TOWARDS REVEALING THE EFFECT OF BATCH SIZE SCHEDULING ON LANGUAGE MODEL PRE-TRAINING

Anonymous authors
Paper under double-blind review

ABSTRACT

Training large-scale foundation models relies on effective parallelism strategies, especially batch size scheduling. However, despite its widespread practical use, the influence of batch size scheduling on training dynamics remains poorly understood. In this work, we first investigate this through a simple two-stage batch size schedule. Specifically, we train the language models with a constant learning rate using three batch size schedules: i) small constant batch size, ii) large constant batch size, and iii) a schedule that switches from small (i) to large (ii) at some switching point. We observe two notable behaviors: (1) **sudden drop**, a sharp drop in loss occurs at the switching point, compared to the loss trajectory of the small batch size; (2) **final merge**, a gradual convergence in loss to the trajectory of the large batch size. To understand the underlying mechanism behind these phenomena, we then provide a theoretical analysis from the perspective of power-law kernel regression setup. We leverage the **Functional Scaling Law (FSL)** introduced in the recent work by Li et al. (2025), which provides a theoretical framework for analyzing LLM pre-training dynamics. Our analysis shows that increasing batch size provably leads to a sudden loss drop by reducing SGD noise and guarantees convergence to the large batch trajectory at the same step level. Under the data-limited regime, our analysis further reveals a trade-off between intrinsic optimization time and SGD noise in the choice of switching point, predicting that the optimal switching point scales as a power law with total data size. Finally, we empirically validate these theoretical findings through language model pre-training experiments up to 1.1B parameters and 1T tokens, confirming the consistency of our theoretical insights.

1 INTRODUCTION

Large language model (LLM) pre-training demands immense computational resources, making training efficiency a central challenge for scaling. Parallelism plays an essential role in training efficiency, and increasing batch size is a critical mechanism for achieving effective parallelization. The critical batch size (CBS) (McCandlish et al., 2018) theory suggests increasing batch size as the noise scale grows during training: small batches are initially efficient due to low noise, but as the noise scale increases, larger batches are beneficial to improve step efficiency. Complementary to this perspective, Smith et al. (2018) proposed increasing the batch size as an alternative to learning rate decay. In practice, real-world large-scale LLMs, including GPT-3 (Brown et al., 2020), LLaMA-3 (Touvron et al., 2023), DeepSeek-V3 (DeepSeek-AI et al., 2024), and MiniMax-01 (MiniMax et al., 2025), all adopt staged batch size increases during LLM pre-training.

Despite its widespread application in large-scale model training, the effect of batch size scheduling on training dynamics remains poorly understood. While existing work has primarily focused on developing more efficient critical batch size estimators (Gray et al., 2023; 2024) and its power-law relationship to training parameters (Kaplan et al., 2020; Zhang et al., 2025), little is known about how explicit batch size schedules impacts training dynamics.

In this paper, we study a simple yet representative setting: a two-stage batch size schedule that switches from a small to a large batch under a constant learning rate regime in language model pre-training. Through this setting, we uncover two robust behaviors that consistently emerge under batch size switching:

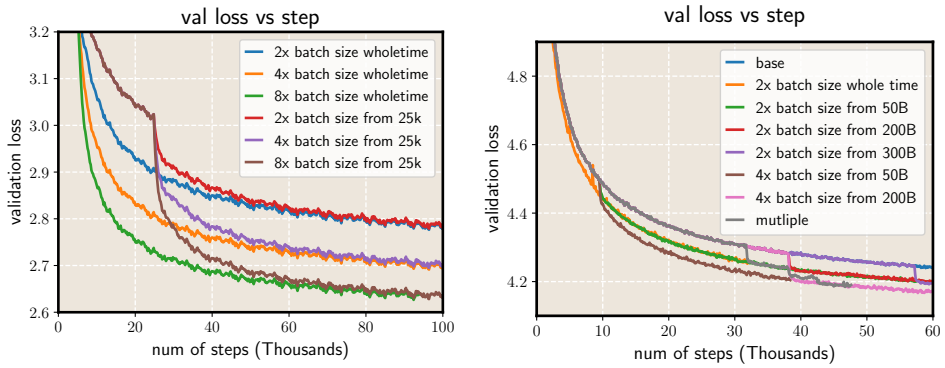

Figure 1: Validation loss versus training token under different batch-size switching times. **Left.** 500M paraemter LLaMA model trained on the C4 dataset with 10B token. The base batch size is 512. '2×/4×/8× batch size wholetime' use constant batch size 1024, 2048, and 4096 throught training process. '2×/4×/8× batch size from 25k' switch from beginning batch size 512 to batch size 1024, 2048, 4096 at 25,000 steps. **Right.** 1001M parameter MoE model trained on around 0.4T token.. 'base' trains with baseline batch size 640 for all steps. '2× batch size whole time' uses 1280 throughout training process. '2× batch size from 50B/200B/300B' switches from 640 to 1280 at 50B, 200B and 300B token, respectively. '4× batch size from 50B/200B' is similar. 'multiple' employs a staged size schedule: first 640, then 1280, then 1920, and finally 2560.

- **Sudden drop.** At the batch size switching point, the loss curve exhibits a **sudden drop** loss trajectory, deviating sharply from the small-batch trajectory.
- **Final merge.** Following the sudden drop, the loss curve gradually converges with the trajectory of training with the large batch size throughout, indicates a **final merge** when measured in training steps.

As illustrated in Figure 1, these two behaviors are consistently observed across different model architectures and training settings. Moreover, when we evaluate a staged schedule that progressively increases the batch size—from 640 to 1280, then 1920, and finally 2560—the resulting trajectory sequentially aligns with the corresponding 1280 and 2560 curves. This observation further confirms that the identified behaviors remain robust even under multi-stage batch size scheduling.

To delve deeper into these phenomena, we consider a teacher–student multi-power kernel regression task, which introduces Functional Scaling Law (FSL) (Li et al., 2025). Building upon the FSL framework, we provide the theoretical underpinnings of the observed phenomena. Under fix step, our analysis explains the sudden drop and merge behaviors. Under fix data budget, our theory further reveals: a qualitatively later switch rule, and a quantitatively power law between the optimal switching point and the total data size, as well as derives conditions for minimax-optimal batch size schedules. These results together extend prior FSL results beyond the constant-batch regime.

Furthermore, we conduct comprehensive LLM pre-training experiment to verify our theoretical analysis. Our experiments are three-fold. First, we demonstrate that a simple two-stage batch size schedule performs better than a constant batch size schedule. Second, we empirically verify our qualitative later switch rule under extensive large-scale LLM pre-training experiment. Third, we provide strong evidence for the predicted power-law between the optimal switching point scales and the total data size. Together, these results ground theoretical analysis with empirical evidence, showing the potential of FSL for understanding and designing efficient batch size schedules in LLM pre-training.

Specifically, **our contributions** can be summarized as follows:

- **Sudden drop and final merge.** Through LLM pre-training experiments, we identify two phenomena in batch size switching in the step level: (i) a sudden drop in validation loss at the switching point, and (ii) a final merge of the loss trajectory toward that of the larger batch. These findings motivate a deeper theoretical investigation of batch size scheduling dynamics.

- **Theoretical analysis via the Functional Scaling Law (FSL).** We extend the FSL framework beyond the constant-batch regime, providing a principled explanation for the sudden drop (Theorem 4.1) and final merge behaviors (Theorem 4.2). Our theory further reveals a qualitative later-switch rule, a quantitative power-law relation between the optimal switching point and total data size (Theorem 4.3), and conditions for minimax-optimal batch size schedules (Theorem 4.4).

- **Empirical validation.** We validate our theoretical predictions through extensive LLM pre-training experiments across diverse architectures and scales. Our results confirm the robustness of sudden drop and final merge, the later switch rule, and the predicted power-law scaling, thereby establishing a strong connection between FSL theory and practical batch size scheduling in LLM training.

2 Related Work

Batch size in deep learning. Batch size impacts both optimization dynamics and computational efficiency. Smith et al. (2018) proposed increasing the batch size as an alternative to learning rate decay. McCandlish et al. (2018) introduced critical batch size (CBS) as an empirical model of large batch training to balance between training step and data efficiency, and used gradient noise scale (GNS) as a proxy to estimate CBS. Several works extend this line: Kaplan et al. (2020); Zhang et al. (2025) reported empirical power-law relationships linking CBS to loss, model size and data scale; Gray et al. (2023; 2024) developed more efficient GNS estimators; Merrill et al. (2025) proposed an empirical method to directly measure CBS via branched training. A complementary direction employs adaptive sampling strategies to dynamically adjust batch sizes (De et al., 2017; Lau et al., 2024b; Ostroukhov et al., 2024) , with distributed variants (Lau et al., 2024a; 2025). In practice, LLM pre-training adopts stage-wise batch schedules: GPT-3 (Brown et al., 2020), PaLM (Chowdhery et al., 2023), LLaMA-3 (Touvron et al., 2023), and Nemotron-4 (Parmar et al., 2024; Nvidia et al., 2024), use three stages; MiniMax-01 (MiniMax et al., 2025) uses four; DeepSeek-V3 (DeepSeek-AI et al., 2024) does not report detailed stage.

Hyperparameter scheduling. Hyperparameter scheduling plays a central role in large-scale pre-training, with learning rate schedules receiving the most attention. Cosine decay (Loshchilov & Hutter, 2017) remains the prevailing choice and is widely used in large models (Brown et al., 2020; Grattafiori et al., 2024). Alternatives include linear decay (Defazio et al., 2023; Bergsma et al., 2025), Warmup-Stable-Decay (WSD) schedules (Hu et al., 2024; Hägele et al., 2024), the latter supported by theoretical understanding developed by river valley landscape (Wen et al., 2025). In parallel, schedule-free approaches aim to replace explicit learning rate schedules with adaptive mechanisms (Defazio & Mishchenko, 2023; Defazio et al., 2024). More recently, theoretical works have begun to incorporate schedules explicitly into predictive scaling-law models of training loss (Tissue et al., 2024; Luo et al., 2025; Li et al., 2025). In contrast, batch size schedules, though ubiquitous in practice, have received little theoretical treatment. Our work provides a theoretical perspective by analyzing batch size scheduling through the lens of power-law kernel regression.

3 Preliminaries

In order to analyze the role of batch size scheduling, we consider a power-law kernel (PLK) regression setup, which serves as a tractable theoretical framework that has been widely adopted in the literature on the theory of LLM pre-training (Maloney et al., 2022; Bordelon et al., 2024a; Paquette et al., 2024; Lin et al., 2024; Bordelon et al., 2024b; Zhang et al., 2025; Bahri et al., 2024; Li et al., 2025). Our analysis is primarily motivated by the recent work of Li et al. (2025), which utilized SDE modeling of SGD training dynamics under the PLK setup and introduced a novel functional scaling law (FSL) to analyze the loss dynamics during the training process for general learning rate schedules. In contrast, we apply this FSL framework to investigate the impact of batch size schedules on loss dynamics, thereby complementing the analysis of Li et al. (2025) by considering another critical training hyperparameter.

