{15}$$

Physically, B_{min} and B_{opt} quantify critical batch size thresholds: B_{min} is the minimum batch size needed to reach a target loss, and B_{opt} is the batch size that yields minimum data consumption. Geometrically (see Appendix A.3 for the full $E(S)$ plot), B_{min} equals the slope of the curve's asymptote, while B_{opt} equals the slope from the origin to the curve's minimum. As shown in Figure 3, both

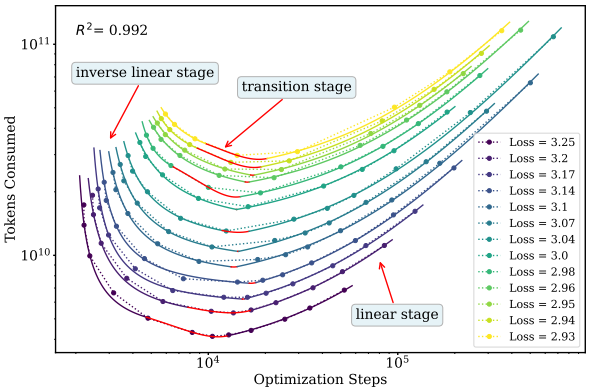

Figure 2: Fitting results of $E(S)$ for 1B model. We select the target loss interval as [2.93, 3.25] and perform fitting on the $E(S)$ curves for target losses within this interval.

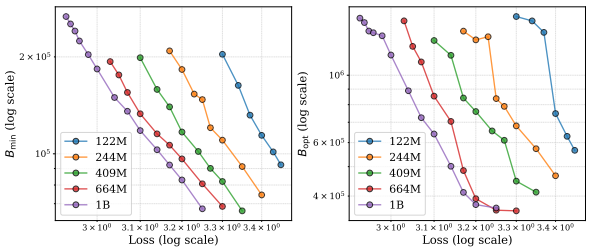

Figure 3: The variation of B_{min} and B_{opt} with respect to loss across different model sizes.

metrics scale monotonically with decreasing target loss (increasing data volume). This scaling behavior provides the empirical basis for the dynamic batch size scheduling strategy proposed later.

3.4 A New Batch Size Schedule

Within the proposed $E(S)$ theoretical framework, several derived metrics associated with batch size can be established. Specifically, the physical interpretation of B_{opt} is as follows: in the context of constant batch size training, it represents the value that maximizes data efficiency for reaching a specific target loss. Equivalently, it serves as the optimal solution for minimizing loss given a fixed data budget. Nevertheless, employing a static batch size throughout the entire training process is rarely globally optimal in practice. Synthesizing this insight with the experimental observation that B_{opt} increases monotonically as training progresses, we can derive a batch size scheduling scheme better suited for large-scale pre-training. This implies abandoning static batch sizes in favor of a strategy that dynamically expands the batch size over time, thereby achieving superior training performance.

Theorem 1 Assume the model size is fixed, and let the loss be expressed as $L(N, B, D)$, which depends on model size N, batch size B and data volume D. The two optimization problems below are equivalent:

Problem 1: For a fixed training data budget, which constant batch size minimizes the model's loss?

Problem 2: For a prescribed target loss, which constant batch size minimizes the data consumption by the model?

Theorem 1 establishes that modeling the evolution of B_{opt} with respect to data volume is mathematically equivalent to characterizing its relationship with the loss function. While Figure 3 explicitly illustrates the trajectory of B_{opt} as the loss decreases, Theorem 1 implies that this curve simultaneously reveals the scaling law of B_{opt} with respect to data consumption. Given that cumulative data volume serves as a more intuitive metric for training progress than the loss value, we adopt data consumption as the reference benchmark for the dynamic batch size scheduling strategy.

Algorithm 1 Batch size scheduling strategy

Input: model size N, learning rate η, the optimal batch size curve $f(N, D)$ under this learning rate, batch size switching interval $D_{interval}$, momentum list $\{\alpha_i\}_{i=1}^{n}$, switching times n.
Initialization: $B_{global,0} = 0, i = 1$
repeat
  $B_{last} = f(N, (i-1)D_{interval})$
  $B_{new} = f(N, iD_{inerval})$
  $B_{global,i} = B_{global,i-1} + (1 + \alpha_i)(B_{new} - B_{last})$
  $i \leftarrow i + 1$
until $i > n$
Output: $B_{global,1}, ..., B_{global,n}$

In the subsequent large-scale pre-training experiments, we validate across diverse scenarios that this batch size scheduling strategy effectively enhances model performance.

4 Experiments

4.1 Dataset

Our experiments utilize the InternLM2 corpus (Cai et al., 2024), categorized into general text, code, and long-context data. The text segment aggregates web pages, academic literature, books, and patents, while the code portion is curated from GitHub and public repositories across languages such as C/C++,

Java, and Python. We process the long-context subset via a three-stage pipeline—comprising length selection, statistical filtering, and perplexity-based pruning—to guarantee high-quality long-range dependencies.

4.2 Model Architectures

For the $E(S)$ curve fitting experiments, we adopt the InternLM2 architecture. Building upon the LLaMA (Touvron et al., 2023) foundation, InternLM2 fuses the query (W_q), key (W_k), and value (W_v) matrices into a consolidated, interleaved layout per head. Furthermore, the architecture incorporates Grouped-Query Attention (GQA) (Ainslie et al., 2023) to enhance efficiency.

