How Learning Rate Decay Wastes Your Best Data in Curriculum-Based LLM Pretraining

Anonymous authors
Paper under double-blind review

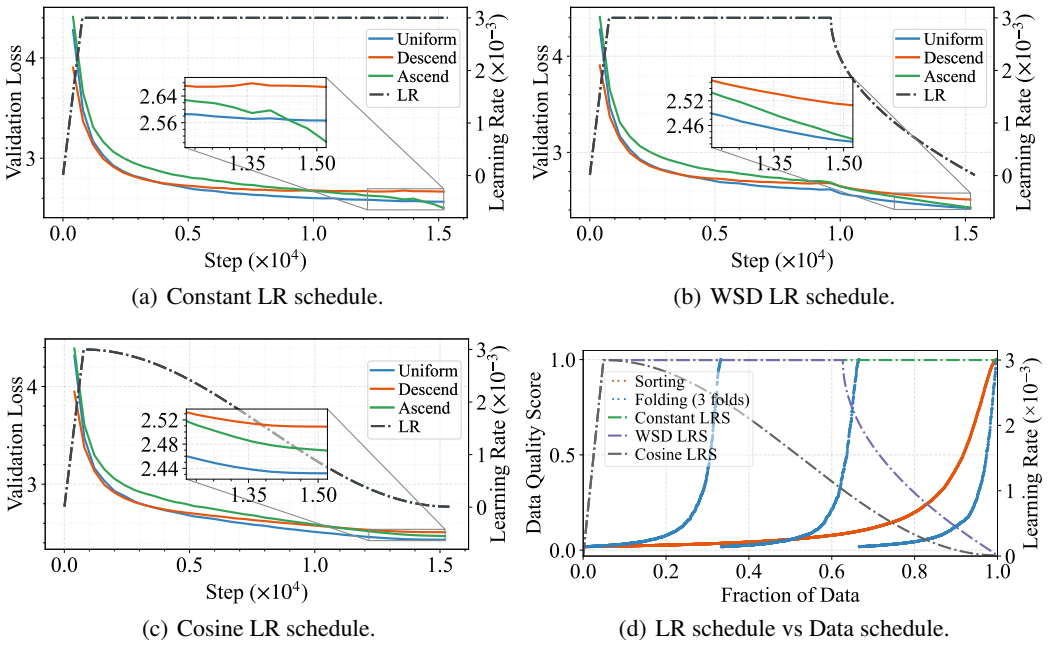
Abstract

Curriculum learning is a powerful paradigm, yet its application to large language model (LLM) pretraining remains underexplored, especially in scenarios where high-quality data is limited yet crucial. Previous works have primarily focused on applying curriculum learning to LLM pretraining by searching for better data quality metrics. However, these approaches have yielded only marginal gains, and curriculum-based training is still not a standard practice. In this work, we explore the problem from the opposite perspective: *if a good quality metric is available, can current curriculum learning strategies produce better results?* We diagnose a key, yet overlooked, factor responsible for this deficiency: the interplay between the data order and the learning rate (LR) schedule. We find that while curriculum learning can greatly outperform pretraining with a uniform data distribution under a constant LR schedule, this advantage diminishes as the learning rate decays. Building on this observation, we propose replacing LR decay with model averaging, which involves computing a weighted average of last several model checkpoints. We find this strategy achieves better results than standard LR decay schedules, especially in a mid-training regime where only a portion of high-quality data is available. Furthermore, this approach reveals that model averaging is greatly strengthened with the occurrence of curriculum learning. Finally, we propose a co-designed strategy for curriculum-based LLM pretraining: combining a moderate LR decay with model averaging. This approach allows the model to strike a balance between learning effectively from high-quality data, reducing knowledge forgetting, and mitigating gradient noise. We find that this combination highlights a previously overlooked opportunity to improve pretraining by co-designing the data curriculum, LR schedule, and model averaging.

1 Introduction

The quality and composition of pretraining data are critical for the performance and efficiency of large language models (LLMs) (Grattafiori et al., 2024; DeepSeek-AI et al., 2025; Yang et al., 2025; OpenAI et al., 2024). Researchers often enhance data quality using methods like rule-based filtering, quality scoring, and score-based selection (Su et al., 2025; Li et al., 2024; Penedo et al., 2025; 2024; Weber et al., 2024). However, they typically train on this curated data in a uniformly random order and weight all samples equally (Li et al., 2024; Penedo et al., 2025; OLMo et al., 2025). This standard approach ignores the fine-grained quality information available in the data scores.

A natural strategy to use this quality information is curriculum learning, where the model trains on data samples in increasing order of quality (Wettig et al., 2024; Dai et al., 2025; Wang et al., 2021). While curriculum learning traditionally refers to an easy-to-hard progression, we use the term here for a low-to-high quality ordering. One key motivation is to mitigate catastrophic forgetting (McCloskey & Cohen, 1989; Tirumala et al., 2022; Liao et al., 2025). By training on high-quality data later, the model can better retain valuable information. Recent works (Yang et al., 2025; DeepSeek-AI et al., 2025; Team et al., 2025; OLMo et al., 2025) have adopted coarse-grained curricula, such as adding high-quality domain data in a second training phase. However, fine-grained, instance-level curricula are not yet standard practice. Previous studies on this topic have shown limited improvements and offered little insight into why they work, which has prevented their broader adoption (Wettig et al., 2024; Dai et al., 2025; Zhang et al., 2025; Kim & Lee, 2024).



(a) Constant LR schedule. (b) WSD LR schedule.

(c) Cosine LR schedule. (d) LR schedule vs Data schedule.

Figure 1: Data curriculum reveals a diminishing benefit with LR decay. We show the validation loss for different data curricula under different LR schedules. The data schedules include Uniform, Ascending-Order, and Descending-Order by score ordering, and include Sorting and Folding (detailed in Section 4) by arrangement strategy. The LR schedules include constant, cosine, and WSD (Hu et al., 2024; Hägele et al., 2024) schedules. For validation loss, data curriculum wins over uniform under constant schedule. The benefit of high-quality data diminishes as the learning rate decays, like in cosine and WSD schedules.

This raises a key question: *even with a good quality metric, why does curriculum learning often fail to deliver significant gains?* In this paper, we identify a critical and overlooked factor: a detrimental coupling between the data schedule and the learning rate (LR) schedule. Standard pretraining uses LR schedules that decay over time, such as cosine decay (Loshchilov & Hutter, 2017) or warmup-stable-decay (WSD) (Hu et al., 2024). Near the end of training, the LR often drops to a very small value, mostly at scale of 10^{-5} and even close to zero (Li et al., 2025b; OLMo et al., 2025). In this case, high-quality data in data curriculum is processed with a greatly reduced learning rate near the end, as shown in Figure 1(d). This small LR effectively prevents the model from learning sufficiently from the most valuable data, which counteracts the curriculum's intended benefit.

To resolve this tension, we propose using model averaging (Li et al., 2025c; Izmailov et al., 2019; Tian et al., 2025) to decouple the data schedule from the LR schedule. Model averaging refers to computing the weighted average of last several checkpoints, typically using moving average (Li et al., 2025c). Model averaging reduces training noise and stabilizes final parameters, which performs a similar role as a decaying LR. Crucially, in curriculum-based pretraining, it allows the model to get rid of LR decay and keeps a peak learning rate, which allows model to take larger update along the gradient direction by high-quality data. We call this strategy **Curriculum Model Averaging (CMA)**, featuring a low-to-high quality data curriculum and constant LR schedule in training and using model average to obtain final model. Our experiments confirm that under a data curriculum, model averaging with a high LR outperforms a standard decaying LR. This result further validate the coupling of LR and data schedule can hurt the performance. Moreover, we find model average with a uniform data order may not match a standard LR decay schedule. These results highlight the importance of the robust and low-noise gradient signal from high-quality data, that allows model averaging to effectively manage noise variance.

Based on understanding of the intricate interplay between LR and data schedules, we propose a guideline in curriculum-based LLM pretraining: (1) use a moderate LR decay schedule (e.g., decay to $1/3$ of peak LR in WSD schedule); (2) use model average to obtain the final model. Through experiments on different LR decay extents, we confirm the benefits of the co-design of LR and data schedules with weight average strategy. Moreover, this combination sheds light on a previously overlooked opportunity to improve pretraining strategy. As shown in Figure 5, previous works (Dai et al., 2025; Li et al., 2025c) attempt to improve pretraining around an aggressive LR decay regime with uniform-data training. But in this regime, adding data curriculum or apply additional model

average can only produce marginal benefits, as shown in prior work (Zhang et al., 2025; Tian et al., 2025). In this work, we find that the benefit of data curriculum and weight average can only appear in a moderate LR decay regime, which is non-optimal in uniform-data training and still under-explored.

We conduct extensive experiments to validate our hypothesis and the effectiveness of our strategies. We train a 1.5B parameter model on 30B tokens and test on datasets with and without model-based quality filtering. In summary, our contributions are threefold:

- We identify and analyze the overlooked coupling between data schedules and learning rate schedules, which explains the limited success of prior work on quality-based curriculum learning for LLM pretraining.
- We propose Curriculum Model Averaging (CMA), a novel strategy that combines a quality-based curriculum with model averaging to resolve the coupling issue, and interpret the synergy relation between model average and data curriculum.
- We propose to co-design LR and data schedules, with weight average, exhibiting an under-explored opportunity for improving data efficiency in LLM pretraining. We also design a theoretical demonstration to show their interplay.

2 Related Work

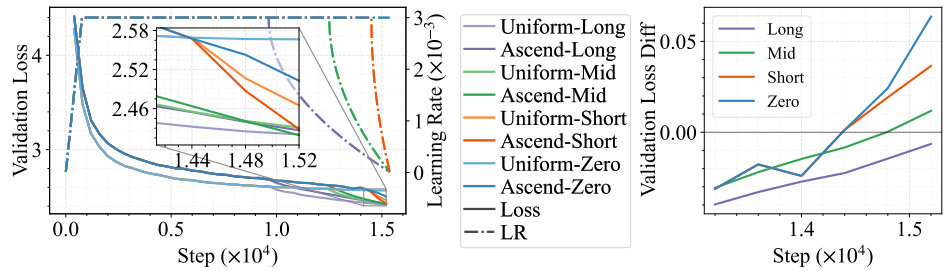
The additional related work discussion about curriculum learning, learning rate schedules, and model averaging can refer to Section B. A detailed discussion on prior work for curriculum learning in LLM pretraining can see Section C.