3.1 Power-Law Kernel Regression

Let $\boldsymbol{x} \in \mathbb{R}^d$ denote the data input that follows a distribution $\mathcal{D}$, and suppose the data label is observed as $y := f^*(\boldsymbol{x}) + \epsilon$, where $f^* : \mathbb{R}^d \to \mathbb{R}$ represents the target function and $\epsilon \sim \mathcal{N}(0, \sigma^2)$ denotes Gaussian noise. The target function f^* is defined as $f^*(\boldsymbol{x}) := \langle \boldsymbol{\phi}(\boldsymbol{x}), \boldsymbol{\theta}^* \rangle$, where $\boldsymbol{\phi}(\cdot) = (\phi_1(\cdot), \phi_2(\cdot), \cdots, \phi_N(\cdot))^\top \in \mathbb{R}^N$ is a feature mapping and $\boldsymbol{\theta}^* \in \mathbb{R}^N$ denotes the target parameter vector. For simplicity, we also assume that $\boldsymbol{\phi}(\boldsymbol{x}) \sim \mathcal{N}(\mathbf{0}, \mathbf{H})$ and the covariance matrix is given by $\mathbf{H} := \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}}[\boldsymbol{\phi}(\boldsymbol{x})\boldsymbol{\phi}(\boldsymbol{x})^\top] = \mathrm{diag}\{\lambda_1, \lambda_2, \cdots, \lambda_N\}$ with eigenvalues $\lambda_1 \geq \lambda_2 \cdots \geq \lambda_N$.

Our goal is to learn the target function using a student model $f(\boldsymbol{x}; \boldsymbol{\theta}) := \langle \boldsymbol{\phi}(\boldsymbol{x}), \boldsymbol{\theta} \rangle$, where $\boldsymbol{\theta}$ denotes the learnable parameter vector. We optimize the following population risk: $\mathcal{R}(\boldsymbol{\theta}) := \frac{1}{2}\mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}}[(f(\boldsymbol{x}; \boldsymbol{\theta}) - y)^2]$, where we denote $\mathcal{E}(\boldsymbol{\theta}) := \mathcal{R}(\boldsymbol{\theta}) - \frac{1}{2}\sigma^2$ as the excess risk. Then, We use an online mini-batch SGD procedure. At each iteration $1 \leq t \leq T$, we take a mini-batch $\mathcal{S}_t = \{\boldsymbol{z}_{t,i} = (\boldsymbol{x}_{t,i}, y_{t,i})\}_{i=1}^{B_t}$ drawn i.i.d. from $\mathcal{D}$ and update

$$\boldsymbol{\theta}_t := \boldsymbol{\theta}_{t-1} - \frac{\eta}{B_t} \sum_{i=1}^{B_t} \nabla_{\boldsymbol{\theta}} \tfrac{1}{2} \big(f(\boldsymbol{x}_{t,i}; \boldsymbol{\theta}_{t-1}) - y_{t,i} \big)^2, \tag{1}$$

where $\eta > 0$ is a constant learning rate, and $b := (B_1, B_2, \cdots, B_T)^\top \in \mathbb{Z}_{>0}^T$ denotes the batch size schedule (BSS).

Assumption 3.1 (Capacity condition)**.** $\lambda_j \eqsim j^{-\beta}$ for some $\beta \in (1, \infty)$.

We refer to β as the capacity parameter, which measures the decay of the kernel's eigenvalues. The assumption $\beta > 1$ ensures a finite kernel trace, i.e., $\mathrm{Tr}(\mathbf{H}) = \sum_{j=1}^{M} \lambda_j < \infty$.

Assumption 3.2 (Source condition)**.** $\theta_j^* \eqsim j^{-1/2}\lambda_j^{(s-1)/2}$ for all $j \in [N]$ for some $s \in (0, \infty)$.

We refer to s as the source parameter, which characterizes the relative task difficulty level (Paquette et al., 2024; Lin et al., 2024; Bordelon et al., 2024b; Li et al., 2025). A smaller s indicates a more challenging task. Here, we focus on the hard-task regime where $s < 1 - 1/\beta$.

3.2 SDE Modeling of SGD

The key technique is build upon the SDE modeling of SGD, which is widely used in analyzing the dynamical behavior of SGD (Li et al., 2017; 2019; Cheng et al., 2020; Li et al., 2020; 2021). For a discretization step size h, the iteration (1) can be expressed as $\boldsymbol{\theta}_t = \boldsymbol{\theta}_{t-1} - \frac{\eta}{h}\nabla\mathcal{R}(\boldsymbol{\theta}_{t-1})h - \frac{\eta}{h}\boldsymbol{\xi}_t h$, where $\boldsymbol{\xi}_t := \frac{1}{B_t}\sum_{\boldsymbol{z}\in\mathcal{S}_t} \nabla\ell(\boldsymbol{z}, \boldsymbol{\theta}_{t-1}) - \nabla\mathcal{R}(\boldsymbol{\theta}_{t-1})$ denotes the gradient noise that follows $\mathbb{E}[\boldsymbol{\xi}_t] = 0, \mathbb{E}[\boldsymbol{\xi}_t\boldsymbol{\xi}_t^\top] = \frac{1}{B_t}\Sigma(\boldsymbol{\theta}_{t-1})$, where $\Sigma(\boldsymbol{\theta})$ represents the noise covariance at $\boldsymbol{\theta}$ with the batch size 1. When h is small, this recursion is approximated by the Itô-type *time-inhomogeneous* SDE, as discussed in Li et al. (2019); Orvieto & Lucchi (2019); Ankirchner & Perko (2024):

$$\mathrm{d}\bar{\boldsymbol{\theta}}_\tau = -\frac{\eta}{h}\nabla\mathcal{R}(\bar{\boldsymbol{\theta}}_\tau)\,\mathrm{d}\tau + \frac{\eta}{h}\sqrt{\frac{h}{b(\tau)}\Sigma(\bar{\boldsymbol{\theta}}_\tau)}\,\mathrm{d}\boldsymbol{B}_\tau, \tag{2}$$

where $\mathbf{B}_\tau \in \mathbb{R}^N$ denotes the N-dimensional Brownian motion, and $b \in C(\mathbb{R}_{\geq 0})$ is the continuous batch size schedule function with $b(th) = B_t$ for all $t \in \mathbb{N}$. One can explain τ as the continuous steps or physical time. A key insight that makes the above SDE (2) analytically tractable is the anisotropic noise structure, which can be formalized as follows:

Lemma 3.3 (Anisotropic noise)**.** *For any $\boldsymbol{\theta} \in \mathbb{R}^N$, it holds that*

$$(2\mathcal{E}(\boldsymbol{\theta}) + \sigma^2)\,\mathbf{H} \preceq \boldsymbol{\Sigma}(\boldsymbol{\theta}) \preceq (4\mathcal{E}(\boldsymbol{\theta}) + \sigma^2)\,\mathbf{H}.$$

Lemma 3.3 demonstrates that the noise covariance $\boldsymbol{\Sigma}(\boldsymbol{\theta})$ approximately admits a closed-form expression: $\boldsymbol{\Sigma}(\boldsymbol{\theta}) \propto \mathcal{R}(\boldsymbol{\theta})\mathbf{H}$, as observed in (Mori et al., 2021; Wu et al., 2022; Wu & Su, 2023). This closed-form expression enables a precise characterization of the noise dynamics, thus providing a framework for tracking the SGD training dynamics.

3.3 Functional Scaling Law

Using the perspective of SDE modeling, Li et al. (2025) introduced a functional scaling law (FSL) that can predict the population risk at any continuous training step, as shown by:

Theorem 3.4 (Functional Scaling Law (FSL), Corollary of Theorem 4.1 in Li et al. (2025)). *Under Assumptions 3.1 and 3.2, the following FSL holds for sufficiently large continuous training step t:*

$$\mathbb{E}[\mathcal{R}(\boldsymbol{\theta}_t)] - \underbrace{\frac{1}{2}\sigma^2}_{\text{irreducible}} \asymp \underbrace{\frac{1}{(\eta t)^s}}_{\text{full-batch GD}} + \underbrace{\eta\sigma^2 \int_0^t \frac{\mathcal{K}(t-r)}{b(r)}\,\mathrm{d}r}_{\text{noise term}} \tag{3}$$

with $\mathcal{K}(t) := \int_0^1 u^{1-\frac{1}{\beta}} e^{-2ut}\,\mathrm{d}u$.

Theorem 3.4 establishes a scaling law for the population risk in SGD training. This law is expressed as **a functional of the batch size schedule** $b(\cdot)$ and also depends on the label-noise level σ, the capacity-source constants β and s, the learning rate η, and the training step t. Specifically, the population risk can be decomposed into three components:

- **Irreducible Risk.** Due to the label noise, there exists an irreducible risk ($\asymp \sigma^2$).
- **Intrinsic-Time Scaling.** The full-batch GD term captures the risk reduction resulting from full-batch gradient descent, following a power law with respect to the intrinsic time ηt. Notably, this term is independent of the batch size schedule.
- **SGD Noise Effects.** The noise term takes a straightforward convolutional form, capturing the risk associated with fitting the noise introduced by SGD during training. Specifically, $\eta\sigma^2/b(r)$ quantifies the magnitude of gradient noise, modulated by the learning rate, label noise level, and batch size schedules at time step r. The forgetting kernel $\mathcal{K}(t-r)$ characterizes the extent to which the fitting noise at time r is forgotten by time t.

Next, we utilize the Functional Scaling Law (FSL) to investigate the impact of batch size schedules on the training dynamics. While the FSL framework proposed by Li et al. (2025) can, in principle, be applied to study the effects of batch size, their analysis assumes a constant batch size schedule and primarily focuses on the role of the learning rate schedule. In contrast, our approach extends their work by incorporating variable batch size schedules, providing a more comprehensive understanding of how batch size scheduling influences the loss dynamics, particularly in relation to the noise effects and the evolution of the risk during training.

4 Theoretical Analysis

4.1 Sudden drop and final merge

We simulate the Functional Scaling Law (FSL) to predict the loss under a two-stage batch-size schedule. As shown in Figure 2, the switched trajectory displays a sharp *sudden drop* at the switching step and then *finally merges* onto the large-batch trajectory when measured in training steps. The simulation clearly matches the empirical phenomenon observed in Figure 1.

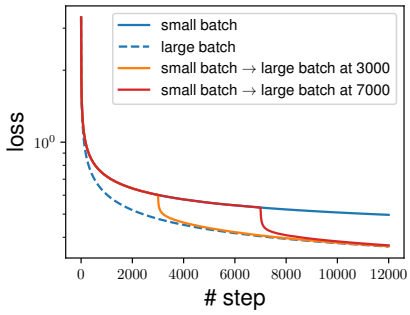

Figure 2: Functional Scaling Law prediction of loss versus step curve.