For the batch size scheduling experiments, we utilize the Qwen3 model series (Yang et al., 2025), comprising both dense and Mixture-of-Experts (MoE) variants. The Qwen3 Dense model refines the Qwen2 architecture (Team et al., 2024) by eliminating QKV-bias and incorporating QK-Norm. Meanwhile, the Qwen3 MoE model extends Qwen2.5-MoE by discarding shared experts and adopting a global-batch load balancing loss (Qiu et al., 2025).

4.3 Training Settings

4.3.1 Fitting of E(S)

To empirically fit the $E(S)$ curve, we trained five InternLM2 model variants with parameter counts ranging from 122M to 1B, utilizing batch sizes spanning 64K to 7.5M. Optimization was performed using AdamW with a fixed learning rate of 6×10^{-4} and a 1,000-step warmup. The total training volume varied between 50B and 120B tokens depending on the configuration.

4.3.2 Batch size Scheduling

Baseline We conduct our approach using the Qwen3 MoE and Qwen3 Dense architectures. Given the widespread adoption of WSD learning rate schedule in modern large-scale pretraining (Team et al., 2025; Liu et al., 2024; Hu et al., 2024), and acknowledging that the stable phase consumes the majority of the training budget, we focus our experiments on the constant learning rate regime. Specifically, we set the learning rate to 3.2×10^{-4} for Qwen3 MoE and 1.75×10^{-4} for Qwen3 Dense. For both architectures, we standardize the training configuration with 1,000 warmup steps, a global batch size of 4M, the AdamW optimizer, and a weight decay of 0.1.

Controlled experiments For comparison, we mirrored the baseline setup while introducing a dynamic batch size strategy. The global batch size was adjusted at 125B-token intervals according to the sequence $\{2\text{M}, 4\text{M}, 5\text{M}, 6\text{M}\}$, achieved by scaling the micro-batch size while keeping other hyperparameters constant.

4.4 Evaluation

4.4.1 Benchmarks

We evaluate the downstream capabilities of our models using the MMLU benchmark (Hendrycks et al., 2020) and the CMMLU benchmark (Li et al., 2024a).

4.4.2 Evaluation Tools

For our evaluation, we employ OpenCompass (Contributors, 2023b) to assess model performance on both the MMLU and CMMLU benchmarks. During evaluation, OpenCompass utilizes LMDeploy (Contributors, 2023a) to accelerate inference execution.

4.5 Results

Figure 4 presents the smoothed training loss trajectory for the Qwen3 MoE model. As illustrated, the curve for the dynamic batch size scheduling strategy consistently lies below that of the fixed-batch baseline, indicating superior convergence. Figure 5 further contrasts performance on the MMLU and CMMLU benchmarks, where the dynamic strategy maintains a consistent advantage. Mirroring these findings, Figures 6 and 7 display the training loss and downstream results for the Qwen3 Dense model, which exhibit identical trends. Collectively, these experiments validate the effectiveness of our dynamic batch size scheduling strategy and corroborate our theoretical analysis of optimization dynamics under WSD learning rate schedule.

5 Ablations

5.1 The Effect of learning rate

Cosine learning rate schedule We further validate our strategy's adaptability using a Cosine scheduler on the Qwen3 MoE model (LR: $0 \rightarrow 1.7 \times 10^{-3} \rightarrow 3.2 \times 10^{-4}$ over 500B tokens). Compared to a fixed 4M batch size baseline, our dynamic schedule—scaling from 2M to 6M in 125B-token increments—yields superior training loss and downstream results (Figure 8). This success aligns with the Critical Batch Size theory (McCandlish et al.,

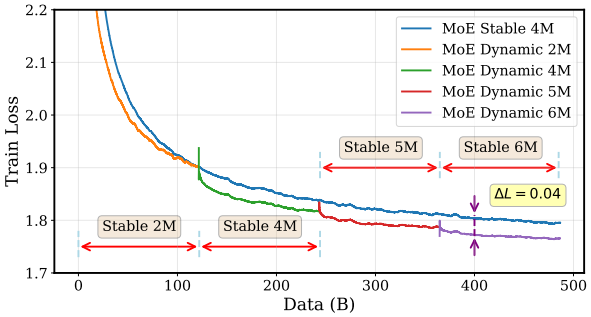

Figure 4: Training loss curves for Qwen3 MoE using fixed and dynamic batch size strategies under a constant learning rate schedule.

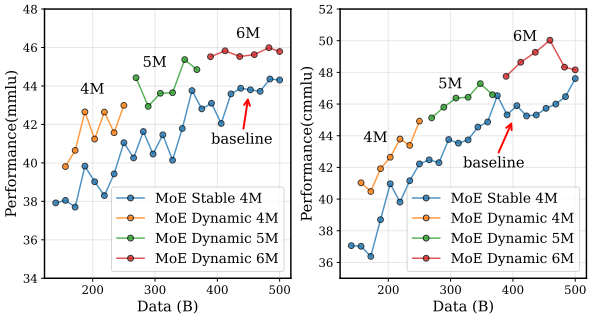

Figure 5: Comparison of downstream benchmark results for Qwen3 MoE under fixed vs. dynamic batch size scheduling at a constant learning rate.