Curriculum Learning in LLM Pretraining. Currently, instance-level curricula have produced only negligible improvements and have not been validated at a sufficient scale (Dai et al., 2025; Zhang et al., 2025; Wettig et al., 2024; Campos, 2021). Wettig et al. (2024) proposes to sort data with LLM-annotated scores but reports a limited improvement and shows benefits from both ascending and descending quality orders, lacking a clear interpretation on the underlying mechanisms. Campos (2021); Kim & Lee (2024) lacks validation experiments for curriculum benefits. Zhang et al. (2025); Dai et al. (2025) finds very marginal improvement of vanilla data curriculum, and propose folding (Section 4) or interleaved curricula to sort data within consecutive stages. However, as shown in Section C, the benefit of folding shows only on a smaller scale experiments with a low LR, and the benefit diminishes and even get worse in a scaled-up and high LR regime.

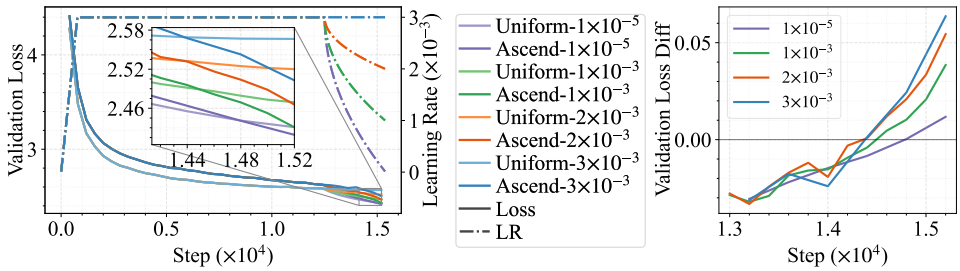
3 Preliminary

Learning Rate Schedule. We consider two primary types of learning rate (LR) schedules apart from the constant LR schedule. The commonly used cosine schedule defines the LR at step t as $\eta(t) = \eta_0 \left(\frac{1+\alpha}{2} + \frac{1-\alpha}{2}\cos\left(\frac{\pi t}{T}\right)\right)$, where η_0 is the peak LR, T is the total number of training steps, and α is the ratio of the final LR to the peak LR. A more recent alternative is the Warmup-Stable-Decay (WSD) schedule, which consists of a linear warmup phase, a stable phase with a constant LR η_0, and a final decay phase. We use the 1-sqrt decay function, $f(t) = \eta_0\left(1 - \sqrt{r(t)}\right) + \eta_T\sqrt{r(t)}$, where $r(t) = \frac{t - t_{\text{decay}}}{T - t_{\text{decay}}}$ represents the progress through the decay phase, which starts at step t_{decay}. The choice details are discussed in Section A.

Model Averaging. Model averaging computes a weighted average of several model checkpoints to produce a single, final model. We consider three common strategies. Suppose we have N checkpoints, $M_1, \ldots, M_N$, typically the last N checkpoints collected in a fixed interval, at steps $t_1, \ldots, t_N$. **Simple Moving Average (SMA)** (Izmailov et al., 2019) applies a uniform weight to each checkpoint: $M_{\text{avg}} = \frac{1}{N}\sum_{i=1}^{N} M_i$. **Exponential Moving Average (EMA)** assigns exponentially decaying weights, giving more importance to recent checkpoints. It is defined recursively: $M_{\text{avg}}^{(i)} = \alpha M_i + (1-\alpha)M_{\text{avg}}^{(i-1)}$, with $M_{\text{avg}}^{(1)} = M_1$. The hyperparameter $\alpha \in (0, 1]$ controls the decay rate; a larger α places more weight on the most recent checkpoint. **Weighted Moving Average (WMA)** uses a predefined set of normalized weights $w_1, \ldots, w_N$ (where $\sum w_i = 1$) to compute the final model as $M_{\text{avg}} = \sum_{i=1}^{N} w_i M_i$. Prior work (Tian et al., 2025) proposes to derive the weights from gradient decay schedule, $w_i \propto \eta(t_i) - \eta(t_{i+1})$ and $w_N \propto \eta(t_N)$.



(a) Validation loss for different decay steps. (b) $\mathcal{L}_{\text{Uniform}} - \mathcal{L}_{\text{Ascend}}$



(c) Validation loss for different ending LR. (d) $\mathcal{L}_{\text{Uniform}} - \mathcal{L}_{\text{Ascend}}$

Figure 2: The benefit of a data curriculum diminishes with more aggressive LR decay. The data curriculum benefit is measured by the validation loss difference between uniform and ascending data order. *Long*, *Mid*, *Short* refer to WSD schedules with decay phases covering approximately 37%, 18%, 6% of training, respectively. *Zero* corresponds to a constant LR schedule. The plots on the right show the benefit ($\mathcal{L}_{\text{Uniform}} - \mathcal{L}_{\text{Ascend}}$) near the training end. It reveals the curriculum's benefit decrease as the LR decays more.

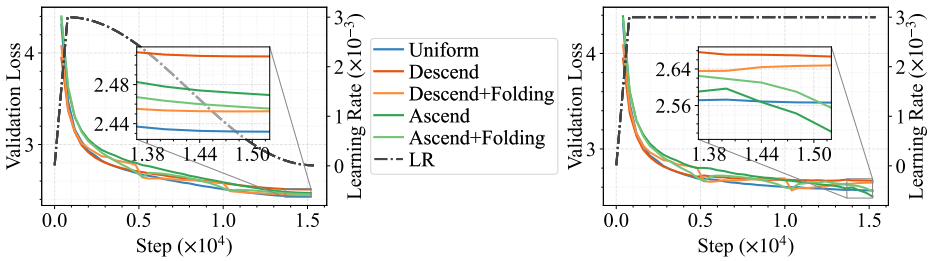
4 Coupling between Learning Rate and Data Schedules

In this section, we analyze the critical yet often overlooked interaction between the learning rate (LR) schedule and the data schedule. We first explain how the learning rate acts as an implicit importance weight for each data sample. We then present empirical results to demonstrate three key points: (1) a data curriculum can yield significant benefits over uniform data under a constant LR schedule; (2) these benefits diminish when a conventional decaying LR schedule is applied, particularly during the final, high-quality data regime; and (3) while adjustments to the curriculum can mitigate this issue, the underlying conflict persists.

Analysis on Coupling between Learning Rate and Data Schedules. A key insight is that the learning rate schedule acts as an implicit importance weight for each training sample. The parameter update at training step t is $\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \eta_t \boldsymbol{g}_t$, where η_t is the learning rate. The gradient $\boldsymbol{g}_t$ can be decomposed into a signal component, $\mathbb{E}[\boldsymbol{g}_t]$, which points in the direction of steady improvement, and a noise component, ϵ_t. A decaying learning rate η_t serves two purposes: it reduces the noise ϵ_t to stabilize training, but it also shrinks the update step taken in the signal direction $\mathbb{E}[\boldsymbol{g}_t]$. While modern optimizers like Adam (Kingma & Ba, 2017) use more complex update rules, the learning rate remains a dominant factor in the update magnitude. This dual role of η_t creates a fundamental conflict in quality-based curricula. High-quality samples are intentionally processed at the end of training, but this is precisely when conventional LR schedules reduce η_t to its minimum. Consequently, the decaying learning rate will diminishes the influence of the most valuable data, counteracting the intended benefit of the curriculum.

Experiment Settings. Our experiments refer to the DataComps-LM (DCLM) framework (Li et al., 2024) at `1B-1x` scale. We adopt the Qwen2.5-1.5B model architecture (Qwen et al., 2025) and train models on a 30B token subset of the DCLM-Baseline dataset (Li et al., 2024). We use the DCLM fasttext scores as the quality metrics for data curriculum. We set peak LR at 3×10^{-3} and ending LR at 1×10^{-5}, aligning with optimal settings found in prior work (Li et al., 2024; Luo et al., 2025; Li et al., 2025b). We choose a high-quality subset of DCLM-Baseline as validation set.

A Data Curriculum is Highly Effective with a Constant Learning Rate. To isolate the effect of the data schedule from the LR schedule, we first conducted experiments using a constant learning rate of 3×10^{-3}. We compared three data schedules: a uniform random baseline, a data curriculum, and a reverse data curriculum (in high-to-low ordering), both sorted by DCLM quality scores. We measure the training results by validation loss of high-quality dataset. As shown in Figure 1(a), the



(a) Cosine LR schedule. (b) Constant LR schedule.

Figure 3: LR decay affects the data utility and thus the validation losses of different data schedules. *folding* strategy divides the data into three chunks and sorts the data within each chunk, with details in Section 4. The score distribution of *Ascend+Folding* is in Figure 1(d). In cosine LR schedule, we find $\mathcal{L}_{\text{Descend}} > \mathcal{L}_{\text{Ascend}} > \mathcal{L}_{\text{Ascend+Folding}} > \mathcal{L}_{\text{Descend+Folding}} > \mathcal{L}_{\text{Uniform}}$, while in constant schedule, $\mathcal{L}_{\text{Descend}} > \mathcal{L}_{\text{Descend+Folding}} > \mathcal{L}_{\text{Uniform}} > \mathcal{L}_{\text{Ascend+Folding}} > \mathcal{L}_{\text{Ascend}}$, the ascending-order curriculum wins back, and all-sorting curriculum benefits more.

data curriculum significantly outperforms the uniform baseline, achieving a much lower validation loss and faster convergence. In contrast, the reverse data curriculum's validation loss trends upward, likely because the data distribution shifts progressively further from the high-quality validation set. These results clearly demonstrate that a quality-based curriculum is effective when its impact is not confounded by a decaying learning rate. We observed similar validation loss trends when using PreSelect scores (Shum et al., 2025) (see Appendix Figure 8(a)).