Notation. We compare three schedules: (i) a two-stage switch b_{switch} that starts with batch B_1 and switches to B_2 at step T_s; (ii) a constant large-batch schedule $b_{\text{large}} \equiv B_2$; and (iii) a constant small-batch schedule $b_{\text{small}} \equiv B_1$, with $B_1 < B_2$.

Formally,

$$b_{\text{switch}}(t) = \begin{cases} B_1, 0 < t \le T_s, \\ B_2, t > T_s, \end{cases} \quad ; \quad b_{\text{large}}(t) = B_2,\ t > 0; \quad b_{\text{small}}(t) = B_1, t\ > 0 \quad \text{with } B_1 < B_2.$$

Here, B_1 and B_2 denote the small and large batch sizes, respectively, and T_s is the switching time. We use $\mathcal{E}_{\text{switch}}(t)$, $\mathcal{E}_{\text{large}}(t)$, and $\mathcal{E}_{\text{large}}(t)$ to denote the expected loss under switched, small, and large batch size.

We first quantify the sudden drop produced by increasing the batch size at T_s, demonstrating that switched loss curve will drop at much higher speed comparing to original loss curve.

Theorem 4.1 (Sudden drop). *As $T_s \to \infty$, for time $t = O_{T_s}(1) > 0$,*
(I) for original loss curve

$$\mathcal{E}_{small}(T_s) - \mathcal{E}_{small}(T_s + t) \sim sT_s^{-s-1}t = o(1).$$

(II) for switched loss curve

$$\mathcal{E}_{switch}(T_s) - \mathcal{E}_{switch}(T_s + t) \sim h\sigma^2(\frac{1}{B_1} - \frac{1}{B_2}) \int_0^t \mathcal{K}(r)\,\mathrm{d}r = \Omega(1).$$

Theorem 4.1 establishes a precise characterization of the *sudden drop* phenomenon when switching from a smaller to a larger batch size. Part (I) shows that along the original loss curve, the reduction in loss after T_s continues to decay at a rate of order $O(T_s^{-s-1})$, reflecting the slow, diminishing improvement typical of small-batch training. In contrast, Part (II) demonstrates that the switched loss curve immediately benefits from a constant-rate reduction, scaling proportionally to the variance level σ^2, the batch size gap $(\frac{1}{B_1} - \frac{1}{B_2})$. Taken together, these results imply that switching induces an abrupt acceleration: while the original curve improves only marginally at $O(T_s^{-s-1})$, the switched curve realizes an $O(1)$ improvement within constant time, thereby producing a sharp and observable sudden drop in the training loss trajectory. *Sudden drop* characterizes the rapid deviation of the switched loss curve from the small-batch trajectory, followed by *final merge*, which quantifies the rate at which it converges toward the large-batch trajectory.

Theorem 4.2 (Final merge). *The gap between the loss of b_{switch} and b_{large} follows as $t \to \infty$*

$$\mathcal{E}_{switch}(T_s + t) - \mathcal{E}_{large}(T_s + t) \sim C_\beta h\sigma^2(\frac{1}{B_1} - \frac{1}{B_2})T_s t^{-(2-\frac{1}{\beta})}$$

where C_β is a constant that only relates to β.

Theorem 4.2 characterizes the *final merge* between the trajectories of a switched batch size schedule and the large batch schedule. As $t \to \infty$, the gap between the loss of switch schedule and large schedule admits an explicit asymptotic form, decaying at rate $O(t^{\frac{1-2\beta}{\beta}})$. This result establishes that the discrepancy induced by switching is not permanent: The switched trajectory converges to the large batch trajectory when measured in training steps.

4.2 Later switch and Power-law

We now study switching on the *data axis*, which is the relevant regime when the training budget is the total number of seen samples D. The question is: *what is the data-optimal switching point*? A natural implication follows after final merge: While loss curves corresponding to different switching times eventually merge to the large batch trajectory when measured against training steps, later switch will proceed more steps on the trajectory. Consequently, a qualitative message it: *Later switch* tends to yield a lower final loss. However, excessively late switching prevents full convergence to the larger-batch trajectory, leaving the merging incomplete. We illustrate this phenomenon in Figure 3 and provide empirical validation in Section 5.2.

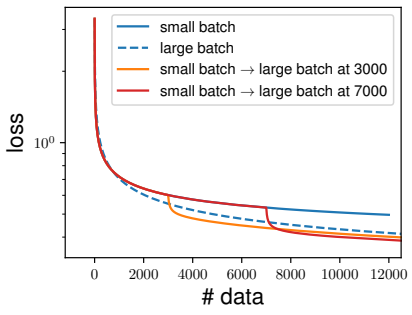

Figure 3: Functional Scaling Law prediction of loss versus data curve.

An intuitive interpretation is that a later switch allows the gradient descent (GD) term to contribute more to loss reduction, but simultaneously amplifies the influence of the noise term. Since $s < l - 1$, $O(t^{-s})$ GD term decays more slowly than the $O(t^{-l+1})$ noise term requiring a longer time horizon for its effect to diminish. We formalize this intuition with power-law theorem.

Notation. Let D denote the total number of training samples, and let $D_s \in [0, D]$ denote the number of samples at which the batch size switch occurs. Then, the switching step T_{s,D_s}, the total number of training steps t_{D_s}, and the batch size schedule $b_{\text{switch}}^{D_s}(t)$ are given by

$$T_{s,D_s} = \frac{D_s}{B_1}; \quad t_{D_s} = \frac{D_s}{B_1} + \frac{D - D_s}{B_2}; \quad b_{\text{switch}}^{D_s}(t) = \begin{cases} B_1, & 0 < t \le T_{s,D_s}, \\ B_2, & T_{s,D_s} < t < t_{D_s}. \end{cases}$$

We denote the expectation loss of batch size schedule $b_{\text{switch}}^{D_s}(t)$ on time point t to be $\mathcal{E}_{\text{switch}}^{D_s}(t)$

Theorem 4.3 (Power-law between the optimal data switch point and total data size)**.** *As $D \to \infty$, the optimal data switch point $D_s^\star(D) = \arg\min_{D_s} \mathcal{E}_{switch}^{D_s}(t_{D_s})$ obeys*

$$D - D_s^\star(D) \asymp D^{\frac{s\beta+\beta}{2\beta-1}}.$$

By the assumption of hard regime $s < 1 - 1/\beta$, we have $\frac{D_s^\star(D)}{D} \to 1,\ D - D_s^\star(D) \to \infty,$ as $D \to \infty$.

Theorem 4.3 characterizes the asymptotic location of the optimal switching point in the data domain. Specifically, it shows that as the total data budget $D \to \infty$, the optimal switching point $D_s^\star(D)$ lies increasingly close to the end of training, with the residual gap to D scaling as $D^{\frac{s\beta+\beta}{2\beta-1}}$, this implies that the relative ratio converge to zero, while the absolute difference to D diverges. Intuitively, this result reveals that in the large-data limit, the optimal strategy is to maintain small-batch training for nearly the entire dataset, and perform *later switch* to the larger batch size at a point very close to completion. However, the switch is not exactly at the end: the diverging yet vanishingly small fraction $D - c^\star(D)$ ensures that the benefit of the sudden-drop phenomenon is still exploited.

4.3 Optimal Batch Size Schedule

Theorem 4.4 (optimal batch size schedule)**.** *For any fixed data size D, the optimal batch size schedule is given by $B^\star(t)_{t=0}^{T_\star}$ with optimal time T^*, where*

$$T^\star \asymp D^{\frac{\beta}{1+s\beta}},\ B_t \asymp \frac{(T-t)^{1/2\beta-1}}{T^{1/2\beta}} D.$$

reaches optimal loss $\mathcal{E}^\star \asymp D^{-\frac{s\beta}{1+s\beta}}$.

Theorem 4.5 (Minimax lower bound, Corollary of Theorem 2 in Caponnetto & De Vito (2007))**.** *Suppose Assumptions 3.1, 3.2 hold. Let $\mathcal{T}_n = \{(x_i, y_i)\}_{i=1}^n$ be a dataset of n samples drawn* i.i.d. *from ρ. The minimax risk satisfies*

$$\inf_{\widehat{\theta}_D} \sup_{\theta^*, \rho} \mathbb{E}_{\mathcal{T}_D} \left[\mathcal{E}(\widehat{\theta}_D) \right] \gtrsim D^{-\frac{s\beta}{1+s\beta}},$$

where the infimum is taken over all estimators $\widehat{\theta}_D$ (i.e., measurable functions of $\mathcal{T}_D$) and the supremum is taken over all targets w^* and data distributions ρ satisfying the stated assumptions.

In large-scale LLM pre-training, practitioners often adopt the heuristic of ramping up the batch size over the course of training (Brown et al., 2020; DeepSeek-AI et al., 2024). Within our theoretical framework, we investigate a fundamental question: under a data-limited regime, what is the optimal batch size schedule (in the asymptotic sense)? Theorem 4.4 shows that an optimal schedule exists in our setup, showing that a progressively increasing batch size and achieving the optimal convergence rate. Furthermore, we demonstrate that this optimal batch size schedule is sufficient to attain the minimax lower bound proven by Theorem 4.5.

5 Experiment

5.1 Experiment Setup

We evaluate batch size scheduling for the task of LLM pre-training across different model architectures, parameter scales, datasets. The main experimental configurations are summarized below, with further implementation details provided in Appendix B.

We conducted experiments in two distinct settings to ensure consistency of observed phenomena across different scales:

- **Small-scale.** We adopt the widely used **NanoGPT** codebase (Karpathy, 2022) in small-scale experiments. Specifically, we evaluate the standard dense **LLaMA** architectures (Touvron et al., 2023). We evaluate on the **Colossal Clean Crawled Corpus (C4)** dataset (Raffel et al., 2020). The total number of training tokens is set to be approximately 20 times the number of model parameters, in accordance with the Chinchilla scaling law (Hoffmann et al., 2022), which is widely used in small-scale experiments. We experiment with model size **50M**, **200M** and **492M** parameters.

- **Large-scale.** We conducted large-scale experiment using the widely adopted **Megatron-LM** codebase (Shoeybi et al., 2019). We employ a novel sparse MoE design, Mixture-of-Experts (MoE) architecture, the shortcut-connected MoE (**ScMoE**) (Cai et al., 2025), which has demonstrated notable gains in inference efficiency and throughput compared to models of a comparable scale (LongCat et al., 2025). To align our experiment with real world large-scale LLM pre-training, the models are trained on a token-to-parameter ratio that significantly exceeds the 20:1 guideline, corresponding to the beyond-Chinchilla-optimal regime (Sardana et al., 2024). We experiment with two model sizes: (i) **1001M** total parameters with **209M** activated for each token, training on around **0.4T** tokens; (ii) **1119M** total parameters with **291M** activated for each token, training on around **1T** tokens.