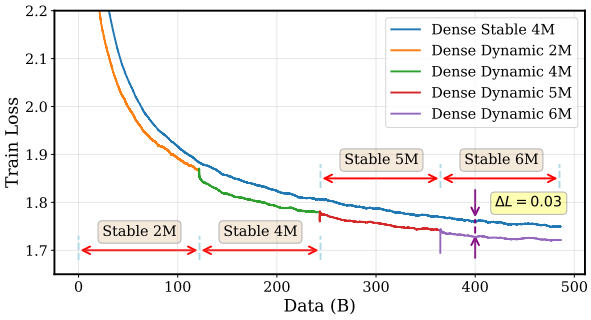

Figure 6: Training loss curves for Qwen3 Dense model under fixed and dynamic batch size strategies at a constant learning rate。

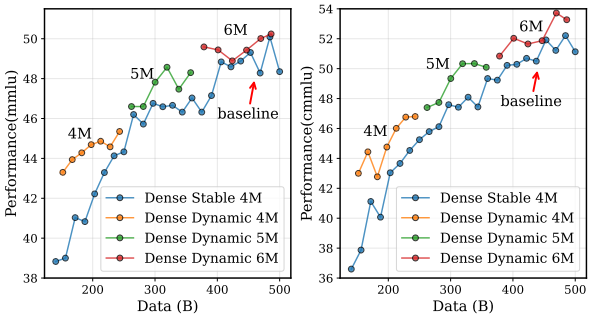

Figure 7: Comparison of downstream benchmark results for Qwen3 Dense under fixed vs. dynamic batch size scheduling at a constant learning rate.

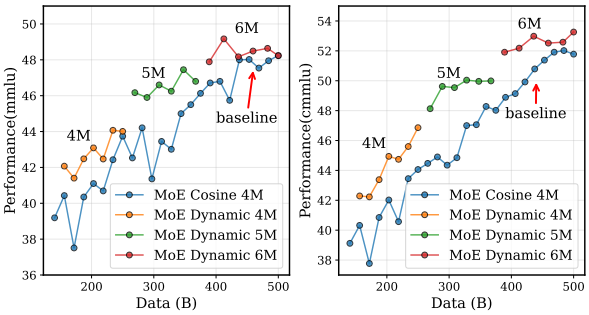

Figure 8: Comparison of downstream benchmark results for Qwen3 MoE under fixed and dynamic batch size scheduling with cosine learning rate schedule.

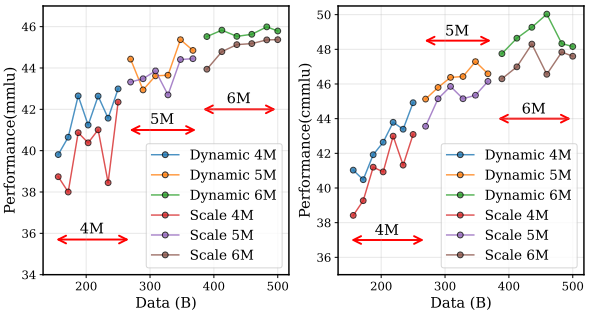

Figure 9: Comparative downstream performance of dynamic batch size scheduling strategies: Constant learning rate versus learning rate scaling regimes.

2018): as gradient noise accumulates during training, expanding the batch size becomes essential to counteract noise-induced instability, thereby facilitating convergence to a deeper loss minimum.

Increase the learning rate as batch size increases We challenge the convention of scaling the learning rate alongside batch size increases. In an ablation study using the Qwen3 MoE model, we compared square-root scaling (LR $\propto \sqrt{B}$) against a constant learning rate while progressively increasing the batch size from 2M to 6M. Empirical results in Figure 9 demonstrate that scaling the learning rate offers no performance improvement. This is because higher learning rates exacerbate gradient noise, effectively neutralizing the noise-suppression benefits of larger batch sizes and rendering the scaling strategy counterproductive.

5.2 The Effect of sequence length

An alternative to micro-batch scaling is the extension of sequence length (*seqlen*) to achieve larger global batch sizes. We evaluated this approach on Qwen3 MoE, comparing a 4K *seqlen* baseline against a strategy that shifted *seqlen* to 5K and 6K at specific intervals (250B and 375B tokens).

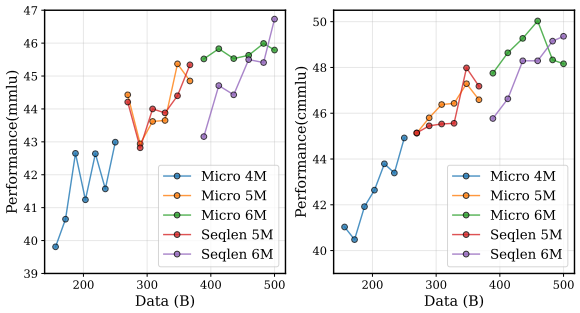

Figure 10: Comparative downstream performance of dynamic batch size scheduling implemented through micro-batch expansion and sequence length extension.

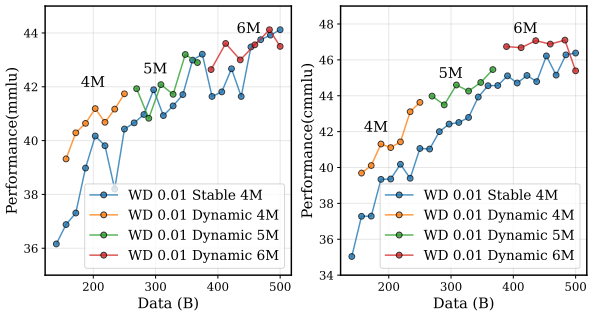

Figure 11: Comparative downstream performance of fixed and dynamic batch size scheduling under different weight decay settings.