The Curriculum's Advantage Diminishes with a Decaying LR Schedule. In contrast to the constant LR experiments, the advantage of data curriculum largely diminishes when we use a WSD schedule (Figure 1(b)). Moreover, as shown in Figure 1(c), the data curriculum even falls back more in the cosine schedule, which decays throughout the range. To further test this relationship, we varied the aggressiveness of the LR decay by adjusting two WSD parameters: the number of decay steps and the final learning rate. The results in Figure 2 show a clear trend. As the decay phase becomes longer and more aggressive, the performance benefit of the data curriculum over the uniform baseline shrinks, eventually becoming negligible. This confirms the tight coupling between the data curriculum and the LR schedule. The learning rate decay undermines the contribution of high-quality in a data curriculum.

Exploring an Alternative Curriculum: Data Folding. We also investigated whether a different curriculum design could mitigate the coupling effect. We tested a *folding* curriculum, inspired by prior work (Dai et al., 2025; Zhang et al., 2025), where the dataset is split into several chunks and each chunk is sorted internally (see Figure 1(d)). The stage-wise design distributes high-quality data more evenly across the training process than sorting, bridging sorting and uniform schedules. As shown in Figure 3(a), under a cosine schedule, the ascending-folding strategy performed better than a simple end-to-end ascending sort but still underperformed the uniform baseline. As a comparison, under a constant schedule, ascending-all-together schedule greatly outperforms the uniform, and the ascending-folding results in a much limited improvement. These results further confirm our hypothesis. Under a constant LR schedule, data schedule with higher density of high-quality data outperform that with a lower density near the end. But when bearing lasting LR decay, uniform baseline can distribute high-quality data evenly in high peak LR regime, and the folding strategy can put more high-quality data in high LR regime than sorting, which may contribute to their better utilization of high-quality data in cosine schedule, and thus better results. A similar argument holds for descending-folding schedule under cosine schedule.

5 Decouple and Co-Design Schedules with Model Average

To resolve the coupling dilemma between the data schedule and the LR schedule, we turn to model average (Izmailov et al., 2019; Li et al., 2025c; Tian et al., 2025). We first investigate replacing LR decay entirely with model average, which allows high-quality data to be processed with a constant learning rate. While model average alone may not match the performance of LR decay with uniform data, we find that combining a data curriculum with model average produces comparable or even superior results to a standard LR decay schedule both with and without a curriculum. Furthermore, we find that combining model average and moderate LR decay can yield even stronger results for curriculum-based pretraining, especially in a mid-training setting where high-quality data is particularly sparse. Our results explore a previously unexplored regime for improving LLM pretraining strategy, exhibiting the great potential of co-designing LR and data schedules, as well as model average strategies.

Table 1: Curriculum Model Average (CMA) exhibits advantages over standard LR decay schedule pretraining, much better than widely used *Cosine+Uniform* setting. **WA**: Weight Average technique (Section 3). **Order**: Data ordering. **LRS**: Learning Rate Schedule (WSD: Warmup-Stable-Decay (to 1×10^{-5}), WSMD: Warmup-Stable-Moderate Decay (to 1×10^{-3}), Cos: Cosine, Const: Constant). **Core**: Average score on the first four, high signal-to-noise tasks according to prior work (Heineman et al., 2025) (MMLU, ARC-c, ARC-e, CSQA). Both the Core and Avg. scores are annotated with a subscript indicating the performance change relative to the baseline (*WSD + Uniform*). Performance changes are color-coded: **bold green** (≥ 0.5 improvement), light green (> 0 improvement), and red (decrease). Our proposed methods are highlighted in gray.

WA	Order	LRS	MMLU	ARC-c	ARC-e	CSQA	Core	OBQA	PIQA	SIQA	Wino.	Avg.
✗	Uniform	Cos	30.49	38.13	59.47	49.14	44.31$_{-1.90}$	42.20	71.87	45.19	56.51	49.13$_{-1.43}$
✗	Ascend	Cos	30.80	39.80	59.12	51.27	45.25$_{-0.96}$	42.60	71.55	45.65	57.06	49.73$_{-0.83}$
✗	Uniform	WSD	30.77	42.14	61.05	50.86	46.21	45.20	72.42	45.75	56.27	50.56
✗	Ascend	WSD	31.58	38.80	61.05	50.37	45.45$_{-0.76}$	45.80	71.82	46.01	57.30	50.34$_{-0.22}$
WMA	Uniform	Const	30.87	37.12	58.95	53.24	45.04$_{-1.17}$	43.40	71.76	46.26	57.38	49.87$_{-0.69}$
SMA	Uniform	Const	31.22	36.12	59.82	53.97	45.28$_{-0.93}$	43.40	71.98	46.42	57.85	50.10$_{-0.46}$
EMA	Uniform	Const	31.39	36.45	59.82	53.48	45.29$_{-0.92}$	42.40	72.14	46.32	57.54	49.94$_{-0.62}$
WMA	Ascend	Const	31.67	39.80	61.40	53.07	46.49$_{+0.28}$	45.00	71.93	45.45	57.14	50.68$_{+0.12}$
SMA	Ascend	Const	32.28	40.80	62.11	52.91	47.02$_{\mathbf{+0.81}}$	44.80	71.60	45.80	57.22	50.94$_{+0.38}$
EMA	Ascend	Const	32.17	40.80	61.75	53.07	46.95$_{\mathbf{+0.74}}$	44.80	71.55	45.85	57.62	50.95$_{+0.39}$

5.1 Model Average can Help Data Curriculum

CMA: Replacing Learning Rate Decay with Model Average. To address the coupling between data and LR schedules, we propose to decouple the data schedule from the side effects of LR annealing by replacing LR decay with model average. During the training process, we replace the decaying LR schedule with a constant LR and perform model averaging to compute a weighted average weights of last several checkpoints of training process. We call this strategy **C**urriculum **M**odel **A**veraging (CMA), detailed in Algorithm 1. In default setting, $\alpha = 0.2$ and average over the last 6 checkpoints, typically 0.2B interval with 30B tokens in total. Weight average strategies include Simple Moving Average (SMA), Exponential Moving Average (EMA) and Weighted Moving Average (WMA) introduced in Section 3. The standard LR schedule pretraining includes the most widely used cosine schedules and recent emerging WSD schedules (introduced in Section 3). Best result of both LR schedules and both data schedules serves as our evaluation baseline (*WSD + Uniform*). The downstream task performances are reported in Table 1.

The Synergy of Data Curriculum and Model Average. The results in Table 1 lead to several key observations: (1) **The combination is more effective than its parts.** The combination of a data curriculum and model average (e.g., *EMA + Ascend*) outperforms models trained with a standard LR decay schedule (e.g., *WSD schedule + Uniform*, or *WSD schedule + Ascend*). It also consistently outperforms model average on models trained with a uniform data order (e.g., *EMA + Uniform*). The model average strategy shows a comparable and even better results than the standard LR decay practice, wins over the traditional cosine schedule training paradigm by a great margin. (2) **The synergy is important: other combinations produce only limited improvement.** Model averaging with a uniform data order (e.g., *EMA + Uniform*) under a constant learning rate, does not fully match to a standard LR decay schedule (e.g., *WSD schedule + Uniform*). In addition, combining a standard LR decay with a data curriculum yields only limited gains and can even get worse (*WSD schedule + Ascend*), confirming the detrimental coupling we identified. The results reveal the necessity to combine both data curriculum and weight average in LLM pretraining, which is largely ignored by prior work. They either focus on weight average side (Tian et al., 2025; Yang et al., 2024), or focus on curriculum side (Dai et al., 2025). (3) **Aligning checkpoint weights with the data schedule is beneficial.** EMA puts more weights on later checkpoints and SMA puts weight evenly while WMA assigns decreasing weights to later (and thus higher-quality) checkpoints. Under data curriculum, EMA and SMA outperform WMA overall, revealing benefits of non-decreasing checkpoint weights.

Interpretation: Synergy Relation Comes from Decoupling LR Schedules from Data Schedules. We try to explain the synergy of data curriculum and model average from the loss landscape view. We focus on the intricate interplay of two factors: (1) **Learning Rate**: The learning rate determines update step size and affects the noise level, with a lower LR leading to lower noise. (2) **Data Quality**: The quality of data influences the gradient's direction and variance, with high-quality data providing a better signal-to-noise ratio. Weight average may not be able to reduce noise at the same level as an aggressive LR decay, thus may not reduce noise sufficiently to match LR decay under a uniform data ordering. Under a curriculum, the high-quality data presented at training end offers a better gradient direction. Hence, weight average can update more along signal direction than aggressive LR decay and bother less about noise under a data curriculum. It is possible for model

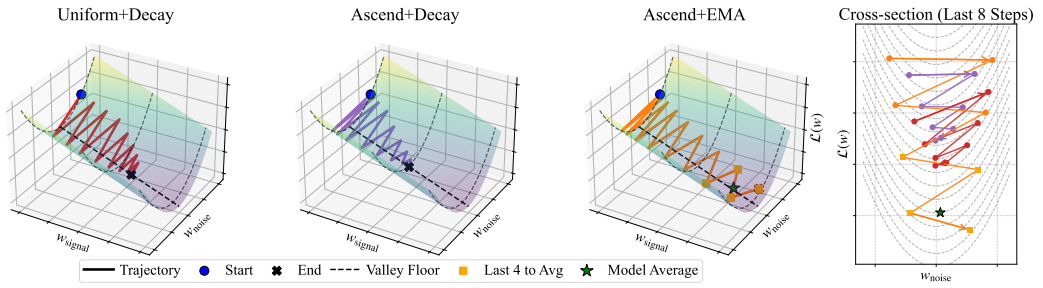


Figure 4: An interpretation of interplay between data ordering and LR schedules. We assume the gradient update can be decomposed as a signal direction and a noise direction. The data quality variation is interpreted as the signal-to-noise ratio. *Uniform+Decay* features consistent progress direction and final decay can reduce the noise; *Ascending+Decay* starts with much noise, but the step-size decays too fast to utilize the good signal by high-quality data near the end; *Ascending+EMA* also starts with great noise, but it keeps forward and can take advantage of good signal near the end, and use model average to sufficiently reduce noise. The cross sections of last 8 steps for these cases are integrated and presented on the right side.

average to reach a better balance between progress and noise reduction than LR decay under a data curriculum. A visualization of our interpretation is presented in Figure 4, and we also propose a simple theoretical model to make the interpretation more clear and tractable in Section 6.