Theorem 4.4 motivates a gradually increasing batch size schedule. However, batch size scheduling faces hardware constraints that limit its flexibility. First, batch size is inherently discrete and hardware-bound: it must be a positive integer and divisible by the number of data-parallel workers. Second, computational efficiency is also a critical factor, too small batch size lead to poor GPU utilization. In LLM pre-training, adjusting batch size incurs non-trivial overhead, requiring reconfiguration of data pipelines, memory allocation, and distributed communication patterns. Consequently, batch size schedules are typically implemented as stage-wise changes among a limited set of discrete values. Due to these hardware constraints, our experiments focus on a two-stage schedule under a fixed data budget.

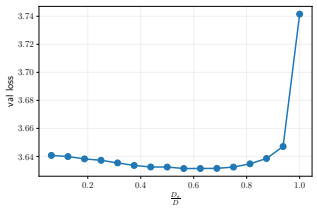

Figure 4: Validation loss under different batch size switching points. The x-axis denotes the fraction of data processed before switching.

In Figure 4, we present results from pre-training a 200M-parameter LLaMA model on 4B tokens using a two-stage batch size schedule, switching from 256 to 512 at different points in training. The results highlight the advantage of two-stage scheduling over a constant batch size, yielding consistently lower validation loss. Moreover, the trend corroborates our theoretical analysis in Section 4.2: later switching generally achieves a smaller final loss, while excessively late switching prevents the noise term from converging, leading to high terminal loss.

5.2 Later Switch

We present validation loss versus training tokens across different batch size switching times to examine the later switch rule under multiple architectures, data scales, and schedules. As shown in Figure 5, for a fixed data budget, later batch size switching consistently yields performance gains than early switching when evaluated under the same data budget, regardless of the specific switching ratio.We also plot validation loss versus training steps, validating the sudden drop and final merge phenomenon, as shown in Figure 1 and Figure 7. Finally, We further explore later switch rule in multi-stage batch size schedule in Appendix B.4, with later switches still outperforming earlier switches. Together, these results demonstrate that the observed phenomena are robust across both model and data scales.

5.3 Power Law

In this section, we verify Theorem 4.3, which establishes a qualitative power law relation between total data size D and optimal data switch point D^*. Specifically, we have

$$D - D_s^\star(D) \sim CD^e \text{ where } e = \frac{s\beta + \beta}{2\beta - 1} \in (0, 1) \tag{4}$$

Here C and e are two constants independent of D. Here, the notation $\sim$ is understood in the asymptotic sense. Take the logarithm of both sides of Equation 4 yields the near-linear relation

$$\log(D - D_s^\star(D)) = \log C + e \log D + o(1), \text{ as } D \to \infty \tag{5}$$

which provides a convenient form for empirical verification in log–log coordinates.

We conduct experiments using a 50M-parameter model trained on the C4 dataset with token budgets ranging from 1.3B to 5B. For each total token budget D, we perform an extensive grid search to

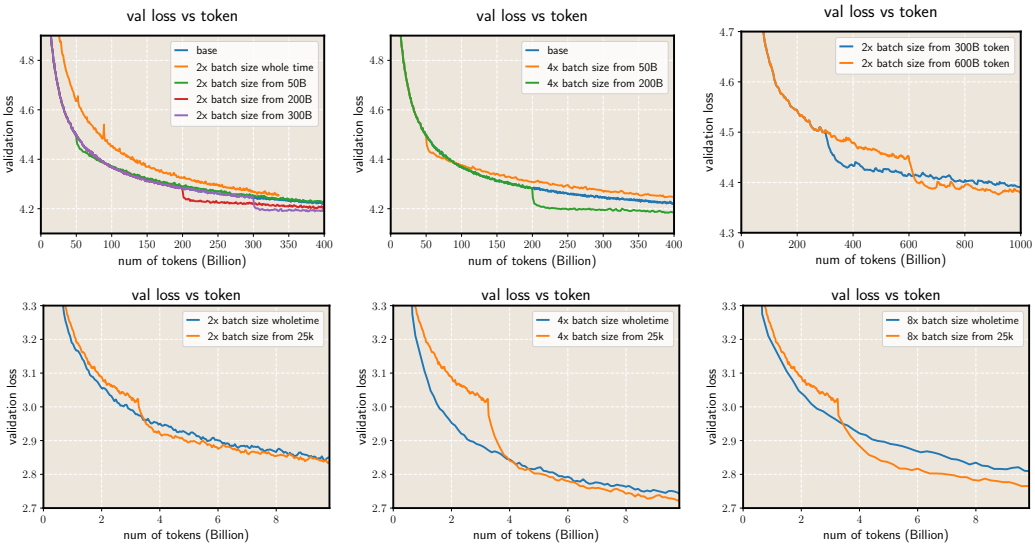

Figure 5: **Upper Left and Middle:** Validation loss versus training token under different batch-size switching times using 1001M parameter MoE model trained on around 0.4T token, switching batch size from 640 to 1280, and 2560. **Upper Right:** Validation loss versus training token under different batch-size switching times using 1119M parameter MoE model trained on around 1T token, switching batch size from 1024 to 2048. **Lower Left, Middle and Right:** Validation loss versus training token under different batch-size switching times using 500M parameter LLaMA model trained on around 10B token, switching batch size from 512 to 1024, 2048 and 4096.

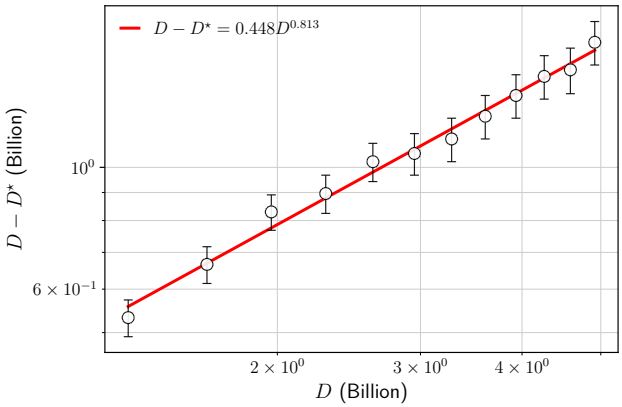

Figure 6: Power law relationship between D and $D - D^\star$. The R^2 of the fits is 0.990. We see that the power law provides a good fit to the empirical data.

identify the optimal batch-size switching point $D^\star$. We then fit the empirical results using least-squares regression against Equation 5. As shown in Figure 6, the fitted curves align closely with the theoretical prediction, achieving an R^2 value of 0.990, a fair high value. This indicates that, despite Equation 5 being derived in an asymptotic regime, it provides a decent approximation. Moreover, the estimated exponent e lies in the interval $(0, 1)$, in agreement with the theoretical analysis.

6 Conclusion

In this work, we study a simple two-stage batch size schedule in language model pre-training, switching from small to large batches under a constant learning rate. Using a power-law kernel regression framework, We explains two consistent behaviors: a sudden loss drop at the switch and a final merge to the large-batch trajectory, these behaviors arise from reduced gradient noise. Furthermore, we predicted that the optimal switching point scales as a power law with data size. Experiments across models and scales validate these predictions.

REFERENCES

Stefan Ankirchner and Stefan Perko. A comparison of continuous-time approximations to stochastic gradient descent. *Journal of Machine Learning Research*, 25(13):1–55, 2024.

Yasaman Bahri, Ethan Dyer, Jared Kaplan, Jaehoon Lee, and Utkarsh Sharma. Explaining neural scaling laws. *Proceedings of the National Academy of Sciences*, 121(27):e2311878121, 2024.

Shane Bergsma, Nolan Dey, Gurpreet Gosal, Gavia Gray, Daria Soboleva, and Joel Hestness. Straight to zero: Why linearly decaying the learning rate to zero works best for llms. *arXiv preprint arXiv:2502.15938*, 2025.

Blake Bordelon, Alexander Atanasov, and Cengiz Pehlevan. A dynamical model of neural scaling laws. *arXiv preprint arXiv:2402.01092*, 2024a.

Blake Bordelon, Alexander Atanasov, and Cengiz Pehlevan. How feature learning can improve neural scaling laws. *arXiv preprint arXiv:2409.17858*, 2024b.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

Weilin Cai, Juyong Jiang, Le Qin, Junwei Cui, Sunghun Kim, and Jiayi Huang. Shortcut-connected expert parallelism for accelerating mixture-of-experts. In *International Conference on Machine Learning*, 2025.

Andrea Caponnetto and Ernesto De Vito. Optimal rates for the regularized least-squares algorithm. *Foundations of Computational Mathematics*, 7:331–368, 2007.

Xiang Cheng, Dong Yin, Peter Bartlett, and Michael Jordan. Stochastic gradient and langevin processes. In *International Conference on Machine Learning*, pp. 1810–1819. PMLR, 2020.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, Parker Schuh, Kensen Shi, Sasha Tsvyashchenko, Joshua Maynez, Abhishek Rao, Parker Barnes, Yi Tay, Noam Shazeer, Vinodkumar Prabhakaran, Emily Reif, Nan Du, Ben Hutchinson, Reiner Pope, James Bradbury, Jacob Austin, Michael Isard, Guy Gur-Ari, Pengcheng Yin, Toju Duke, Anselm Levskaya, Sanjay Ghemawat, Sunipa Dev, Henryk Michalewski, Xavier Garcia, Vedant Misra, Kevin Robinson, Liam Fedus, Denny Zhou, Daphne Ippolito, David Luan, Hyeontaek Lim, Barret Zoph, Alexander Spiridonov, Ryan Sepassi, David Dohan, Shivani Agrawal, Mark Omernick, Andrew M. Dai, Thanumalayan Sankaranarayana Pillai, Marie Pellat, Aitor Lewkowycz, Erica Moreira, Rewon Child, Oleksandr Polozov, Katherine Lee, Zongwei Zhou, Xuezhi Wang, Brennan Saeta, Mark Diaz, Orhan Firat, Michele Catasta, Jason Wei, Kathy Meier-Hellstern, Douglas Eck, Jeff Dean, Slav Petrov, and Noah Fiedel. Palm: Scaling language modeling with pathways. *Journal of Machine Learning Research*, 24(240):1–113, 2023.

Soham De, Abhay Yadav, David Jacobs, and Tom Goldstein. Automated inference with adaptive batches. In *Artificial Intelligence and Statistics*, pp. 1504–1513. PMLR, 2017.