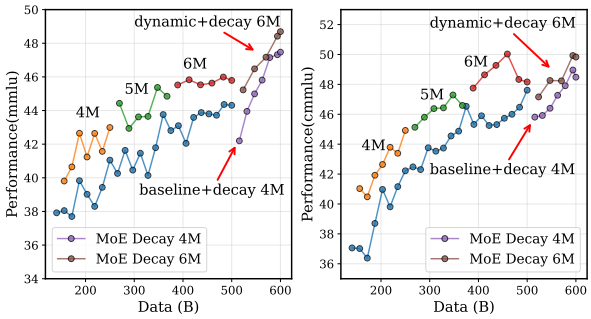

Figure 12: Comparison of downstream benchmark results for fixed and dynamic batch size strategies in the annealing phase.

While both methods reach equivalent batch sizes, *seqlen* extension perturbs the training distribution by altering the sample structure. Empirical results in Figure 10 reveal an immediate performance drop upon increasing *seqlen* to 6K, suggesting a non-trivial adaptation period is necessary to reconcile the distribution shift. Although the model eventually recovers, the risk of a learning preference shift toward long-context sequences makes this approach less desirable for standard large-scale pre-training.

5.3 The Effect of weight decay

We further investigate the sensitivity of our strategy to weight decay. By reducing the coefficient to 0.01 on the Qwen3 MoE model, we observe in Figure 11 that the initial advantage of the dynamic strategy diminishes and nearly vanishes as training progresses. Furthermore, a cross-comparison with the main experiments (Figure 5, WD=0.1) confirms that the 0.01 setting results in significantly inferior baselines. These findings indicate that the effectiveness of dynamic batch sizing is coupled with regularization strength; specifically, the full benefits of the strategy are contingent upon an optimal weight decay setting.

5.4 The Effect of Continued Training

To validate compatibility with modern pretraining protocols, we extended our evaluation to the decay phase of the WSD schedule, characterized by high-quality data annealing. Using the pre-trained MoE model, we conducted a comparative run over 100B tokens with a linear learning rate decay to 10%. The baseline maintained a 4M batch size, whereas the dynamic strategy retained its peak 6M batch size. Figure 12 demonstrates that the performance advantage of the dynamic strategy is sustained throughout this phase. This confirms the robustness of our approach against data distribution shifts, validating its effectiveness in standard large-scale training pipelines.

6 Conclusion

This work first elucidates the limitations of the seminal Critical Batch Size theory (McCandlish et al., 2018), demonstrating its inapplicability to the Warmup-Stable-Decay (WSD) scheduler prevalent in modern large-scale pre-training. To address this gap, we propose a novel $E(S)$ formulation tailored specifically for the WSD paradigm. Within this framework, we identify two pivotal metrics: B_{min}, the minimum batch size threshold required to reach a target loss, and B_{opt}, the optimal batch size for maximizing data efficiency. We observe that both B_{min} and B_{opt} increase monotonically as training loss decreases. Motivated by this finding, we introduce a dynamic batch size adjustment strategy and validate its effectiveness across multiple large-scale pre-training scenarios.

Limitations

While our batch size adjustment paradigm proves effective for large-scale pre-training, it is circumscribed by certain limitations. (1) Computational costs restricted our $E(S)$ curve fitting to a specific learning rate (6×10^{-4}), leaving its behavior under other learning rates unexplored. (2) Although empirically successful, the dynamic strategy has not yet been formalized with a theoretical proof. (3) The training instabilities associated with sequence length (*seqlen*) switching remain unaddressed, limiting the overall flexibility of the scheduling strategy. We aim to explore these aspects in subsequent studies.

Use of AI Assistants

We primarily use AI assistants to improve and enrich our writing.

References

Joshua Ainslie, James Lee-Thorp, Michiel De Jong, Yury Zemlyanskiy, Federico Lebrón, and Sumit Sanghai. 2023. Gqa: Training generalized multi-query transformer models from multi-head checkpoints. *arXiv preprint arXiv:2305.13245*.

Xiao Bi, Deli Chen, Guanting Chen, Shanhuang Chen, Damai Dai, Chengqi Deng, Honghui Ding, Kai Dong, Qiushi Du, Zhe Fu, and 1 others. 2024. Deepseek llm: Scaling open-source language models with longtermism. *CoRR*.

Zheng Cai, Maosong Cao, Haojiong Chen, Kai Chen, Keyu Chen, Xin Chen, Xun Chen, Zehui Chen, Zhi Chen, Pei Chu, and 1 others. 2024. Internlm2 technical report. *arXiv preprint arXiv:2403.17297*.

LMDeploy Contributors. 2023a. Lmdeploy: A toolkit for compressing, deploying, and serving llm. https://github.com/InternLM/lmdeploy.

OpenCompass Contributors. 2023b. Opencompass: A universal evaluation platform for foundation models. https://github.com/open-compass/opencompass.

Priya Goyal, Piotr Dollár, Ross Girshick, Pieter Noordhuis, Lukasz Wesolowski, Aapo Kyrola, Andrew Tulloch, Yangqing Jia, and Kaiming He. 2017. Accurate, large minibatch sgd: Training imagenet in 1 hour. *arXiv preprint arXiv:1706.02677*.

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

Shengding Hu, Yuge Tu, Xu Han, Chaoqun He, Ganqu Cui, Xiang Long, Zhi Zheng, Yewei Fang, Yuxiang Huang, Weilin Zhao, and 1 others. 2024. Minicpm: Unveiling the potential of small language models with scalable training strategies. *arXiv preprint arXiv:2404.06395*.

Peter J Huber. 1992. Robust estimation of a location parameter. In *Breakthroughs in statistics: Methodology and distribution*, pages 492–518. Springer.