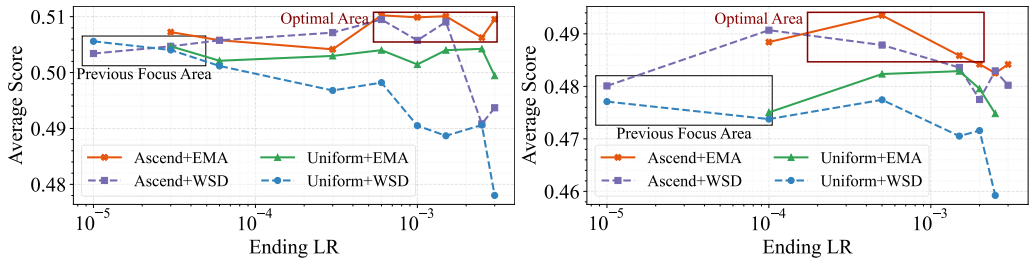
5.2 Results on Mid-Training with Mixed Quality Data

Table 2: The benefit of CMA becomes more prominent in mid-training setting. **WA**: Weight Averaging technique (Section 3). **Order**: Data ordering in two phases (U: Uniform, A: Ascend). A-T (All-Together) sorts data samples in both phases as a whole. **LRS**: Learning Rate Schedule (WSD: decay to 1×10^{-5}, Const: Constant LR). **Core**: Average score on the first four, high signal-to-noise tasks (MMLU, ARC-c, ARC-e, CSQA). Both the Core and Avg. scores are annotated with a subscript indicating the performance change relative to the baseline (*WSD + U,U*). Performance changes are color-coded: **bold green** (≥ 0.5 improvement), light green (> 0 improvement), and red (decrease). Our proposed methods are highlighted in gray.

WA	Order	LRS	MMLU	ARC-c	ARC-e	CSQA	Core	OBQA	PIQA	SIQA	Wino.	Avg.
✗	U,U	WSD	29.23	33.78	53.86	49.55	41.61	40.40	71.87	44.78	56.43	47.49
✗	U,A	WSD	29.44	34.45	52.63	50.12	$41.66_{+0.05}$	41.00	71.76	44.42	56.75	$47.57_{+0.08}$
✗	A,A	WSD	30.22	33.11	56.84	47.34	$41.88_{+0.27}$	39.40	71.55	44.78	56.67	$47.49_{0.00}$
✗	A-T	WSD	29.93	37.12	54.39	49.47	$42.73_{\mathbf{+1.12}}$	39.00	72.20	45.14	56.83	$48.01_{\mathbf{+0.52}}$
EMA	U,U	Const	29.84	32.78	52.28	51.52	$41.60_{-0.01}$	42.00	71.60	44.68	56.99	$47.71_{+0.22}$
EMA	U,A	Const	29.75	35.12	51.75	48.57	$41.30_{-0.31}$	42.20	70.51	44.83	56.91	$47.45_{-0.04}$
EMA	A,A	Const	30.31	36.45	57.54	50.12	$43.61_{\mathbf{+2.00}}$	41.40	72.14	45.09	56.43	$48.69_{\mathbf{+1.20}}$
EMA	A-T	Const	30.81	36.29	57.89	50.29	$43.82_{\mathbf{+2.21}}$	44.50	70.62	44.68	54.46	$48.69_{\mathbf{+1.20}}$
SMA	A-T	Const	30.65	36.79	57.37	50.78	$43.90_{\mathbf{+2.29}}$	43.60	70.89	44.73	54.74	$48.69_{\mathbf{+1.20}}$

CMA Helps More in Mid-Training. Mid-training is a recently emerging practice in LLM pretraining (Yang et al., 2025; OLMo et al., 2025; Hu et al., 2024). Mid-training uses average-quality data in the stable phase and incorporates high-quality data in the decay stage of WSD schedule. This experiment serves as a more practical setting: where most pretraining data is limited, while a small portion of data is of high quality. The experiment settings are detailed in Section A. As shown in Table 2, CMA exhibits a larger benefit margin in the mid-training setting than the experiments over high-quality data (Detailed in Section 5.1). The CMA results (e.g., *EMA + A-T*) shows advantages over WSD schedule results (e.g, *WSD + A-T*, or *WSD + U,U*) on both Core scores or average scores in Table 2 by a margin. The margin is surprising given there is no filtering or other fantastic processing on data. An explanation of increasing benefit is that, the high-quality is sparse but can offer a more valuable direction for parameter update in this setting.

A Practical and Simplified Strategy also Works Well. In practice, it may not be feasible to sort the whole data corpus globally according to a unified quality metric. As an alternative, we can shuffle data in each phase in ascending order separately (*A,A* in Table 2). The benefits of our approach over LR decay mostly persist. But only applying data curriculum over the high-quality regime is not enough for better results (e.g, *EMA + U,A*). Forgetting of some extremely low-quality data in the first phase can contribute to *A,A* benefits over *U,A*. Moreover, model average with uniform data and data curriculum with LR decay both produce a relatively marginal improvement compared to their combination. This result further confirms the synergy between model average and data curriculum.



(a) DCLM. (b) Mid-Training (WebOrganizer-DCLM).

Figure 5: Combined performance comparison of different strategies on average scores. We identify an underexplored Optimal Area through a combination of moderate LR decay and weight average with curriculum learning. *EMA* compute the EMA of last 6 checkpoints from WSD training.

5.3 Overlooked Benefit: Co-Design of Data Curriculum, LR Schedule, and Weight Average

CDMA: CMA with Moderate LR Decay Provides an Additional Optimization Opportunity. A natural question is whether combining a moderate LR decay with model averaging under an ascending-order data curriculum can yield further improvements. We conduct series of experiments to ablate ending learning rates, when fixing decay steps of WSD schedules. The ending learning rates range from 3×10^{-5} to 2.5×10^{-3}. Then for each run, we compute EMA of the last several checkpoints to get the final results. This process is a combination of LR decay and CMA. As shown in Figure 5, the combination can achieve a stable and optimal results with a moderate LR-decay (much higher ending LR than standard practice). Data curriculum with a moderate LR decay can also perform a close-to-optimal results, but can not fully match, especially when ending LR is close to the peak LR. In contrast, the strategies of LR decay and weight average on uniform data, can not match data curriculum results. This result motivate the following guideline for curriculum-based pretraining: **use a moderate LR decay** (e.g., decay to $1/3$ of a tuned peak LR), and **adopt model average** over last several checkpoints. To tell from CMA, we call this strategy Curriculum with LR Decay Model Averaging (CDMA). From the landscape view, we hypothesize that this combination can strike a better balance between noise reduction, maintaining a sufficient update magnitude and memorizing more knowledge.

Discussion: *Why is the Combination Under-Explored?* The CDMA strategy is straightforward and easy-to-implement. Tuning ending LR can also improve the result. This raises the question: *why is the combination under-explored?* A possible explanation is that prior works focus on an aggressive LR decay regime which stuck the discovery of a more satisfying pretraining approach. As show in Figure 5, this regime achieves best results for decay-only strategy under the uniform data scenario and there is a clean trend favoring a close-to-zero ending LR. This observation aligns with previous work (Li et al., 2025b). However, a best regime for uniform data may not be the best for other settings. For example, previous works (Zhang et al., 2025) focus on curriculum design around this regime, and thus may deduce a marginal or disappointing result about curriculum-based LLM pretraining. Moreover, prior work (Tian et al., 2025) suggests that LR decay and model averaging are mutually exclusive under a standard LR schedule, which aligns with our results in this regime. But in a moderate LR decay regime, it is probably beneficial to introduce weight average. Currently we only ablate on ending LR and fix others. This suggests an under-explored optimization space of pretraining involving LR schedules, data curricula, and model averaging strategies. More extensive experiments and the design of more sophisticated strategies are promising directions for future work.

6 A Theoretical Demonstration Sketch

As we reported and discussed above, the benefit of curriculum learning emerges when we apply a weight averaging manner or moderate LR decay instead of a LR schedule with excess decaying, such as Cosine or WSD schedule in practical pretraining. In the following, we present a simple theoretical model that recovers the above empirical insight. A full theoretical demonstration is shown in Section D. The main proof of this section can be found in Section E.

Problem Setup. We consider a quadratic loss function $\mathcal{L}(\boldsymbol{w}) = \frac{1}{2}\|\boldsymbol{w} - \boldsymbol{w}^*\|_2^2$, where $\boldsymbol{w} = (w_1, w_2) \in \mathbb{R}^2$ represents the trainable parameter, and $\boldsymbol{w}^*$ denotes the ground truth, which is set to $(0, 0)$. We use Stochastic Gradient Descent (SGD) to optimize. We denote the one-sample loss for the t-th iteration as $\ell_t(\boldsymbol{w}) := \|\boldsymbol{w} - \boldsymbol{x}_t\|_2^2$, where data point $\boldsymbol{x}_t$ is from some given dataset

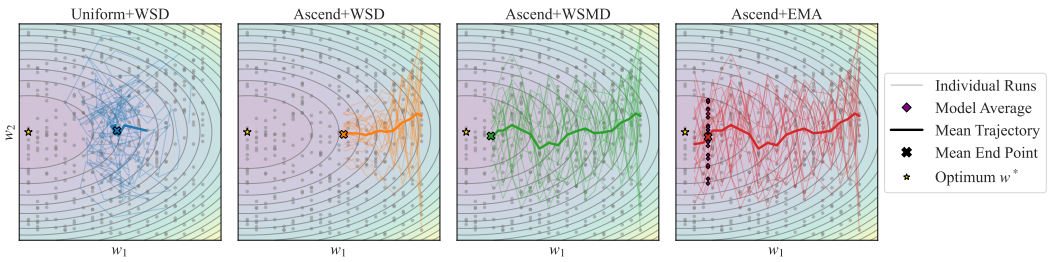


Figure 6: Visualization of the simulation experiments of theoretical example. The mean trajectory is averaged over $R = 20$ runs. The yellow star marks the global optimal, and w_1 represents a signal direction and w_2 represents a noise direction. The data samples are distributed evenly along the signal direction and randomly locate along noise direction. *Ascend+WSMD* and *Ascend+EMA* wins by sufficient progress along signal direction; *Uniform+WSD* fails for inconsistent signal and thus large variance along signal direction; *Ascend+WSD* fails for early-decay, resulting in insufficient update along the signal direction.