DeepSeek-AI, Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, Damai Dai, Daya Guo, Dejian Yang, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, H. Zhang, Han Bao, Hanwei Xu, Haocheng Wang, Haowei Zhang, Honghui Ding, Huajian Xin, Huazuo Gao, Hui Li, Hui Qu, J. L. Cai, Jian Liang, Jianzhong Guo, Jiaqi Ni, Jiashi Li, Jiawei Wang, Jin Chen, Jingchang Chen, Jingyang Yuan, Junjie Qiu, Junlong Li, Junxiao Song, Kai Dong, Kai Hu, Kaige Gao, Kang Guan, Kexin Huang, Kuai Yu, Lean Wang, Lecong Zhang, Lei Xu, Leyi Xia, Liang Zhao, Litong Wang, Liyue Zhang, Meng Li, Miaojun Wang, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Mingming Li, Ning Tian, Panpan Huang,

Peiyi Wang, Peng Zhang, Qiancheng Wang, Qihao Zhu, Qinyu Chen, Qiushi Du, R. J. Chen, R. L. Jin, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang, Runxin Xu, Ruoyu Zhang, Ruyi Chen, S. S. Li, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shaoqing Wu, Shengfeng Ye, Shengfeng Ye, Shirong Ma, Shiyu Wang, Shuang Zhou, Shuiping Yu, Shunfeng Zhou, Shuting Pan, T. Wang, et al. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*, 2024.

Aaron Defazio and Konstantin Mishchenko. Learning-rate-free learning by d-adaptation. In *International Conference on Machine Learning*, pp. 7449–7479. PMLR, 2023.

Aaron Defazio, Ashok Cutkosky, Harsh Mehta, and Konstantin Mishchenko. Optimal linear decay learning rate schedules and further refinements. *arXiv preprint arXiv:2310.07831*, 2023.

Aaron Defazio, Xingyu Yang, Ahmed Khaled, Konstantin Mishchenko, Harsh Mehta, and Ashok Cutkosky. The road less scheduled. *Advances in Neural Information Processing Systems*, 37: 9974–10007, 2024.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, Arun Rao, Aston Zhang, Aurelien Rodriguez, Austen Gregerson, Ava Spataru, Baptiste Roziere, Bethany Biron, Binh Tang, Bobbie Chern, Charlotte Caucheteux, Chaya Nayak, Chloe Bi, Chris Marra, Chris McConnell, Christian Keller, Christophe Touret, Chunyang Wu, Corinne Wong, Cristian Canton Ferrer, Cyrus Nikolaidis, Damien Allonsius, Daniel Song, Danielle Pintz, Danny Livshits, Danny Wyatt, David Esiobu, Dhruv Choudhary, Dhruv Mahajan, Diego Garcia-Olano, Diego Perino, Dieuwke Hupkes, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Gavia Gray, Anshul Samar, and Joel Hestness. Efficient and approximate per-example gradient norms for gradient noise scale. In *Workshop on Advancing Neural Network Training: Computational Efficiency, Scalability, and Resource Optimization (WANT@ NeurIPS 2023)*, 2023.

Gavia Gray, Shane Bergsma, Joel Hestness, et al. Normalization layer per-example gradients are sufficient to predict gradient noise scale in transformers. *Advances in Neural Information Processing Systems*, 37:93510–93539, 2024.

Alex Hägele, Elie Bakouch, Atli Kosson, Leandro Von Werra, Martin Jaggi, et al. Scaling laws and compute-optimal training beyond fixed training durations. *Advances in Neural Information Processing Systems*, 37:76232–76264, 2024.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, Tom Hennigan, Eric Noland, Katie Millican, George van den Driessche, Bogdan Damoc, Aurelia Guy, Simon Osindero, Karen Simonyan, Erich Elsen, Jack W. Rae, Oriol Vinyals, and Laurent Sifre. Training compute-optimal large language models. *arXiv preprint arXiv:2203.15556*, 2022.

Shengding Hu, Yuge Tu, Xu Han, Chaoqun He, Ganqu Cui, Xiang Long, Zhi Zheng, Yewei Fang, Yuxiang Huang, Weilin Zhao, Xinrong Zhang, Zheng Leng Thai, Kaihuo Zhang, Chongyi Wang, Yuan Yao, Chenyang Zhao, Jie Zhou, Jie Cai, Zhongwu Zhai, Ning Ding, Chao Jia, Guoyang Zeng, Dahai Li, Zhiyuan Liu, and Maosong Sun. Minicpm: Unveiling the potential of small language models with scalable training strategies. In *Conference on Language Modeling*, 2024.

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

Andrej Karpathy. NanoGPT. https://github.com/karpathy/nanoGPT, 2022.

Tim Tsz-Kit Lau, Weijian Li, Chenwei Xu, Han Liu, and Mladen Kolar. Communication-efficient adaptive batch size strategies for distributed local gradient methods. *arXiv preprint arXiv:2406.13936*, 2024a.

Tim Tsz-Kit Lau, Han Liu, and Mladen Kolar. Adadagrad: Adaptive batch size schemes for adaptive gradient methods. *arXiv preprint arXiv:2402.11215*, 2024b.

Tim Tsz-Kit Lau, Weijian Li, Chenwei Xu, Han Liu, and Mladen Kolar. Adaptive batch size schedules for distributed training of language models with data and model parallelism. In *Proceedings of Conference on Parsimony and Learning*, 2025.

Binghui Li, Fengling Chen, Zixun Huang, Lean Wang, and Lei Wu. Unveiling the role of learning rate schedules via functional scaling laws. *arXiv preprint arXiv:2509.19189*, 2025.

Qianxiao Li, Cheng Tai, and E Weinan. Stochastic modified equations and adaptive stochastic gradient algorithms. In *International Conference on Machine Learning*, pp. 2101–2110. PMLR, 2017.

Qianxiao Li, Cheng Tai, and E Weinan. Stochastic modified equations and dynamics of stochastic gradient algorithms I: Mathematical foundations. *Journal of Machine Learning Research*, 20(40): 1–47, 2019.

Zhiyuan Li, Kaifeng Lyu, and Sanjeev Arora. Reconciling modern deep learning with traditional optimization analyses: The intrinsic learning rate. *Advances in Neural Information Processing Systems*, 33:14544–14555, 2020.

Zhiyuan Li, Sadhika Malladi, and Sanjeev Arora. On the validity of modeling SGD with stochastic differential equations (SDEs). *Advances in Neural Information Processing Systems*, 34:12712–12725, 2021.

Licong Lin, Jingfeng Wu, Sham M Kakade, Peter L Bartlett, and Jason D Lee. Scaling laws in linear regression: Compute, parameters, and data. *arXiv preprint arXiv:2406.08466*, 2024.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

Meituan LongCat, Bayan, Bei Li, Bingye Lei, Bo Wang, Bolin Rong, Chao Wang, Chao Zhang, Chen Gao, Chen Zhang, Cheng Sun, Chengcheng Han, Chenguang Xi, Chi Zhang, Chong Peng, Chuan Qin, Chuyu Zhang, Cong Chen, Congkui Wang, Dan Ma, Daoru Pan, Defei Bu, Dengchang Zhao, Deyang Kong, Dishan Liu, Feiye Huo, Fengcun Li, Fubao Zhang, Gan Dong, Gang Liu, Gang Xu, Ge Li, Guoqiang Tan, Guoyuan Lin, Haihang Jing, Haomin Fu, Haonan Yan, Haoxing Wen, Haozhe Zhao, Hong Liu, Hongmei Shi, Hongyan Hao, Hongyin Tang, Huantian Lv, Hui Su, Jiacheng Li, Jiahao Liu, Jiahuan Li, Jiajun Yang, Jiaming Wang, Jian Yang, Jianchao Tan, Jiaqi Sun, Jiaqi Zhang, Jiawei Fu, Jiawei Yang, Jiaxi Hu, and Jiayu Qin. Longcat-flash technical report. *arXiv preprint arXiv:2509.01322*, 2025.

Ilya Loshchilov and Frank Hutter. Sgdr: Stochastic gradient descent with warm restarts. In *International Conference on Learning Representations*, 2017.

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 *International Conference on Learning Representations*, 2025.

Alexander Maloney, Daniel A Roberts, and James Sully. A solvable model of neural scaling laws. *arXiv preprint arXiv:2210.16859*, 2022.

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

William Merrill, Shane Arora, Dirk Groeneveld, and Hannaneh Hajishirzi. Critical batch size revisited: A simple empirical approach to large-batch language model training. *arXiv preprint arXiv:2505.23971*, 2025.

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, Jin Zhu, Jingtao Han, Jingyang Li, Junbin Xie, Junhao Xu, Junjie Yan, Kaishun Zhang, Kecheng Xiao, Kexi Kang, Le Han, Leyang Wang, Lianfei Yu, Liheng Feng, Lin Zheng, Linbo Chai, Long Xing, Meizhi Ju, Mingyuan Chi, Mozhi Zhang, Peikai Huang, Pengcheng Niu, Pengfei Li, Pengyu Zhao, Qi Yang, Qidi Xu, Qiexiang

Wang, Qin Wang, Qiuhui Li, Ruitao Leng, Shengmin Shi, Shuqi Yu, Sichen Li, Songquan Zhu, Tao Huang, Tianrun Liang, Weigao Sun, Weixuan Sun, Weiyu Cheng, Wenkai Li, Xiangjun Song, Xiao Su, Xiaodong Han, Xinjie Zhang, Xinzhu Hou, Xu Min, Xun Zou, Xuyang Shen, Yan Gong, Yingjie Zhu, Yipeng Zhou, Yiran Zhong, Yongyi Hu, Yuanxiang Fan, Yue Yu, Yufeng Yang, Yuhao Li, Yunan Huang, Yunji Li, Yunpeng Huang, Yunzhi Xu, Yuxin Mao, Zehan Li, Zekang Li, Zewei Tao, Zewen Ying, Zhaoyang Cong, Zhen Qin, Zhenhua Fan, Zhihang Yu, Zhuo Jiang, and Zijia Wu. Minimax-01: Scaling foundation models with lightning attention. *arXiv preprint arXiv:2501.08313*, 2025.

Takashi Mori, Liu Ziyin, Kangqiao Liu, and Masahito Ueda. Power-law escape rate of SGD. *arXiv preprint arXiv:2105.09557*, 2021.

Nvidia, Bo Adler, Niket Agarwal, Ashwath Aithal, Dong H. Anh, Pallab Bhattacharya, Annika Brundyn, Jared Casper, Bryan Catanzaro, Sharon Clay, Jonathan Cohen, Sirshak Das, Ayush Dattagupta, Olivier Delalleau, Leon Derczynski, Yi Dong, Daniel Egert, Ellie Evans, Aleksander Ficek, Denys Fridman, Shaona Ghosh, Boris Ginsburg, Igor Gitman, Tomasz Grzegorzek, Robert Hero, Jining Huang, Vibhu Jawa, Joseph Jennings, Aastha Jhunjhunwala, John Kamalu, Sadaf Khan, Oleksii Kuchaiev, Patrick LeGresley, Hui Li, Jiwei Liu, Zihan Liu, Eileen Long, Ameya Sunil Mahabaleshwarkar, Somshubra Majumdar, James Maki, Miguel Martinez, Maer Rodrigues de Melo, Ivan Moshkov, Deepak Narayanan, Sean Narenthiran, Jesus Navarro, Phong Nguyen, Osvald Nitski, Vahid Noroozi, Guruprasad Nutheti, Christopher Parisien, Jupinder Parmar, Mostofa Patwary, Krzysztof Pawelec, Wei Ping, Shrimai Prabhumoye, Rajarshi Roy, Trisha Saar, Vasanth Rao Naik Sabavat, Sanjeev Satheesh, Jane Polak Scowcroft, Jason Sewall, Pavel Shamis, Gerald Shen, Mohammad Shoeybi, Dave Sizer, Misha Smelyanskiy, Felipe Soares, Makesh Narsimhan Sreedhar, Dan Su, Sandeep Subramanian, Shengyang Sun, Shubham Toshniwal, Hao Wang, Zhilin Wang, Jiaxuan You, Jiaqi Zeng, Jimmy Zhang, Jing Zhang, Vivienne Zhang, Yian Zhang, and Chen Zhu. Nemotron-4 340b technical report. *arXiv preprint arXiv:2406.11704*, 2024.