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

Nitish Shirish Keskar, Dheevatsa Mudigere, Jorge Nocedal, Mikhail Smelyanskiy, and Ping Tak Peter Tang. 2017. On large-batch training for deep learning: Generalization gap and sharp minima. In *International Conference on Learning Representations*.

Diederik P Kingma. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Alex Krizhevsky. 2014. One weird trick for parallelizing convolutional neural networks. *arXiv preprint arXiv:1404.5997*.

Haonan Li, Yixuan Zhang, Fajri Koto, Yifei Yang, Hai Zhao, Yeyun Gong, Nan Duan, and Timothy Baldwin. 2024a. Cmmlu: Measuring massive multitask language understanding in chinese. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 11260–11285.

Houyi Li, Wenzhen Zheng, Jingcheng Hu, Qiufeng Wang, Hanshan Zhang, Zili Wang, Shijie Xuyang, Yuantao Fan, Shuigeng Zhou, Xiangyu Zhang, and 1 others. 2025. Predictable scale: Part i–optimal hyperparameter scaling law in large language model pretraining. *arXiv e-prints*, pages arXiv–2503.

Shuaipeng Li, Penghao Zhao, Hailin Zhang, Xingwu Sun, Hao Wu, Dian Jiao, Weiyan Wang, Chengjun Liu, Zheng Fang, Jinbao Xue, and 1 others. 2024b. Surge phenomenon in optimal learning rate and batch size scaling. *Advances in Neural Information Processing Systems*, 37:132722–132746.

Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, and 1 others. 2024. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*.

Liyuan Liu, Haoming Jiang, Pengcheng He, Weizhu Chen, Xiaodong Liu, Jianfeng Gao, and Jiawei Han. 2019. On the variance of the adaptive learning rate and beyond. *arXiv preprint arXiv:1908.03265*.

Kairong Luo, Haodong Wen, Shengding Hu, Zhenbo Sun, Zhiyuan Liu, Maosong Sun, Kaifeng Lyu, and Wenguang Chen. A multi-power law for loss curve

prediction across learning rate schedules. In *The Thirteenth International Conference on Learning Representations*.

Sadhika Malladi, Kaifeng Lyu, Abhishek Panigrahi, and Sanjeev Arora. 2022. On the sdes and scaling rules for adaptive gradient algorithms. *Advances in Neural Information Processing Systems*, 35:7697–7711.

Sam McCandlish, Jared Kaplan, Dario Amodei, and OpenAI Dota Team. 2018. An empirical model of large-batch training. *arXiv preprint arXiv:1812.06162*.

MiniMax, Aonian Li, Bangwei Gong, Bo Yang, Boji Shan, Chang Liu, Cheng Zhu, Chunhao Zhang, Congchao Guo, Da Chen, Dong Li, Enwei Jiao, Gengxin Li, Guojun Zhang, Haohai Sun, Houze Dong, Jiadai Zhu, Jiaqi Zhuang, Jiayuan Song, and 71 others. 2025. Minimax-01: Scaling foundation models with lightning attention. *Preprint*, arXiv:2501.08313.

Zihan Qiu, Zeyu Huang, Bo Zheng, Kaiyue Wen, Zekun Wang, Rui Men, Ivan Titov, Dayiheng Liu, Jingren Zhou, and Junyang Lin. 2025. Demons in the detail: On implementing load balancing loss for training specialized mixture-of-expert models. *arXiv preprint arXiv:2501.11873*.

Xian Shuai, Yiding Wang, Yimeng Wu, Xin Jiang, and Xiaozhe Ren. 2024. Scaling law for language models training considering batch size. *arXiv preprint arXiv:2412.01505*.

Samuel Smith, Erich Elsen, and Soham De. 2020. On the generalization benefit of noise in stochastic gradient descent. In *International Conference on Machine Learning*, pages 9058–9067. PMLR.

Kimi Team, Yifan Bai, Yiping Bao, Guanduo Chen, Jiahao Chen, Ningxin Chen, Ruijue Chen, Yanru Chen, Yuankun Chen, Yutian Chen, and 1 others. 2025. Kimi k2: Open agentic intelligence. *arXiv preprint arXiv:2507.20534*.

Qwen Team and 1 others. 2024. Qwen2 technical report. *arXiv preprint arXiv:2407.10671*, 2(3).

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, and 1 others. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, and 1 others. 2025. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.

Yang You, Jing Li, Sashank Reddi, Jonathan Hseu, Sanjiv Kumar, Srinadh Bhojanapalli, Xiaodan Song, James Demmel, Kurt Keutzer, and Cho-Jui Hsieh. 2019. Large batch optimization for deep learning: Training bert in 76 minutes. *arXiv preprint arXiv:1904.00962*.

A Appendix

A.1 Proof of the theorem

Theorem 1 Assume the model size is fixed, and let the loss be expressed as $L(N, B, D)$, which depends on model size N, batch size B and data volume D. The two optimization problems below are equivalent:

Problem 1: For a fixed training data budget, which constant batch size minimizes the model's loss?

Problem 2: For a prescribed target loss, which constant batch size minimizes the data consumption by the model?

Proof We first express the two problems in the framework of optimization theory.