$\mathcal{D} = \{\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}, \ldots, \boldsymbol{x}^{(M)}\}$. Thus, the SGD update rule is $\boldsymbol{w}_t = \boldsymbol{w}_{t-1} - \eta_t \nabla \ell_t(\boldsymbol{w}_{t-1})$, where the η_t denotes the learning rate in the t-th iteration and $\boldsymbol{w}_t = (w_t^{(1)}, w_t^{(2)})$. The initial parameter $\boldsymbol{w}_0 = (Md, 0)$. We denote learning rate schedule by $E := \{\eta_1, \eta_2, \ldots, \eta_M\}$. $\mathcal{W}_{M;E}$ is the distribution of $\boldsymbol{w}_M$. The randomness within $\boldsymbol{w}_M$ comes from the random draw of the distribution in SGD. The expected loss is $\bar{\mathcal{L}}(M; E) := \mathbb{E}_{\boldsymbol{w} \sim \mathcal{W}_{M;E}}[\mathcal{L}(\boldsymbol{w})]$.

Then considering a training dataset $\mathcal{D}$, which consists of M different data points with varying data qualities. Data point $\boldsymbol{x}^{(i)} = (x_1^{(i)}, x_2^{(i)})$ satisfy that $x_1^{(i)} = (i-1)d$ and $x_2^{(i)} \sim \mathrm{Uniform}(-L, L)$, where $d = L/M$. $x^{(i)}$ provides signal in first dimension and introduce noise in the second dimension. Next, we consider two sampling strategies for each iteration of SGD: (1) sample one data point uniformly from $\mathrm{Uniform}(\mathcal{D})$; (2) sample one data point from $\mathcal{D}$ in an ascending order. Then in t-th iteration, $\boldsymbol{x}_t = \boldsymbol{x}^{(M-t+1)} \in \mathcal{D}$. See Figure 6 for visualization of optimization trajectories in simulation experiment.

Uniform Sampling + Learning Rate Schedule. SGD acts as an exponential averaging of the current parameter and the sampled data point. For uniform sampling, the parameter would approximately oscillate from 0 to $(m-1)d$ with a large variance along x-axis and the expected loss for uniformly sampling SGD has a lower bound

$$\min_E \bar{\mathcal{L}}(M; E) = \Omega(L^2). \tag{1}$$

Ascending Data-Ordering + Practical WSD Schedule. For an ascending order from $\mathcal{D}^{(M)}$ to $\mathcal{D}^{(1)}$, following a WSD schedule with substantial decay, the expected loss be

$$\bar{\mathcal{L}}(M; \tilde{E}) = \Theta(L^2).$$

Ascending Data-Ordering + WSMD Schedule. For a Warmup-Stable-Moderate-Decay (WSMD) schedule (denoted as E^*) with less decay and a larger ending learning rate can better utilize the ascending data-ordering and break through the above lower bound,

$$\bar{\mathcal{L}}(M; E^*) = \Theta(M^{-\frac{2}{3}} L^2). \tag{2}$$

Ascending Data-Ordering + Stochastic Weight Averaging (SWA). With a constant learning rate, a sample SWA surpasses the aforementioned lower bound. SWA can both get accumulation towards the ground truth and reduce noise.

Theorem 6.1. *Given a learning rate $\eta_0 \leq 1$, the parameter derived by the averaging on the last n weights $\bar{\boldsymbol{w}}_M = \frac{1}{n}\sum_{t=0}^{n-1} \boldsymbol{w}_{M-t}$, where $n = \Theta(M^{\frac{2}{3}})$ such that the expected loss*

$$\mathbb{E}[\mathcal{L}(\bar{\boldsymbol{w}}_M)] = \tilde{O}(M^{-\frac{2}{3}} L^2),$$

where $\tilde{O}(\cdot)$ hides log factors and constants independent of L and M.

7 Conclusion

In this paper, we investigate the interplay between data scheduling and learning rate (LR) schedules. We identify a key conflict in curriculum learning: placing high-quality data towards the end of training is beneficial, but its impact is severely limited by the decayed learning rate at that stage. To address this, we demonstrate that replacing sharp LR decay with model averaging can achieve comparable or even superior results when combined with a data curriculum, especailly in a mid-training setting. Building on this, we propose an approach that integrates model averaging with a moderate LR decay, and discover an under-explored optimization regime for pretraining strategy.

REFERENCES

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML '09, pp. 41–48, New York, NY, USA, 2009. Association for Computing Machinery. ISBN 9781605585161. doi: 10.1145/1553374.1553380. URL https://doi.org/10.1145/1553374.1553380.

Yonatan Bisk, Rowan Zellers, Ronan Le Bras, Jianfeng Gao, and Yejin Choi. Piqa: Reasoning about physical commonsense in natural language, 2019. URL https://arxiv.org/abs/1911.11641.

Daniel Campos. Curriculum learning for language modeling, 2021. URL https://arxiv.org/abs/2108.02170.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? try arc, the ai2 reasoning challenge, 2018. URL https://arxiv.org/abs/1803.05457.

Yalun Dai, Yangyu Huang, Xin Zhang, Wenshan Wu, Chong Li, Wenhui Lu, Shijie Cao, Li Dong, and Scarlett Li. Data efficacy for language model training, 2025. URL https://arxiv.org/abs/2506.21545.

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, Tao Yun, Tian Pei, Tianyu Sun, W. L. Xiao, Wangding Zeng, Wanjia Zhao, Wei An, Wen Liu, Wenfeng Liang, Wenjun Gao, Wenqin Yu, Wentao Zhang, X. Q. Li, Xiangyue Jin, Xianzu Wang, Xiao Bi, Xiaodong Liu, Xiaohan Wang, Xiaojin Shen, Xiaokang Chen, Xiaokang Zhang, Xiaosha Chen, Xiaotao Nie, Xiaowen Sun, Xiaoxiang Wang, Xin Cheng, Xin Liu, Xin Xie, Xingchao Liu, Xingkai Yu, Xinnan Song, Xinxia Shan, Xinyi Zhou, Xinyu Yang, Xinyuan Li, Xuecheng Su, Xuheng Lin, Y. K. Li, Y. Q. Wang, Y. X. Wei, Y. X. Zhu, Yang Zhang, Yanhong Xu, Yanhong Xu, Yanping Huang, Yao Li, Yao Zhao, Yaofeng Sun, Yaohui Li, Yaohui Wang, Yi Yu, Yi Zheng, Yichao Zhang, Yifan Shi, Yiliang Xiong, Ying He, Ying Tang, Yishi Piao, Yisong Wang, Yixuan Tan, Yiyang Ma, Yiyuan Liu, Yongqiang Guo, Yu Wu, Yuan Ou, Yuchen Zhu, Yuduan Wang, Yue Gong, Yuheng Zou, Yujia He, Yukun Zha, Yunfan Xiong, Yunxian Ma, Yuting Yan, Yuxiang Luo, Yuxiang You, Yuxuan Liu, Yuyang Zhou, Z. F. Wu, Z. Z. Ren, Zehui Ren, Zhangli Sha, Zhe Fu, Zhean Xu, Zhen Huang, Zhen Zhang, Zhenda Xie, Zhengyan Zhang, Zhewen Hao, Zhibin Gou, Zhicheng Ma, Zhigang Yan, Zhihong Shao, Zhipeng Xu, Zhiyu Wu, Zhongyu Zhang, Zhuoshu Li, Zihui Gu, Zijia Zhu, Zijun Liu, Zilin Li, Ziwei Xie, Ziyang Song, Ziyi Gao, and Zizheng Pan. Deepseek-v3 technical report, 2025. URL https://arxiv.org/abs/2412.19437.