Antonio Orvieto and Aurelien Lucchi. Continuous-time models for stochastic optimization algorithms. *Advances in Neural Information Processing Systems*, 32, 2019.

Petr Ostroukhov, Aigerim Zhumabayeva, Chulu Xiang, Alexander Gasnikov, Martin Takáč, and Dmitry Kamzolov. Adabatchgrad: Combining adaptive batch size and adaptive step size. *arXiv preprint arXiv:2402.05264*, 2024.

Elliot Paquette, Courtney Paquette, Lechao Xiao, and Jeffrey Pennington. 4+3 phases of compute-optimal neural scaling laws. *arXiv preprint arXiv:2405.15074*, 2024.

Jupinder Parmar, Shrimai Prabhumoye, Joseph Jennings, Mostofa Patwary, Sandeep Subramanian, Dan Su, Chen Zhu, Deepak Narayanan, Aastha Jhunjhunwala, Ayush Dattagupta, Vibhu Jawa, Jiwei Liu, Ameya Mahabaleshwarkar, Osvald Nitski, Annika Brundyn, James Maki, Miguel Martinez, Jiaxuan You, John Kamalu, Patrick LeGresley, Denys Fridman, Jared Casper, Ashwath Aithal, Oleksii Kuchaiev, Mohammad Shoeybi, Jonathan Cohen, and Bryan Catanzaro. Nemotron-4 15b technical report. *arXiv preprint arXiv:2402.16819*, 2024.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *The Journal of Machine Learning Research*, 21(1):5485–5551, 2020.

Nikhil Sardana, Jacob Portes, Sasha Doubov, and Jonathan Frankle. Beyond chinchilla-optimal: Accounting for inference in language model scaling laws. In *International Conference on Machine Learning*, 2024.

Mohammad Shoeybi, Mostofa Patwary, Raul Puri, Patrick LeGresley, Jared Casper, and Bryan Catanzaro. Megatron-lm: Training multi-billion parameter language models using model parallelism. *arXiv preprint arXiv:1909.08053*, 2019.

Samuel L Smith, Pieter-Jan Kindermans, Chris Ying, and Quoc V Le. Don't decay the learning rate, increase the batch size. In *International Conference on Learning Representations*, 2018.

Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063, 2024.

Howe Tissue, Venus Wang, and Lu Wang. Scaling law with learning rate annealing. *arXiv preprint arXiv:2408.11029*, 2024.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

Kaiyue Wen, Zhiyuan Li, Jason Wang, David Hall, Percy Liang, and Tengyu Ma. Understanding warmup-stable-decay learning rates: A river valley loss landscape perspective. *International Conference on Learning Representations*, 2025.

Lei Wu and Weijie J Su. The implicit regularization of dynamical stability in stochastic gradient descent. In *International Conference on Machine Learning*, pp. 37656–37684. PMLR, 2023.

Lei Wu, Mingze Wang, and Weijie Su. The alignment property of SGD noise and how it helps select flat minima: A stability analysis. *Advances in Neural Information Processing Systems*, 35: 4680–4693, 2022.

Hanlin Zhang, Depen Morwani, Nikhil Vyas, Jingfeng Wu, Difan Zou, Udaya Ghai, Dean Foster, and Sham Kakade. How does critical batch size scale in pre-training? *International Conference on Learning Representations*, 2025.

Jiawei Zhao, Zhenyu Zhang, Beidi Chen, Zhangyang Wang, Anima Anandkumar, and Yuandong Tian. Galore: Memory-efficient llm training by gradient low-rank projection. *International Conference on Machine Learning*, 2024.

Hanqing Zhu, Zhenyu Zhang, Wenyan Cong, Xi Liu, Sem Park, Vikas Chandra, Bo Long, David Z Pan, Zhangyang Wang, and Jinwon Lee. Apollo: Sgd-like memory, adamw-level performance. *Conference on Machine Learning and Systems*, 2025.

Appendix

A Proof in Section 4

Lemma A.1 (Asymptotic behavior for $\mathcal{K}(t)$)**.**

(I) As $t \to 0^+$

$$\mathcal{K}(t) = \frac{\beta}{2\beta - 1} - \frac{2\beta}{3\beta - 1} t + O\left(t^2\right).$$

(II) As $t \to \infty$

$$\mathcal{K}(t) \sim \frac{\Gamma(\frac{2\beta-1}{\beta})}{(2t)^{\frac{2\beta-1}{\beta}}}, \quad \mathcal{K}'(t) \sim -\frac{2\Gamma(\frac{3\beta-1}{\beta})}{(2t)^{\frac{3\beta-1}{\beta}}}.$$

(III)

$$\int_0^\infty \mathcal{K}(t)\,\mathrm{d}t = \frac{\beta}{2(\beta-1)}.$$

(IV)

As $T \to \infty$,

$$\int_0^T \mathcal{K}(t)\,\mathrm{d}t = \frac{\beta}{2(\beta-1)} - \frac{\Gamma(\frac{2\beta-1}{\beta})\beta}{2^{\frac{2\beta-1}{\beta}}(\beta-1)} T^{\frac{1-\beta}{\beta}} + o(T^{\frac{1-\beta}{\beta}}).$$

Proof. (I)

$$\mathcal{K}(t) = \int_0^1 u^{1-\frac{1}{\beta}} \sum_{i=0}^{n} \frac{(2ut)^i}{i!}\,\mathrm{d}u = \sum_{i=0}^{t} \frac{(-2t)^i}{i!(i+1-\frac{1}{\beta})} = \frac{\beta}{2\beta-1} - \frac{2\beta}{3\beta-1}t + O\left(t^2\right).$$

(II)

$$(2t)^{2-\frac{1}{\beta}}\mathcal{K}(t) = (2t)^s \int_0^1 u^{1-\frac{1}{\beta}} e^{-2ut}\,\mathrm{d}u = \int_0^{2t} u^{1-\frac{1}{\beta}} e^{-t}\,\mathrm{d}u \to \int_0^{\infty} u^{1-\frac{1}{\beta}} e^{-t}\,\mathrm{d}u = \Gamma(\frac{2\beta-1}{\beta}).$$

Hence

$$\mathcal{K}'(t) = -2\int_0^1 u^{2-\frac{1}{\beta}} e^{-2ut} du \sim -2\frac{\Gamma(3-\frac{1}{\beta})}{(2t)^{3-\frac{1}{\beta}}} = -\frac{2\Gamma(\frac{3\beta-1}{\beta})}{(2t)^{\frac{3\beta-1}{\beta}}}.$$

(III)

$$\int_0^{\infty} \mathcal{K}(t)\,\mathrm{d}t = \int_0^{\infty}\int_0^1 u^{1-\frac{1}{\beta}} e^{-2ut}\,\mathrm{d}u\,\mathrm{d}t = \int_0^1 \frac{1}{2}u^{-\frac{1}{\beta}}\,\mathrm{d}u = \frac{\beta}{2(\beta-1)}.$$

(IV)

By (II), we have

$$\int_T^{\infty} \mathcal{K}(t)\,\mathrm{d}t = \int_T^{\infty} \frac{\Gamma(\frac{2\beta-1}{\beta})}{(2t)^{\frac{2\beta-1}{\beta}}} + o(t^{-l})\,\mathrm{d}t = \frac{\Gamma(\frac{2\beta-1}{\beta})\beta}{2^{\frac{2\beta-1}{\beta}}(\beta-1)}T^{\frac{1-\beta}{\beta}} + o(T^{\frac{1-\beta}{\beta}}).$$

Hence

$$\int_0^T \mathcal{K}(t)\,\mathrm{d}t = \int_0^{\infty} \mathcal{K}(t)\,\mathrm{d}t - \int_T^{\infty} \mathcal{K}(t)\,\mathrm{d}t = \frac{\beta}{2(\beta-1)} - \frac{\Gamma(\frac{2\beta-1}{\beta})\beta}{2^{\frac{2\beta-1}{\beta}}(\beta-1)}T^{\frac{1-\beta}{\beta}} + o(T^{\frac{1-\beta}{\beta}}).$$

□

A.1 Proof for Theorem 4.1

Proof. (I) Following Theorem 3.4, for original loss curve

$$\mathcal{E}_{\text{small}}(T_s) - \mathcal{E}_{\text{small}}(T_s+t) = (\frac{1}{T_s^s} - \frac{1}{(T_s+t)^s}) + h\sigma^2 B_1^{-1}(\int_{T_s}^{T_s+t} \mathcal{K}(r)\,\mathrm{d}r).$$

With the help of Lemma A.1, we have

$$\begin{aligned}\mathcal{E}_{\text{small}}(T_s) - \mathcal{E}_{\text{small}}(T_s+t) &= sT_s^{-s-1}t + O(T_s^{-s-2}) + O(T_s^{2-1/\beta}) \\ &\sim sT_s^{-s-1}t \asymp T_s^{-s-1}.\end{aligned}$$

(II) Following Theorem 3.4, for switched loss curve

$$\mathcal{E}_{\text{switch}}(T_s) - \mathcal{E}_{\text{switch}}(T_s+t) = (\mathcal{E}_{\text{small}}(T_s) - \mathcal{E}_{\text{small}}(T_s+t)) + h\sigma^2(\frac{1}{B_1} - \frac{1}{B_2})\int_0^t \mathcal{K}(r)\,\mathrm{d}r.$$

With the help of Lemma A.1, we have

$$\begin{aligned}\mathcal{E}_{\text{switch}}(T_s) - \mathcal{E}_{\text{switch}}(T_s+t) &= O(T_s^{-s-1}) + h\sigma^2(\frac{1}{B_1} - \frac{1}{B_2})\int_0^t \mathcal{K}(r)\,\mathrm{d}r \\ &\sim h\sigma^2(\frac{1}{B_1} - \frac{1}{B_2})\int_0^t \mathcal{K}(r)\,\mathrm{d}r \asymp 1.\end{aligned}$$