Problem 1

$$\begin{aligned} &\min_B L(N, B, D) \\ &s.t. D = D_0. \end{aligned} \tag{16}$$

Problem 2

$$\begin{aligned} &\min_B D \\ &s.t. L(N, B, D) = L_0. \end{aligned} \tag{17}$$

Let D_0 be fixed, and define:

$$L_0 = \min_B L(N, B, D_0) = L(N, B^*, D_0). \tag{18}$$

Taking L_0 as the target loss in Problem 2, we prove that the solution to Problem 2 is also B^*. To do so, it suffices to show that training with batch size B^* consumes less data than any other batch size.

$$\begin{aligned} \forall B, L(N, B, D) = L_0 &= L(N, B^*, D_0) \\ &\leq L(N, B, D_0). \end{aligned} \tag{19}$$

Since $L(N, B, D)$ is monotonically decreasing in D, it necessarily follows that:

$$D_0 \leq D. \tag{20}$$

Thus, the solution to Problem 2 is also B^*. Problem 1 and Problem 2 are equivalent. Q.E.D.

A.2 Analysis of the Model Optimization Process in the Stable Phase

A.2.1 Analysis of Training Dynamics

Following the analysis of the model training process under SGD optimization by McCandlish et al., we re-analyzed the training dynamics under

the condition of stable learning rate schedule. Using the Taylor expansion formula, we performed a quadratic approximation of the loss curve, yielding:

$$L(\theta - \epsilon V) \approx L(\theta) - \epsilon G^T V + \frac{1}{2}\epsilon^2 V^T H V, \quad (21)$$

where θ is model parameter, G is gradient, H is matrix of Hessian, V is descending direction, ϵ is learning rate. Let B denotes the batch size in SGD. The stochastic gradient estimate at each step takes the following form:

$$G_{est}(\theta) = \frac{1}{B}\sum_{i=1}^{B} G_i. \quad (22)$$

We assume that G_i are i.i.d, $G_i \sim N(G, \Sigma)$, then we have:

$$E[G_{est}(\theta)] = G, Cov(G_{est}(\theta)) = \frac{\Sigma}{B}. \quad (23)$$

We substitute $V = G_{est}(\theta)$ into formula (21). Taking the expectation of both sides, we obtain:

$$\begin{aligned} E[L(\theta - \epsilon G_{est})] &= L(\theta) - \epsilon|G|^2 \\ &+ \frac{1}{2}\epsilon^2(G^T H G + \frac{tr(H\Sigma)}{B}). \end{aligned} \quad (24)$$

Hence,

$$\begin{aligned} E[\Delta L] &= E[L(\theta) - L(\theta - \epsilon G_{est})] \\ &= \epsilon|G|^2 - \frac{1}{2}\epsilon^2(G^T H G + \frac{tr(H\Sigma)}{B}). \end{aligned} \quad (25)$$

For analytical convenience, we approximate the Hessian matrix as the identity matrix.Then, the formula (25) can be approximated as:

$$E[\Delta L] \approx \epsilon|G|^2 - \frac{1}{2}\epsilon^2(|G|^2 + \frac{tr(\Sigma)}{B}). \quad (26)$$

Using formula (26) as our foundation, we investigate how the per-step loss reduction varies with different batch sizes. For full-batch gradient descent, we have:

$$E[\Delta L]_{full-batch} = \epsilon|G|^2 - \frac{1}{2}\epsilon^2|G|^2. \quad (27)$$

To achieve the same amount of loss reduction as one full-batch gradient descent step, the required number of steps for batch size B is given by:

$$\begin{aligned} \delta S &= \frac{E[\Delta L]_{full-batch}}{E[\Delta L]_B} \\ &= \frac{1 - \frac{1}{2}\epsilon}{1 - \frac{1}{2}\epsilon(1 + \frac{B_{noise}}{B})}, \end{aligned} \quad (28)$$

where $B_{noise} = tr(H)/|G|^2$ denotes the gradient noise scale of the model. Since the learning rate is very small in practice, we can approximate formula (28) as:

$$\delta S \approx \frac{1}{1 - \frac{1}{2}\epsilon\frac{B_{noise}}{B}}. \quad (29)$$

The volume of training data processed by the model thus far is:

$$\delta E = B\delta S \approx \frac{B}{1 - \frac{1}{2}\epsilon\frac{B_{noise}}{B}}. \quad (30)$$

Given that the model reaches loss L after S_{min} steps under full-batch gradient descent, the required step S and data volume E to achieve the same loss with batch size B are respectively:

$$S = \int_0^{S_{min}} \delta S ds \approx \int_0^{S_{min}} \frac{1}{1 - \frac{1}{2}\epsilon\frac{B_{noise}(s)}{B}} ds, \quad (31)$$

$$E = \int_0^{S_{min}} \delta E ds \approx \int_0^{S_{min}} \frac{B}{1 - \frac{1}{2}\epsilon\frac{B_{noise}(s)}{B}} ds. \quad (32)$$

A.2.2 Asymptotic Analysis

We analyze the asymptotic behavior of $E(S)$ curve at both ends:as $S \to S_{min}$ and as $S \to +\infty$.