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, Egor Lakomkin, Ehab AlBadawy, Elina Lobanova, Emily Dinan, Eric Michael Smith, Filip Radenovic, Francisco Guzmán, Frank Zhang, Gabriel Synnaeve, Gabrielle Lee, Georgia Lewis Anderson, Govind Thattai, Graeme Nail, Gregoire Mialon, Guan Pang, Guillem Cucurell, Hailey Nguyen, Hannah Korevaar, Hu Xu, Hugo Touvron, Iliyan Zarov, Imanol Arrieta Ibarra, Isabel Kloumann, Ishan Misra, Ivan Evtimov, Jack Zhang, Jade Copet, Jaewon Lee, Jan Geffert, Jana Vranes, Jason Park, Jay Mahadeokar, Jeet Shah, Jelmer van der Linde, Jennifer Billock, Jenny Hong, Jenya Lee, Jeremy Fu, Jianfeng Chi, Jianyu Huang, Jiawen Liu, Jie Wang, Jiecao Yu, Joanna Bitton, Joe Spisak, Jongsoo Park, Joseph Rocca, Joshua Johnstun, Joshua Saxe, Junteng Jia, Kalyan Vasuden Alwala, Karthik Prasad, Kartikeya Upasani, Kate Plawiak, Ke Li, Kenneth Heafield, Kevin Stone, Khalid El-Arini, Krithika Iyer, Kshitiz Malik, Kuenley Chiu, Kunal Bhalla, Kushal Lakhotia, Lauren Rantala-Yeary, Laurens van der Maaten, Lawrence Chen, Liang Tan, Liz Jenkins, Louis Martin, Lovish Madaan, Lubo Malo, Lukas Blecher, Lukas Landzaat, Luke de Oliveira, Madeline Muzzi, Mahesh Pasupuleti, Mannat Singh, Manohar Paluri, Marcin Kardas, Maria Tsimpoukelli, Mathew Oldham, Mathieu Rita, Maya Pavlova, Melanie Kambadur, Mike Lewis, Min Si, Mitesh Kumar Singh, Mona Hassan, Naman Goyal, Narjes Torabi, Nikolay Bashlykov, Nikolay Bogoychev, Niladri Chatterji, Ning Zhang, Olivier Duchenne, Onur Çelebi, Patrick Alrassy, Pengchuan Zhang, Pengwei Li, Petar Vasic, Peter Weng, Prajjwal Bhargava, Pratik Dubal, Praveen Krishnan, Punit Singh Koura, Puxin Xu, Qing He, Qingxiao Dong, Ragavan Srinivasan, Raj Ganapathy, Ramon Calderer, Ricardo Silveira Cabral, Robert Stojnic, Roberta Raileanu, Rohan Maheswari, Rohit Girdhar, Rohit Patel, Romain Sauvestre, Ronnie Polidoro, Roshan Sumbaly, Ross Taylor, Ruan Silva, Rui Hou, Rui Wang, Saghar Hosseini, Sahana Chennabasappa, Sanjay Singh, Sean Bell, Seohyun Sonia Kim, Sergey Edunov, Shaoliang Nie, Sharan Narang, Sharath Raparthy, Sheng Shen, Shengye Wan, Shruti Bhosale, Shun Zhang, Simon Vandenhende, Soumya Batra, Spencer Whitman, Sten Sootla, Stephane Collot, Suchin Gururangan, Sydney Borodinsky, Tamar Herman, Tara Fowler, Tarek Sheasha, Thomas Georgiou, Thomas Scialom, Tobias Speckbacher, Todor Mihaylov, Tong Xiao, Ujjwal Karn, Vedanuj Goswami, Vibhor Gupta, Vignesh Ramanathan, Viktor Kerkez, Vincent Gonguet, Virginie Do, Vish Vogeti, Vítor Albiero, Vladan Petrovic, Weiwei Chu, Wenhan Xiong, Wenyin Fu, Whitney Meers, Xavier Martinet, Xiaodong Wang, Xiaofang Wang, Xiaoqing Ellen Tan, Xide Xia, Xinfeng Xie, Xuchao Jia, Xuewei Wang, Yaelle Goldschlag, Yashesh Gaur, Yasmine Babaei, Yi Wen, Yiwen Song, Yuchen Zhang, Yue Li, Yuning Mao, Zacharie Delpierre Coudert, Zheng Yan, Zhengxing Chen, Zoe Papakipos, Aaditya Singh, Aayushi Srivastava, Abha Jain, Adam Kelsey, Adam Shajnfeld, Adithya Gangidi, Adolfo Victoria, Ahuva Goldstand, Ajay Menon, Ajay Sharma, Alex Boesenberg, Alexei Baevski, Allie Feinstein, Amanda Kallet, Amit Sangani, Amos Teo, Anam Yunus, Andrei Lupu, Andres Alvarado, Andrew Caples, Andrew Gu, Andrew Ho, Andrew Poulton, Andrew Ryan, Ankit Ramchandani, Annie Dong, Annie Franco, Anuj Goyal, Aparajita Saraf, Arkabandhu Chowdhury, Ashley Gabriel, Ashwin Bharambe, Assaf Eisenman, Azadeh Yazdan, Beau James, Ben Maurer, Benjamin Leonhardi, Bernie Huang, Beth Loyd, Beto De Paola, Bhargavi Paranjape, Bing Liu, Bo Wu, Boyu Ni, Braden Hancock, Bram Wasti, Brandon Spence, Brani Stojkovic, Brian Gamido, Britt Montalvo, Carl Parker, Carly Burton, Catalina Mejia, Ce Liu, Changhan Wang, Changkyu Kim, Chao Zhou, Chester Hu, Ching-Hsiang Chu, Chris Cai, Chris Tindal, Christoph Feichtenhofer, Cynthia Gao, Damon Civin, Dana Beaty, Daniel Kreymer, Daniel Li, David Adkins, David Xu, Davide Testuggine, Delia David, Devi Parikh, Diana Liskovich, Didem Foss, Dingkang Wang, Duc Le, Dustin Holland, Edward Dowling, Eissa Jamil, Elaine Montgomery, Eleonora Presani, Emily Hahn, Emily Wood, Eric-Tuan Le, Erik Brinkman, Esteban Arcaute, Evan Dunbar, Evan Smothers, Fei Sun, Felix Kreuk, Feng Tian, Filippos Kokkinos, Firat Ozgenel, Francesco Caggioni, Frank Kanayet, Frank Seide, Gabriela Medina Florez, Gabriella Schwarz, Gada Badeer, Georgia Swee, Gil Halpern, Grant Herman, Grigory Sizov, Guangyi, Zhang, Guna Lakshminarayanan, Hakan Inan, Hamid Shojanazeri, Han Zou, Hannah Wang, Hanwen Zha, Haroun Habeeb, Harrison Rudolph, Helen Suk, Henry Aspegren, Hunter Goldman, Hongyuan Zhan, Ibrahim Damlaj, Igor Molybog, Igor Tufanov, Ilias Leontiadis, Irina-Elena Veliche, Itai Gat, Jake Weissman, James Geboski, James Kohli, Janice Lam, Japhet Asher, Jean-Baptiste Gaya, Jeff Marcus, Jeff Tang, Jennifer Chan, Jenny Zhen, Jeremy Reizenstein, Jeremy Teboul, Jessica Zhong, Jian Jin, Jingyi Yang, Joe Cummings, Jon Carvill, Jon Shepard, Jonathan McPhie, Jonathan Torres, Josh Ginsburg, Junjie Wang, Kai Wu, Kam Hou U, Karan Saxena, Kartikay Khandelwal, Katayoun Zand, Kathy Matosich, Kaushik Veeraraghavan, Kelly Michelena, Keqian Li, Kiran Jagadeesh, Kun Huang, Kunal Chawla, Kyle Huang, Lailin Chen, Lakshya Garg, Lavender A, Leandro Silva, Lee Bell, Lei Zhang, Liangpeng Guo, Licheng Yu, Liron Moshkovich, Luca Wehrstedt, Madian Khabsa,

Manav Avalani, Manish Bhatt, Martynas Mankus, Matan Hasson, Matthew Lennie, Matthias Reso, Maxim Groshev, Maxim Naumov, Maya Lathi, Meghan Keneally, Miao Liu, Michael L. Seltzer, Michal Valko, Michelle Restrepo, Mihir Patel, Mik Vyatskov, Mikayel Samvelyan, Mike Clark, Mike Macey, Mike Wang, Miquel Jubert Hermoso, Mo Metanat, Mohammad Rastegari, Munish Bansal, Nandhini Santhanam, Natascha Parks, Natasha White, Navyata Bawa, Nayan Singhal, Nick Egebo, Nicolas Usunier, Nikhil Mehta, Nikolay Pavlovich Laptev, Ning Dong, Norman Cheng, Oleg Chernoguz, Olivia Hart, Omkar Salpekar, Ozlem Kalinli, Parkin Kent, Parth Parekh, Paul Saab, Pavan Balaji, Pedro Rittner, Philip Bontrager, Pierre Roux, Piotr Dollar, Polina Zvyagina, Prashant Ratanchandani, Pritish Yuvraj, Qian Liang, Rachad Alao, Rachel Rodriguez, Rafi Ayub, Raghotham Murthy, Raghu Nayani, Rahul Mitra, Rangaprabhu Parthasarathy, Raymond Li, Rebekkah Hogan, Robin Battey, Rocky Wang, Russ Howes, Ruty Rinott, Sachin Mehta, Sachin Siby, Sai Jayesh Bondu, Samyak Datta, Sara Chugh, Sara Hunt, Sargun Dhillon, Sasha Sidorov, Satadru Pan, Saurabh Mahajan, Saurabh Verma, Seiji Yamamoto, Sharadh Ramaswamy, Shaun Lindsay, Shaun Lindsay, Sheng Feng, Shenghao Lin, Shengxin Cindy Zha, Shishir Patil, Shiva Shankar, Shuqiang Zhang, Shuqiang Zhang, Sinong Wang, Sneha Agarwal, Soji Sajuyigbe, Soumith Chintala, Stephanie Max, Stephen Chen, Steve Kehoe, Steve Satterfield, Sudarshan Govindaprasad, Sumit Gupta, Summer Deng, Sungmin Cho, Sunny Virk, Suraj Subramanian, Sy Choudhury, Sydney Goldman, Tal Remez, Tamar Glaser, Tamara Best, Thilo Koehler, Thomas Robinson, Tianhe Li, Tianjun Zhang, Tim Matthews, Timothy Chou, Tzook Shaked, Varun Vontimitta, Victoria Ajayi, Victoria Montanez, Vijai Mohan, Vinay Satish Kumar, Vishal Mangla, Vlad Ionescu, Vlad Poenaru, Vlad Tiberiu Mihailescu, Vladimir Ivanov, Wei Li, Wenchen Wang, Wenwen Jiang, Wes Bouaziz, Will Constable, Xiaocheng Tang, Xiaojian Wu, Xiaolan Wang, Xilun Wu, Xinbo Gao, Yaniv Kleinman, Yanjun Chen, Ye Hu, Ye Jia, Ye Qi, Yenda Li, Yilin Zhang, Ying Zhang, Yossi Adi, Youngjin Nam, Yu, Wang, Yu Zhao, Yuchen Hao, Yundi Qian, Yunlu Li, Yuzi He, Zach Rait, Zachary DeVito, Zef Rosnbrick, Zhaoduo Wen, Zhenyu Yang, Zhiwei Zhao, and Zhiyu Ma. The llama 3 herd of models, 2024. URL https://arxiv.org/abs/2407.21783.

Yuling Gu, Oyvind Tafjord, Bailey Kuehl, Dany Haddad, Jesse Dodge, and Hannaneh Hajishirzi. Olmes: A standard for language model evaluations, 2025. URL https://arxiv.org/abs/2406.08446.

Alexander Hägele, Elie Bakouch, Atli Kosson, Loubna Ben Allal, Leandro Von Werra, and Martin Jaggi. Scaling laws and compute-optimal training beyond fixed training durations. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 76232–76264. Curran Associates, Inc., 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/8b970e15a89bf5d12542810df8eae8fc-Paper-Conference.pdf.

David Heineman, Valentin Hofmann, Ian Magnusson, Yuling Gu, Noah A. Smith, Hannaneh Hajishirzi, Kyle Lo, and Jesse Dodge. Signal and noise: A framework for reducing uncertainty in language model evaluation, 2025. URL https://arxiv.org/abs/2508.13144.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=d7KBjmI3GmQ.

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, 2024. URL https://arxiv.org/abs/2404.06395.

Pavel Izmailov, Dmitrii Podoprikhin, Timur Garipov, Dmitry Vetrov, and Andrew Gordon Wilson. Averaging weights leads to wider optima and better generalization, 2019. URL https://arxiv.org/abs/1803.05407.