□

A.2 Proof for Theorem 4.2

Proof. Following Theorem 3.4,

$$\mathcal{E}_{\text{switch}}(T_s+t)-\mathcal{E}_{\text{large}}(T_s+t)=h\sigma^2(\frac{1}{B_1}-\frac{1}{B_2})\int_0^{T_s}\mathcal{K}(T_s+t-r)\,\mathrm{d}r.$$

By variable substitution, we have

$$\mathcal{E}_{\text{switch}}(T_s+t)-\mathcal{E}_{\text{large}}(T_s+t)=h\sigma^2(\frac{1}{B_1}-\frac{1}{B_2})\int_t^{t+T_s}\mathcal{K}(r)\,\mathrm{d}r.$$

Notice that as $t\to\infty$, by Lemma A.1,

$$r\to\infty\Rightarrow\mathcal{K}(r)\,\frac{\Gamma(\frac{2\beta-1}{\beta})}{(2r)^{\frac{2\beta-1}{\beta}}}\sim\frac{\Gamma(\frac{2\beta-1}{\beta})}{(2t)^{\frac{2\beta-1}{\beta}}}.$$

Hence

$$\mathcal{E}_{\text{switch}}(T_s+t)-\mathcal{E}_{\text{large}}(T_s+t)\sim C_\beta h\sigma^2(\frac{1}{B_1}-\frac{1}{B_2})T_s t^{\frac{1-2\beta}{\beta}}$$

□

A.3 Supplementary theorem for ratio of final merge

Theorem A.2 (Final merge ratio). *The gap ratio between the loss difference of b_{switch} and b_{large} with the loss difference of b_{small} and b_{large} is defined as*

$$f(t):=\frac{\mathcal{E}_{switch}(t)-\mathcal{E}_{large}(t)}{\mathcal{E}_{small}(t)-\mathcal{E}_{large}(t)}$$

We define $T_t^a=\inf\{t>0|f(T_s+t)\le a\}$ to be the time to reduce the relative gap to a.

(I) $f(t)$ is strictly increasing.

(II) As $T_s\to\infty$, there exists T^a such that $T_t^a=T_a-O(T_s^{\frac{1-\beta}{\beta}})\to T_a^-$.

(III) As $a\to0^+$, $T_a=O(a^{-\frac{\beta}{\beta-1}})$

Theorem A.2 provides a precise characterization of the gap ratio dynamics that govern the transition between small-batch, switched, and large-batch training curves. Once the switch occurs, the trajectory of b_{switch} monotonically approaches that of b_{large}, ensuring that the loss gap steadily closes. In particular, part (III) shows that in order to reduce the relative gap to a tolerance of a, one requires a time on the order of $-\beta/(\beta-1)$, which makes precise how the convergence slows as we demand tighter closeness. This scaling reveals that approaching near-perfect alignment of the switched and large-batch curves incurs rapidly growing time.

Proof. (I) Notice that by Theorem 3.4,

$$\mathcal{E}_{\text{small}}(t)-\mathcal{E}_{\text{switch}}(t)=h\sigma^2\int_0^{t-T_s}\mathcal{K}(r)\,\mathrm{d}r,\ \mathcal{E}_{\text{small}}(t)-\mathcal{E}_{\text{large}}(t)=h\sigma^2\int_0^{t}\mathcal{K}(r)\,\mathrm{d}r.$$

Hence, We define

$$f_{T_s}(t):=1-\frac{\mathcal{E}_{\text{small}}(T+t)-\mathcal{E}_{\text{switch}}(T+t)}{\mathcal{E}_{\text{small}}(T+t)-\mathcal{E}_{\text{large}}(T+t)}=\frac{\int_0^t\mathcal{K}(r)\,\mathrm{d}r}{\int_0^{t+T_s}\mathcal{K}(r)\,\mathrm{d}r}.$$

Then

$$\frac{\mathrm{d}}{\mathrm{d}t}f_{T_s}(t)=-\frac{\mathcal{K}(t)\mathcal{K}(t+T_s)}{(\int_0^{t+T_s}\mathcal{K}(r)\,\mathrm{d}r)^2}(\frac{\int_0^{t+T_s}\mathcal{K}(r)\,\mathrm{d}r}{\mathcal{K}(t+T_s)}-\frac{\int_0^t\mathcal{K}(r)\,\mathrm{d}r}{\mathcal{K}(t)}).$$

Let $p(t) := \frac{\int_0^t \mathcal{K}(r)\,\mathrm{d}r}{\mathcal{K}(t)}$. Then

$$p'(t) = \frac{\mathcal{K}(t)^2 - \int_0^t \mathcal{K}(r)\,\mathrm{d}r\mathcal{K}'(t)}{\mathcal{K}(t)^2} = 1 - \frac{\int_0^t \mathcal{K}(r)\,\mathrm{d}r\mathcal{K}'(t)}{\mathcal{K}(t)^2} > 1.$$

Hence p is strictly increasing, so $p(t+T_s) > p(t)$, therefore $f_{T_s}(t)$ is strictly decreasing. By $f_{T_s}(0) = 1$ and $f_{T_s}(\infty) = 0$, the existence of T_s^a is ensured.

(II)

We define

$$f_\infty(t) := \lim_{T_s \to \infty} f_{T_s}(t) = 1 - \frac{\int_0^t \mathcal{K}(r)\,\mathrm{d}r}{\int_0^\infty \mathcal{K}(r)\,\mathrm{d}r}.$$

f_∞ is continuous, strictly increasing, $f_\infty(0) = 1$, $f_\infty(\infty) = 0$. Hence for each $a \in (0,1)$, there exists a unique T_a such that $f_\infty(T_a) = a$.

We have

$$a\int_0^\infty \mathcal{K}(r)\,\mathrm{d}r \geq \int_0^{T_s^a} \mathcal{K}(r)\,\mathrm{d}r \geq a\int_0^{T_s} \mathcal{K}(r)\,\mathrm{d}r.$$

Let $\bar{T}_s^a$ and T_a be

$$\int_0^{T_a} \mathcal{K}(r) = a\int_0^\infty \mathcal{K}(r)\,\mathrm{d}r,\ \int_0^{\bar{T}_s^a} \mathcal{K}(r) = a\int_0^{T_s} \mathcal{K}(r)\,\mathrm{d}r.$$

Obviously by the monotonicity, we have $T_a \geq T_s^a \geq \bar{T}_s^a$. We can use squeeze theorem to estimate T_s^a . Note that as $T_s \to \infty$,

$$\int_0^{T_s} \mathcal{K}(r)\,\mathrm{d}r = \int_0^\infty \mathcal{K}(r)\,\mathrm{d}r - \frac{\Gamma(\frac{2\beta-1}{\beta})\beta}{2^{\frac{2\beta-1}{\beta}}(\beta-1)} T_s^{\frac{1-\beta}{\beta}} + o(T_s^{\frac{1-\beta}{\beta}}).$$

So

$$\bar{T}_s^a = T_a - \mathcal{K}^{-1}(T_a)\frac{\Gamma(\frac{2\beta-1}{\beta})\beta}{2(\beta-1)} T_s^{\frac{1-\beta}{\beta}} + o(T_s^{\frac{1-\beta}{\beta}}).$$

Hence

$$T_s^a = T_a - O(T_s^{1/\beta-1}) = T_a - O(T_s^{1/\beta-1}).$$

(III)

As $a \to 0^+$, we have $T_a \to \infty$. By Lemma A.1, we have

$$a = f_\infty(T_a) \sim \frac{\Gamma(\frac{2\beta-1}{\beta})}{2^{\frac{\beta-1}{\beta}}} T_a^{\frac{1-\beta}{\beta}} \Rightarrow T_a \sim a^{-\frac{\beta}{\beta-1}}.$$

□

A.4 Proof for Theorem 4.3

Following Theorem 3.4,

$$\frac{\mathrm{d}}{\mathrm{d}D_s}\mathcal{E}_{\text{switch}}(t_{D_s}) = (\frac{1}{B_1} - \frac{1}{B_2})\left[-st_c^{-s-1} + h\sigma^2(\frac{\mathcal{K}(t_{D_s})}{B_1} + \frac{\mathcal{K}((d-D_s)/B_2)}{B_2})\right].$$

We have $t_c \asymp D$, hence

$$-st_{D_s}^{s-1} \asymp -D^{-s-1},\ h\sigma^2\frac{\mathcal{K}(t_{D_s})}{B_1} \asymp D^{-l}.$$

Since $D^{-l} = o(D^{-s-1})$, to make

$$\frac{\mathrm{d}}{\mathrm{d}D_s}\mathcal{E}_{\text{switch}}(t_{D_s}) = 0.$$

We must have

$$\mathcal{K}((D - D_s)/B_2) \asymp -D^{-s-1} \Rightarrow \mathcal{K}((D - D_s)/B_2) \to \infty.$$

By monotonicity of $\mathcal{K}$, we get $D - D_s \to \infty$, combining with Lemma A.1, we have

$$(D - D_s)^{2-1/\beta} \asymp D^{-s-1}, (D - D_s) \asymp D^{\frac{s+1}{l}} = D^{\frac{s+1}{2-1/\beta}}.$$

A.5 Proof for Theorem 4.4

By Theorem 3.4, The optimal batch size schedule can be formalized as below problem

$$\underset{\substack{T, B(t)_{t=0}^{T} \\ s.t.\ \int_0^T B(t)\,\mathrm{d}t = D}}{\arg\min} \quad \mathcal{E}(B(t)) := \frac{1}{T^s} + \int_{t=0}^{T} \frac{\mathcal{K}(T-t)}{B_t}\,\mathrm{d}t.$$

It can be converted to

$$\underset{\substack{T, B(t)_{t=0}^{T} \\ \int_0^T B(t)\,\mathrm{d}t = D}}{\arg\min} \frac{1}{T^s} + \int_{t=0}^{T} \frac{\mathcal{K}(T-t)}{B_t}\,\mathrm{d}t = \underset{T}{\arg\min} \underset{\substack{B(t)_{t=0}^{T} \\ \int_0^T B(t)\,\mathrm{d}t = D}}{\arg\min} \frac{1}{T^s} + \int_{t=0}^{T} \frac{\mathcal{K}(T-t)}{B_t}\,\mathrm{d}t.$$

For the last $\arg\min$ with fixed k, by Cauchy Inequality, we have

$$(\int_{t=0}^{T} \frac{\mathcal{K}(T-t)}{B_t}\,\mathrm{d}t)(\int_{t=0}^{T} B(t)\,\mathrm{d}t) \geq (\int_{t=0}^{T} \mathcal{K}^{1/2}(T-t)\,\mathrm{d}t)^2.$$

Hence by the equal condition, in order to minimize $\mathcal{E}(B(t))$, we must have $B(t) \asymp (T-t)^{-l/2}$. By $\int_{t=0}^{T} B_t\,\mathrm{d}t = D$, we have $B(t) \asymp D\frac{(k-t)^{-l/2}}{k^{-l/2+1}}$, thus

$$\int_{t=0}^{T} \frac{\mathcal{K}(T-t)}{B_t} = \frac{(\int_{t=0}^{T} \mathcal{K}^{1/2}(T-t))^2\,\mathrm{d}t}{\int_{t=0}^{T} B_t\,\mathrm{d}t} \asymp \frac{T^{1/\beta}}{D}.$$