1.$S \to S_{min}$

When S approaches S_{min}, it corresponds to the case where the batch size tends to infinity. We analyze the scaling relationship captured by the product $(S - S_{min})E$:

$$\begin{aligned} (S - S_{min})E = &\int_0^{S_{min}} \frac{B\frac{1}{2}\epsilon B_{noise}(s)}{B - \frac{1}{2}\epsilon B_{noise}(s)} ds * \\ &\int_0^{S_{min}} \frac{B}{B - \frac{1}{2}\epsilon B_{noise}(s)} ds. \end{aligned} \quad (33)$$

Letting $B \to +\infty$, we have:

$$\lim_{B\to+\infty}(S - S_{min})E = S_{min}\int_0^{s_{min}} \frac{1}{2}\epsilon B_{noise}(s) ds. \quad (34)$$

Using an infinite series expansion, we can express $E(S)$ as:

$$E(S) = \frac{B_{-1}}{S - S_{min}} + \sum_{i=0}^{\infty} B_i (S - S_{min})^i. \quad (35)$$

By truncating the higher-order terms, the above formula is approximated as:

$$E(S) \approx \frac{B_{-1}}{S - S_{min}} + B_0. \quad (36)$$

2.$S \to +\infty$

We note that, since the learning rate is constant, the necessary and sufficient condition for the loss curve to continue decreasing under batch size B is:

$$E[\Delta L]_B > 0 \Leftrightarrow 1 - \frac{1}{2}\epsilon\frac{B_{noise}}{B} > 0 \\ \Leftrightarrow B > \frac{1}{2}\epsilon B_{noise}. \tag{37}$$

In other words, to sustain loss reduction, the batch size must be larger than a dynamic lower bound that scales with the instantaneous gradient noise. Thus, when loss stagnation occurs, the batch size has effectively hit this bound, signaling convergence. Formally, we have:

$$\lim_{S \to +\infty} \frac{E}{S} = A_1. \tag{38}$$

Similarly, formula (38) can also be expressed in the form of an infinite series:

$$E(S) = A_1 S + \sum_{i=-\infty}^{0} A_i S^i. \tag{39}$$

By truncating the higher-order terms, the above formula is approximated as:

$$E(S) \approx A_1 S + A_0. \tag{40}$$

A.2.3 Reconstruction of E(S)

Through above asymptotic analysis of $E(S)$ curve, we have understood the forms that $E(S)$ takes when S tends to S_{min} and to infinity, respectively. What remains an open question is the variation of $E(S)$ when S falls within the intermediate interval. Since $E(S) \to +\infty$ as $S \to S_{min}$ and as $S \to +\infty$, Rolle's Theorem implies the existence of a point $S^* \in (S_{min}, +\infty)$ such that $E'(S^*) = 0$. That is, $E(S)$ has a minimum point, at which the model reaches data optimality - consuming the least amount of data.

Lacking a tractable closed-form expression for $E(S)$ in the intermediate regime, we approximate the curve with a piecewise function. The specific expression for $E(S)$ can be found in formula (6).Meanwhile, we require that $E(S)$ be continuous, smooth, and differentiable, thereby leading to equality constraint conditions, from formula (7) to formula (10). simultaneously, we require that $E(S)$ has an extreme point the quadratic function stage, thus an inequality constraint is imposed as given in formula (11).

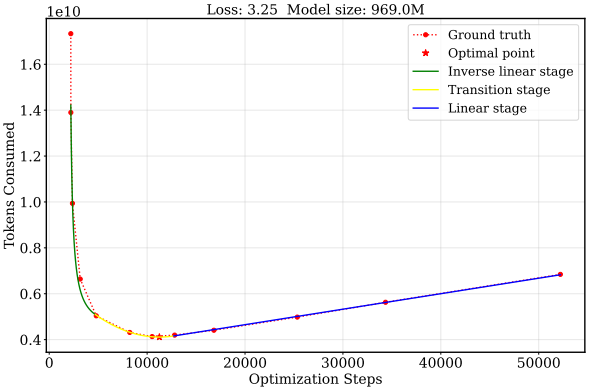

Figure 13: For a fixed loss of 3.25, the $E(S)$ curve of the InternLM2-1B model demonstrates a tripartite structure. Specifically, the optimization process is partitioned into three functional stages: the *Inverse Linear Stage*, the *Transition Stage*, and the *Linear Stage*, each representing a different scaling relationship between data consumption and training steps.

A.3 Detailed Experimental Settings and Results

A.3.1 Fitting of the New E(S) Formula

For the empirical fitting of our proposed $E(S)$ formulation, we employ the InternLM2 architecture, training 5 model variants across 13 distinct batch size configurations. The architectural specifications for these models are summarized in Table 1, while their corresponding batch size experimental setups are detailed in Table 2.

Since the $E(S)$ curves in Figure 2 are presented in log-log space, intuiting their progression in a linear coordinate system can be challenging. To provide a clearer physical interpretation, we select a fixed loss threshold and illustrate the corresponding $E(S)$ relationship in linear coordinates in Figure 13.

A.3.2 Experimental Settings and Results of Ablations

The architectural configurations for the Qwen3 models are as follows. The Qwen3-Dense model features a hidden dimension of 2,048, 48 layers, and an attention mechanism with 32 query heads and 4 key-value (KV) heads, totaling approximately 2 billion (B) parameters. The Qwen3-MoE model is configured with a hidden dimension of 1,024, 24 layers, 32 query heads and 4 KV-heads. This Mixture-of-Experts (MoE) variant incorporates 128 total experts, with 8 experts activated per token. While the total parameter count for the MoE model is 4B, it maintains only 538M active

Table 1: Architectural configurations of the InternLM2 model series.

Models	Hidden Size	Layers	Heads (KV/Q)	MLP Ratio
InternLM2-122M	1024	12	2/32	2.5
InternLM2-244M	1280	15	2/32	2.5
InternLM2-409M	1536	18	2/32	2.5
InternLM2-664M	1792	21	2/32	2.5
InternLM2-1B	2048	24	2/32	2.5

Table 2: Batch size configurations for different InternLM2 model scales in the $E(S)$ fitting experiments.