Xisen Jin, Xiang Ren, Daniel Preotiuc-Pietro, and Pengxiang Cheng. Dataless knowledge fusion by merging weights of language models. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=FCnohuR6AnM.

Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. Bag of tricks for efficient text classification, 2016. URL https://arxiv.org/abs/1607.01759.

Jean Kaddour. Stop wasting my time! saving days of imagenet and BERT training with latest weight averaging. In *Has it Trained Yet? NeurIPS 2022 Workshop*, 2022. URL https://openreview.net/forum?id=0OrABUHZuz.

Jisu Kim and Juhwan Lee. Strategic data ordering: Enhancing large language model performance through curriculum learning, 2024. URL https://arxiv.org/abs/2405.07490.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017. URL https://arxiv.org/abs/1412.6980.

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*, 2025a.

Houyi Li, Wenzhen Zheng, Qiufeng Wang, Hanshan Zhang, Zili Wang, Shijie Xuyang, Yuantao Fan, Zhenyu Ding, Haoying Wang, Ning Ding, Shuigeng Zhou, Xiangyu Zhang, and Daxin Jiang. Predictable scale: Part i, step law – optimal hyperparameter scaling law in large language model pretraining, 2025b. URL https://arxiv.org/abs/2503.04715.

Jeffrey Li, Alex Fang, Georgios Smyrnis, Maor Ivgi, Matt Jordan, Samir Gadre, Hritik Bansal, Etash Guha, Sedrick Keh, Kushal Arora, Saurabh Garg, Rui Xin, Niklas Muennighoff, Reinhard Heckel, Jean Mercat, Mayee Chen, Suchin Gururangan, Mitchell Wortsman, Alon Albalak, Yonatan Bitton, Marianna Nezhurina, Amro Abbas, Cheng-Yu Hsieh, Dhruba Ghosh, Josh Gardner, Maciej Kilian, Hanlin Zhang, Rulin Shao, Sarah Pratt, Sunny Sanyal, Gabriel Ilharco, Giannis Daras, Kalyani Marathe, Aaron Gokaslan, Jieyu Zhang, Khyathi Chandu, Thao Nguyen, Igor Vasiljevic, Sham Kakade, Shuran Song, Sujay Sanghavi, Fartash Faghri, Sewoong Oh, Luke Zettlemoyer, Kyle Lo, Alaaeldin El-Nouby, Hadi Pouransari, Alexander Toshev, Stephanie Wang, Dirk Groeneveld, Luca Soldaini, Pang Wei Koh, Jenia Jitsev, Thomas Kollar, Alexandros G. Dimakis, Yair Carmon, Achal Dave, Ludwig Schmidt, and Vaishaal Shankar. Datacomp-lm: In search of the next generation of training sets for language models. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 14200–14282. Curran Associates, Inc., 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/19e4ea30dded58259665db375885e412-Paper-Datasets_and_Benchmarks_Track.pdf.

Yunshui Li, Yiyuan Ma, Shen Yan, Chaoyi Zhang, Jing Liu, Jianqiao Lu, Ziwen Xu, Mengzhao Chen, Minrui Wang, Shiyi Zhan, Jin Ma, Xunhao Lai, Deyi Liu, Yao Luo, Xingyan Bin, Hongbin Ren, Mingji Han, Wenhao Hao, Bairen Yi, LingJun Liu, Bole Ma, Xiaoying Jia, Xun Zhou, Siyuan Qiao, Liang Xiang, and Yonghui Wu. Model merging in pre-training of large language models, 2025c. URL https://arxiv.org/abs/2505.12082.

Chonghua Liao, Ruobing Xie, Xingwu Sun, Haowen Sun, and Zhanhui Kang. Exploring forgetting in large language model pre-training. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 2112–2127, Vienna, Austria, July 2025. Association for Computational Linguistics. ISBN 979-8-89176-251-0. doi: 10.18653/v1/2025.acl-long.105. URL https://aclanthology.org/2025.acl-long.105/.

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, 2025. URL https://arxiv.org/abs/2503.12811.

Michael McCloskey and Neal J. Cohen. Catastrophic interference in connectionist networks: The sequential learning problem. volume 24 of *Psychology of Learning and Motivation*, pp. 109–165. Academic Press, 1989. doi: https://doi.org/10.1016/S0079-7421(08)60536-8. URL https://www.sciencedirect.com/science/article/pii/S0079742108605368.

Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. Can a suit of armor conduct electricity? a new dataset for open book question answering. In Ellen Riloff, David Chiang, Julia Hockenmaier, and Jun'ichi Tsujii (eds.), *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 2381–2391, Brussels, Belgium, oct-nov 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-1260. URL https://aclanthology.org/D18-1260/.

Team OLMo, Pete Walsh, Luca Soldaini, Dirk Groeneveld, Kyle Lo, Shane Arora, Akshita Bhagia, Yuling Gu, Shengyi Huang, Matt Jordan, Nathan Lambert, Dustin Schwenk, Oyvind Tafjord, Taira Anderson, David Atkinson, Faeze Brahman, Christopher Clark, Pradeep Dasigi, Nouha Dziri, Michal Guerquin, Hamish Ivison, Pang Wei Koh, Jiacheng Liu, Saumya Malik, William Merrill, Lester James V. Miranda, Jacob Morrison, Tyler Murray, Crystal Nam, Valentina Pyatkin, Aman Rangapur, Michael Schmitz, Sam Skjonsberg, David Wadden, Christopher Wilhelm, Michael Wilson, Luke Zettlemoyer, Ali Farhadi, Noah A. Smith, and Hannaneh Hajishirzi. 2 olmo 2 furious, 2025. URL https://arxiv.org/abs/2501.00656.

OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, Red Avila, Igor Babuschkin, Suchir Balaji, Valerie Balcom, Paul Baltescu, Haiming Bao, Mohammad Bavarian, Jeff Belgum, Irwan Bello, Jake Berdine, Gabriel Bernadett-Shapiro, Christopher Berner, Lenny Bogdonoff, Oleg Boiko, Madelaine Boyd, Anna-Luisa Brakman, Greg Brockman, Tim Brooks, Miles Brundage, Kevin Button, Trevor Cai, Rosie Campbell, Andrew Cann, Brittany Carey, Chelsea Carlson, Rory Carmichael, Brooke Chan, Che Chang, Fotis Chantzis, Derek Chen, Sully Chen, Ruby Chen, Jason Chen, Mark Chen, Ben Chess, Chester Cho, Casey Chu, Hyung Won Chung, Dave Cummings, Jeremiah Currier, Yunxing Dai, Cory Decareaux, Thomas Degry, Noah Deutsch, Damien Deville, Arka Dhar, David Dohan, Steve Dowling, Sheila Dunning, Adrien Ecoffet, Atty Eleti, Tyna Eloundou, David Farhi, Liam Fedus, Niko Felix, Simón Posada Fishman, Juston Forte, Isabella Fulford, Leo Gao, Elie Georges, Christian Gibson, Vik Goel, Tarun Gogineni, Gabriel Goh, Rapha Gontijo-Lopes, Jonathan Gordon, Morgan Grafstein, Scott Gray, Ryan Greene, Joshua Gross, Shixiang Shane Gu, Yufei Guo, Chris Hallacy, Jesse Han, Jeff Harris, Yuchen He, Mike Heaton, Johannes Heidecke, Chris Hesse, Alan Hickey, Wade Hickey, Peter Hoeschele, Brandon Houghton, Kenny Hsu, Shengli Hu, Xin Hu, Joost Huizinga, Shantanu Jain, Shawn Jain, Joanne Jang, Angela Jiang, Roger Jiang, Haozhun Jin, Denny Jin, Shino Jomoto, Billie Jonn, Heewoo Jun, Tomer Kaftan, Łukasz Kaiser, Ali Kamali, Ingmar Kanitscheider, Nitish Shirish Keskar, Tabarak Khan, Logan Kilpatrick, Jong Wook Kim, Christina Kim, Yongjik Kim, Jan Hendrik Kirchner, Jamie Kiros, Matt Knight, Daniel Kokotajlo, Łukasz Kondraciuk, Andrew Kondrich, Aris Konstantinidis, Kyle Kosic, Gretchen Krueger, Vishal Kuo, Michael Lampe, Ikai Lan, Teddy Lee, Jan Leike, Jade Leung, Daniel Levy, Chak Ming Li, Rachel Lim, Molly Lin, Stephanie Lin, Mateusz Litwin, Theresa Lopez, Ryan Lowe, Patricia Lue, Anna Makanju, Kim Malfacini, Sam Manning, Todor Markov, Yaniv Markovski, Bianca Martin, Katie Mayer, Andrew Mayne, Bob McGrew, Scott Mayer McKinney, Christine McLeavey, Paul McMillan, Jake McNeil, David Medina, Aalok Mehta, Jacob Menick, Luke Metz, Andrey Mishchenko, Pamela Mishkin, Vinnie Monaco, Evan Morikawa, Daniel Mossing, Tong Mu, Mira Murati, Oleg Murk, David Mély, Ashvin Nair, Reiichiro Nakano, Rajeev Nayak, Arvind Neelakantan, Richard Ngo, Hyeonwoo Noh, Long Ouyang, Cullen O'Keefe, Jakub Pachocki, Alex Paino, Joe Palermo, Ashley Pantuliano, Giambattista Parascandolo, Joel Parish, Emy Parparita, Alex Passos, Mikhail Pavlov, Andrew Peng, Adam Perelman, Filipe de Avila Belbute Peres, Michael Petrov, Henrique Ponde de Oliveira Pinto, Michael, Pokorny, Michelle Pokrass, Vitchyr H. Pong, Tolly Powell, Alethea Power, Boris Power, Elizabeth Proehl, Raul Puri, Alec Radford, Jack Rae, Aditya Ramesh, Cameron Raymond, Francis Real, Kendra Rimbach, Carl Ross, Bob Rotsted, Henri Roussez, Nick Ryder, Mario Saltarelli, Ted Sanders, Shibani Santurkar, Girish Sastry, Heather Schmidt, David Schnurr, John Schulman, Daniel Selsam, Kyla Sheppard, Toki Sherbakov, Jessica Shieh, Sarah Shoker, Pranav Shyam, Szymon Sidor, Eric Sigler, Maddie Simens, Jordan Sitkin, Katarina Slama, Ian Sohl, Benjamin Sokolowsky, Yang Song, Natalie Staudacher, Felipe Petroski Such, Natalie Summers, Ilya Sutskever, Jie Tang, Nikolas Tezak, Madeleine B. Thompson, Phil Tillet, Amin Tootoonchian, Elizabeth Tseng, Preston Tuggle, Nick Turley, Jerry Tworek, Juan Felipe Cerón Uribe, Andrea Vallone, Arun Vijayvergiya, Chelsea Voss, Carroll Wainwright, Justin Jay Wang, Alvin Wang, Ben Wang, Jonathan Ward, Jason Wei, CJ Weinmann, Akila Welihinda, Peter Welinder, Jiayi Weng, Lilian Weng,

Matt Wiethoff, Dave Willner, Clemens Winter, Samuel Wolrich, Hannah Wong, Lauren Workman, Sherwin Wu, Jeff Wu, Michael Wu, Kai Xiao, Tao Xu, Sarah Yoo, Kevin Yu, Qiming Yuan, Wojciech Zaremba, Rowan Zellers, Chong Zhang, Marvin Zhang, Shengjia Zhao, Tianhao Zheng, Juntang Zhuang, William Zhuk, and Barret Zoph. Gpt-4 technical report, 2024. URL https://arxiv.org/abs/2303.08774.