Thus we get

$$g(T) := \underset{\substack{T, B(t)_{t=0}^{T} \\ s.t.\ \int_0^T B(t)\,\mathrm{d}t = D}}{\arg\min} \frac{1}{T^s} + \int_{t=0}^{T} \frac{\mathcal{K}(T-t)}{B_t}\,\mathrm{d}t \asymp \frac{1}{T^s} + \frac{T^{1/\beta}}{D}.$$

To minimize $g(T)$, the optimal $T_\star$ satisfies

$$\frac{\mathrm{d}}{\mathrm{d}k}(T_\star^{-s} + \frac{T_\star^{1/\beta}}{D}) = 0 \Rightarrow -sT_\star^{-s-1} + \frac{T_\star^{\frac{1}{\beta}-1}}{\beta D} = 0 \Rightarrow T_\star \asymp D^{\frac{1}{1/\beta+s}}.$$

Hence the minimum $\mathcal{E}$ satisfy

$$\mathcal{E}^\star = T_\star^{-s} + \frac{T_\star^{1/\beta}}{D} \asymp D^{-\frac{s}{1/\beta+s}} = D^{-\frac{s}{1/\beta+s}} = D^{-\frac{s\beta}{1+s\beta}}.$$

The optimal batch size schedule satisfy

$$B_t \asymp \frac{(T-t)^{1/2\beta-1}}{T^{1/2\beta}} D.$$

B Experimental Details

Table 1: Model configurations

Type	LLaMA	LLaMA	LLaMA	MoE	MoE
Size	50M	200M	492M	1001M	1119M
Activated Size	—	—	—	209M	291M
d_{model}	512	1024	1280	512	576
d_{FF}	2048	4096	5120	1408	1152
$d_{\text{FF\_MoE}}$	—	—	—	1408	192
q_head	8	16	20	8	6
k_head	8	16	20	4	2
depth	4	8	15	12	24
n_expert	—	—	—	64	224
activated_expert	—	—	—	3	16

We performed experiments under two distinct settings to verify the consistency of the observed phenomena across scales.

Small-scale.

- **Model.** LLaMA (Touvron et al., 2023) is a dense, decoder-only Transformer architecture that integrates several modern design components, including Rotary Positional Encoding (RoPE) (Su et al., 2024), Swish-Gated Linear Units (SwiGLU), and Root Mean Square Layer Normalization (RMSNorm). We pre-train LLaMA models with parameter counts ranging from 50M to 492M. A full list of model configurations is provided in Table 1.
- **Dataset.** Colossal Clean Crawled Corpus (C4) (Raffel et al., 2020) is a large-scale, publicly available language dataset widely adopted for LLM pre-training, including models such as RoBERTa (Liu et al., 2019) and T5 (Raffel et al., 2020). For tokenization, we employ the T5 tokenizer with a vocabulary size of 32,100. Following the setup of Zhao et al. (2024); Zhu et al. (2025), we train with a sequence length of 256. We use 1,000 linear warm-up steps.

Large-scale.

- **Model.** Shortcut-connected Mixture of Experts (ScMoE) (Cai et al., 2025) is a novel MoE architecture that addresses communication overheads in expert parallelism by introducing shortcut connections and an overlapping parallelization strategy. ScMoE decouples the usual sequential dependency between communication (All-to-All operations among expert modules) and computation, enabling up to 100% overlap of those two processes. A full list of model configurations is provided in Table 1.
- **Dataset.** We train on a private real-world LLM dataset to ensure that our experiments closely reflect practical deployment scenarios. The tokenizer is configured with a vocabulary size of 131,072, and training is performed with a maximum sequence length of 8,192.

Optimizer. For both small-scale and large-scale experiments, we adopt the standard Adam optimizer with decoupled weight decay as the baseline. The baseline configuration follows protocols from LLaMA pre-training (Touvron et al., 2023), using hyperparameters $\beta_1 = 0.9$, $\beta_2 = 0.95$, weight decay $\lambda = 0.1$, and a gradient clipping threshold of 1.0.

B.1 Experimental details for Section 5.1

We conduct experiments with a 200M-parameter LLaMA model on 4B tokens with learning rate 1×10^{-3} using a two-stage batch size schedule, switching from 256 to 512 at different points in

training. The total data size corresponding to the full large batch size training step are 30000. We switch batch size at different ratio {0, 1/16, 2/16, 3/16, 4/16, 5/16, 6/16, 7/16, 8/16, 9/16, 10/16, 11/16, 12/16, 13/16, 14/16, 15/16, 16/16}. Each ratio is repeated multiple times to reduce variance in the results.

B.2 Experimental details for Section 5.2

- **492M** We conduct experiments with a 492M-parameter LLaMA model on 4B tokens with learning rate 5×10^{-4} using a two-stage batch size schedule, switching from 512 to 1024, 2048, 4096 at 0 and 25,000 step in training.
- **1001M** We conduct experiments with a 1001M-parameter MoE model on 0.4T tokens using a two-stage batch size schedule, switching from 640 to 1280, 2560 at 50B, 200B and 300B token in training. In addition, we evaluate a staged schedule that progressively increases the batch size—from 640 to 1280, then 1920, and finally 2560 at 100B, 150B and 200B token in training.
- **1119M** We conduct experiments with a 1119M-parameter MoE model on 1T tokens using a two-stage batch size schedule, switching from 1024 to 2048 at 300B and 600B token.

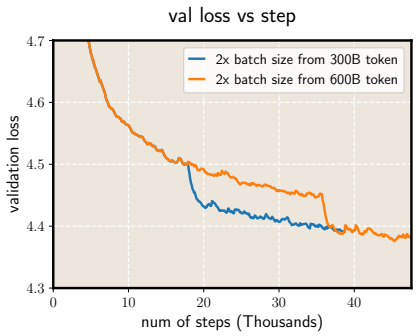

Figure 7: Validation loss versus training token under different batch-size switching times using 1119M parameter MoE model trained on 1T token.

B.3 Experimental details for Section 5.3

We conduct experiments with a 50M-parameter LLaMA model trained on the C4 dataset with learning rate 1×10^{-3} , using a small batch size of 128 and a large batch size of 256. The total data size corresponding to the full large batch size training step are {20000, 25000, 30000, 35000, 40000, 45000, 50000, 55000, 60000, 65000, 70000, 75000}. For each data size, we perform a grid search to determine the optimal switching point $D^\star$, with a precision of $D/32$. Each configuration of $D^\star/D$ is repeated multiple times to reduce variance in the results.

B.4 Schedule comparison

We compare multi-stage batch size scheduling strategies for 200M LLaMA model and 1119M parameter MoE model. For 1119M-parameter MoE model, we train on 1T tokens using a four-stage batch size schedule, switching from 1024 to 2048, then 3072 and finally 4096 at different time steps. For 200M-parameter LLaMA model, we train on 4B tokens using a four-stage batch size schedule, switching from 128 to 256, then finally 512 at different time steps.

In Figures 8 and Figure 9, the left panels show how batch size evolves with training tokens, while the right panels report the corresponding validation loss. Across both model scales, later switching consistently yields lower validation loss than earlier switching, validating the effective of late switch phenomenon in multi-stage batch size scheduling regime.

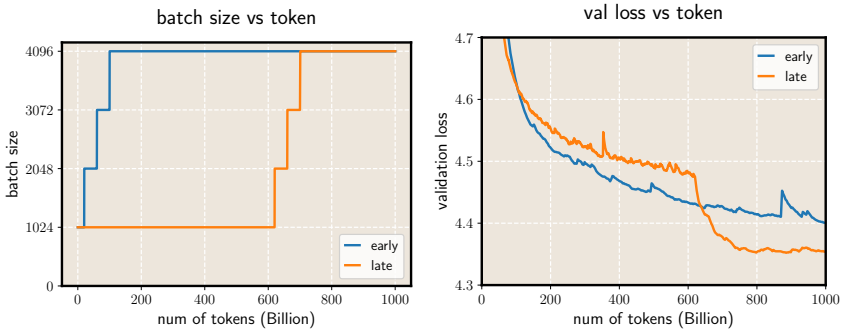

Figure 8: Validation loss versus training token with four-stage batch size schedule using 1119M parameter MoE model trained on 1T token. **Left:** batch size versus training tokens; (right) validation loss versus training tokens.

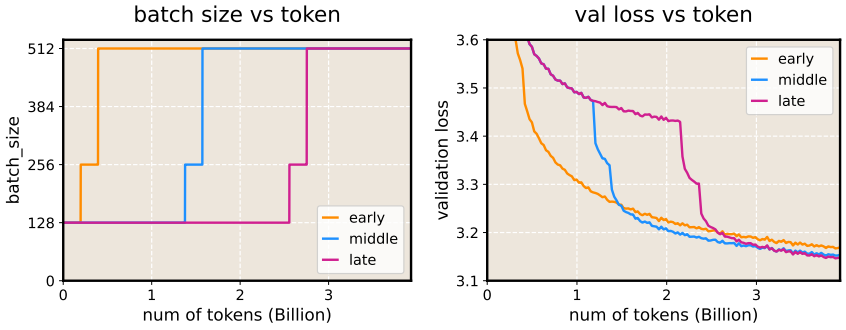

Figure 9: Validation loss versus training token with three-stage batch size schedule using 200M parameter LLaMA model trained on 4B token. **Left:** batch size versus training tokens; **Right:** validation loss versus training tokens.

C Statement

C.1 Ethics Statement

We have confirmd that this research was conducted in full compliance with the ICLR Code of Ethics. All experiments respect the principles of integrity, fairness, and transparency. No part of this work involves harm to humans, animals, or the environment, and we have taken care to ensure the responsible use of data, models, and computational resources.

C.2 Reproducibility Statement

We believe that all experimental results in this work are reproducible. The paper specifies comprehensive training and evaluation details—including hyperparameters, optimizer choices, and other relevant settings—in Section 5 and Appendix B. For small-scale experiments, we provide open-source code in the supplemental material, and all datasets used are publicly available. For large-scale experiments, we believe that employing comparable datasets and training pipelines will reproduce the same phenomena.

C.3 LLM usage statement

We used LLM as a writing assistant during paper preparation. The model found and corrected grammar mistakes throughout the manuscript. It suggested ways to make our sentences clearer and smoother. The LLM helped polish the language while keeping our meaning intact. We limited LLM use to only language editing tasks. All research content and ideas came entirely from human work.

Beyond serving as tools, LLMs were themselves the subject of our study. We trained these models and analyzed their behavior to uncover and explain novel phenomena. Importantly, this use of LLMs

as research objects should not be misinterpreted as a substantive contribution from the models to the work itself.