Models	Batch Sizes
InternLM2-121M	128k, 256k, 512k, 1M, 2M, 4M, 6M, 7.5M
InternLM2-244M	128k, 256k, 512k, 1M, 2M, 4M, 6M, 7.5M
InternLM2-409M	128k, 256k, 512k, 1M, 2M, 4M, 6M, 7.5M
InternLM2-664M	128k, 256k, 512k, 1M, 2M, 4M, 6M, 7.5M
InternLM2-1B	64k, 128k, 160k, 192k, 256k, 320k, 384k, 512k, 1M, 2M, 4M, 6M, 7.5M

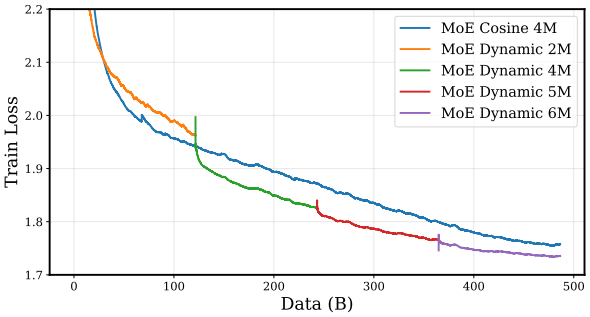

Figure 14: Loss curves of the Qwen3 MoE model trained with fixed and dynamic batch size strategies under cosine learning rate schedule.

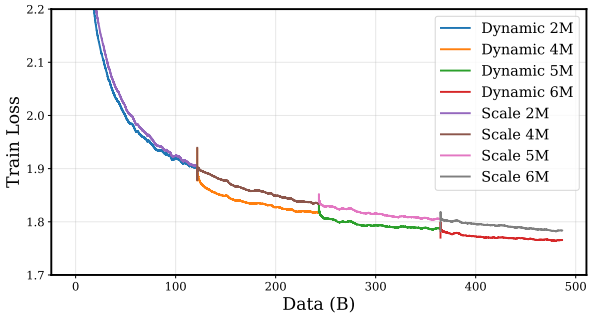

Figure 15: Comparison of training loss trajectories for dynamic batch size strategies featuring constant LR versus LR scaling proportional to the batch size.

parameters per forward pass.

Cosine learning rate schedule The learning rate follows a cosine schedule, which linearly warms up from 0 to 1.7×10^{-3} over the first 1,000 steps, followed by a cosine decay to 3.2×10^{-4} over a total training duration of 500B tokens. For the baseline configuration, we maintain a constant batch size of 4M. In contrast, our dynamic batch size strategy progressively scales the batch size at intervals of 125B tokens, following the sequence: 2M, 4M, 5M, and 6M. The training result is shown in Figure 14.

Increase the learning rate as batch size increases Using the Qwen3 MoE model, we implement a stepwise batch size adjustment—transitioning through 2M, 4M, 5M, and 6M at 125B-token intervals. In this configuration, the learning rate is scaled synchronously according to the square-root rule ($\eta \propto \sqrt{B}$). This approach is then compared against our primary experimental setup, which employs dynamic batch size adjustment while maintaining a constant learning rate.The training result is shown in Figure 15.

Switching the Sequence Length We conducted a comparative ablation study using the Qwen3 MoE model to investigate the impact of sequence length scaling. In the baseline configuration, the sequence length (*seqlen*) was fixed at 4K, with 125B tokens processed for each batch size stage: 2M, 4M, 5M, and 6M. For the experimental group, we transitioned the *seqlen* to 5K upon reaching the 250B-token mark and further increased it to 6K at 375B tokens. These adjustments resulted in batch sizes of 5M and 6M, respectively, effectively maintaining parity with the baseline's batch size trajectory while varying the underlying sample composition. The training result is shown in Figure 16.

Weight Decay We performed an ablation study based on the Qwen3 MoE setup, fixing the weight decay at 0.01 to compare the constant batch size regime against the dynamic batch size adjustment

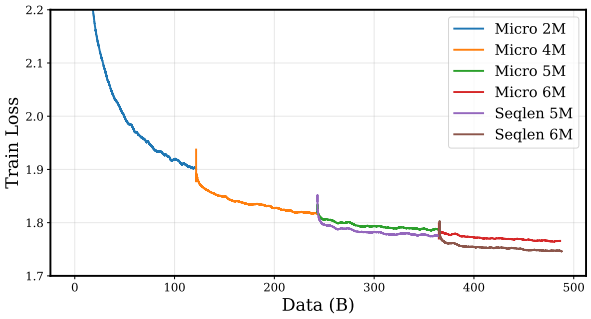

Figure 16: Comparison of training loss trajectories for dynamic batch size scaling through micro-batch adjustment and sequence length scaling.

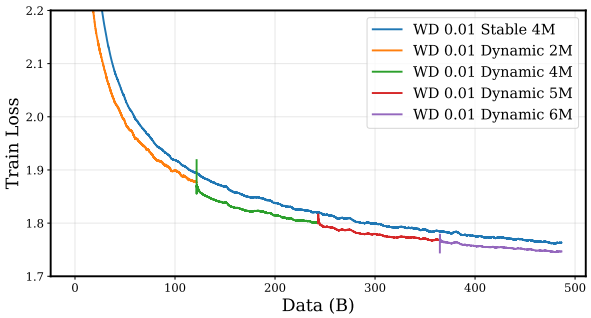

Figure 17: Comparison of training loss trajectories for fixed and dynamic batch size scheduling under different weight decay settings.

strategy. The training result is shown in Figure 17.