Guilherme Penedo, Hynek Kydlíček, Loubna Ben allal, Anton Lozhkov, Margaret Mitchell, Colin Raffel, Leandro Von Werra, and Thomas Wolf. The fineweb datasets: Decanting the web for the finest text data at scale. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 30811–30849. Curran Associates, Inc., 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/370df50ccfdf8bde18f8f9c2d9151bda-Paper-Datasets_and_Benchmarks_Track.pdf.

Guilherme Penedo, Hynek Kydlíček, Vinko Sabolčec, Bettina Messmer, Negar Foroutan, Amir Hossein Kargaran, Colin Raffel, Martin Jaggi, Leandro Von Werra, and Thomas Wolf. Fineweb2: One pipeline to scale them all – adapting pre-training data processing to every language, 2025. URL https://arxiv.org/abs/2506.20920.

Qwen, :, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tianyi Tang, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. Qwen2.5 technical report, 2025. URL https://arxiv.org/abs/2412.15115.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. Winogrande: An adversarial winograd schema challenge at scale, 2019. URL https://arxiv.org/abs/1907.10641.

Maarten Sap, Hannah Rashkin, Derek Chen, Ronan Le Bras, and Yejin Choi. Social IQa: Commonsense reasoning about social interactions. In Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan (eds.), *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 4463–4473, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1454. URL https://aclanthology.org/D19-1454/.

Kashun Shum, Yuzhen Huang, Hongjian Zou, Ding Qi, Yixuan Liao, Xiaoxin Chen, Qian Liu, and Junxian He. Predictive data selection: The data that predicts is the data that teaches. *arXiv preprint arXiv:2503.00808*, 2025.

Vaibhav Singh, Paul Janson, Paria Mehrbod, Adam Ibrahim, Irina Rish, Eugene Belilovsky, and Benjamin Thérien. Beyond cosine decay: On the effectiveness of infinite learning rate schedule for continual pre-training. In *ES-FoMo III: 3rd Workshop on Efficient Systems for Foundation Models*, 2025. URL https://openreview.net/forum?id=JX3lLTQPiG.

Dan Su, Kezhi Kong, Ying Lin, Joseph Jennings, Brandon Norick, Markus Kliegl, Mostofa Patwary, Mohammad Shoeybi, and Bryan Catanzaro. Nemotron-cc: Transforming common crawl into a refined long-horizon pretraining dataset, 2025. URL https://arxiv.org/abs/2412.02595.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. CommonsenseQA: A question answering challenge targeting commonsense knowledge. In Jill Burstein, Christy Doran, and Thamar Solorio (eds.), *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 4149–4158, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1421. URL https://aclanthology.org/N19-1421/.

Meituan LongCat Team, 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, Jiayu Qin, Jingang Wang, Jiyuan He, Jun Kuang, Junhui Mei, Kai Liang, Ke He, Kefeng Zhang, Keheng Wang, Keqing He, Liang Gao, Liang Shi, Lianhui Ma, Lin Qiu, Lingbin Kong, Lingtong Si, Linkun Lyu, Linsen Guo, Liqi Yang, Lizhi Yan, Mai Xia, Man Gao, Manyuan Zhang, Meng Zhou, Mengxia Shen, Mingxiang Tuo, Mingyang Zhu, Peiguang Li, Peng Pei, Peng Zhao, Pengcheng Jia, Pingwei Sun, Qi Gu, Qianyun Li, Qingyuan Li, Qiong Huang, Qiyuan Duan, Ran Meng, Rongxiang Weng, Ruichen Shao, Rumei Li, Shizhe Wu, Shuai Liang, Shuo Wang, Suogui Dang, Tao Fang, Tao Li, Tefeng Chen, Tianhao Bai, Tianhao Zhou, Tingwen Xie, Wei He, Wei Huang, Wei Liu, Wei Shi, Wei Wang, Wei Wu, Weikang Zhao, Wen Zan, Wenjie Shi, Xi Nan, Xi Su, Xiang Li, Xiang Mei, Xiangyang Ji, Xiangyu Xi, Xiangzhou Huang, Xianpeng Li, Xiao Fu, Xiao Liu, Xiao Wei, Xiaodong Cai, Xiaolong Chen, Xiaoqing Liu, Xiaotong Li, Xiaowei Shi, Xiaoyu Li, Xili Wang, Xin Chen, Xing Hu, Xingyu Miao, Xinyan He, Xuemiao Zhang, Xueyuan Hao, Xuezhi Cao, Xunliang Cai, Xurui Yang, Yan Feng, Yang Bai, Yang Chen, Yang Yang, Yaqi Huo, Yerui Sun, Yifan Lu, Yifan Zhang, Yipeng Zang, Yitao Zhai, Yiyang Li, Yongjing Yin, Yongkang Lv, Yongwei Zhou, Yu Yang, Yuchen Xie, Yueqing Sun, Yuewen Zheng, Yuhua Wei, Yulei Qian, Yunfan Liang, Yunfang Tai, Yunke Zhao, Zeyang Yu, Zhao Zhang, Zhaohua Yang, Zhenchao Zhang, Zhikang Xia, Zhiye Zou, Zhizhao Zeng, Zhongda Su, Zhuofan Chen, Zijian Zhang, Ziwen Wang, Zixu Jiang, Zizhe Zhao, Zongyu Wang, and Zunhai Su. Longcat-flash technical report, 2025. URL https://arxiv.org/abs/2509.01322.

Teknium. Openhermes 2.5: An open dataset of synthetic data for generalist llm assistants, 2023. URL https://huggingface.co/datasets/teknium/OpenHermes-2.5.

Changxin Tian, Jiapeng Wang, Qian Zhao, Kunlong Chen, Jia Liu, Ziqi Liu, Jiaxin Mao, Wayne Xin Zhao, Zhiqiang Zhang, and Jun Zhou. Wsm: Decay-free learning rate schedule via checkpoint merging for llm pre-training, 2025. URL https://arxiv.org/abs/2507.17634.

Kushal Tirumala, Aram Markosyan, Luke Zettlemoyer, and Armen Aghajanyan. Memorization without overfitting: Analyzing the training dynamics of large language models. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 38274–38290. Curran Associates, Inc., 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/fa0509f4dab6807e2cb465715bf2d249-Paper-Conference.pdf.

Howe Tissue, Venus Wang, and Lu Wang. Scaling law with learning rate annealing, 2024. URL https://arxiv.org/abs/2408.11029.

Xin Wang, Yudong Chen, and Wenwu Zhu. A survey on curriculum learning, 2021. URL https://arxiv.org/abs/2010.13166.

Maurice Weber, Daniel Fu, Quentin Anthony, Yonatan Oren, Shane Adams, Anton Alexandrov, Xiaozhong Lyu, Huu Nguyen, Xiaozhe Yao, Virginia Adams, Ben Athiwaratkun, Rahul Chalamala, Kezhen Chen, Max Ryabinin, Tri Dao, Percy Liang, Christopher Ré, Irina Rish, and Ce Zhang. Redpajama: an open dataset for training large language models, 2024. URL https://arxiv.org/abs/2411.12372.

Kaiyue Wen, Zhiyuan Li, Jason S. Wang, David Leo Wright Hall, Percy Liang, and Tengyu Ma. Understanding warmup-stable-decay learning rates: A river valley loss landscape view. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=m51BgoqvbP.

Alexander Wettig, Aatmik Gupta, Saumya Malik, and Danqi Chen. QuRating: Selecting high-quality data for training language models. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings*

of Machine Learning Research, pp. 52915–52971. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/wettig24a.html.

Alexander Wettig, Kyle Lo, Sewon Min, Hannaneh Hajishirzi, Danqi Chen, and Luca Soldaini. Organize the web: Constructing domains enhances pre-training data curation, 2025. URL https://arxiv.org/abs/2502.10341.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. Qwen3 technical report, 2025. URL https://arxiv.org/abs/2505.09388.

Enneng Yang, Zhenyi Wang, Li Shen, Shiwei Liu, Guibing Guo, Xingwei Wang, and Dacheng Tao. Adamerging: Adaptive model merging for multi-task learning. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=nZP6NgD3QY.

Yang Zhang, Amr Mohamed, Hadi Abdine, Guokan Shang, and Michalis Vazirgiannis. Beyond random sampling: Efficient language model pretraining via curriculum learning, 2025. URL https://arxiv.org/abs/2506.11300.

A Experiments Details

Pretraining Setting. Our experiments refer to the DataComps-LM (DCLM) framework (Li et al., 2024) at 1B-1x scale. We adopt the Qwen2.5-1.5B model architecture (Qwen et al., 2025) and train models on a 30B token subset of the DCLM-Baseline dataset (Li et al., 2024). The architecture incorporates modern advancements such as SwiGLU activation functions, Grouped-Query Attention (GQA). We use the DCLM fasttext scores as the quality metrics and sort data in ascending-order (AO) to create a data curriculum. The reverse data curriculum will sort data in descending-order (DO). We moderately tune the key parameters, and set peak learning rate to 3×10^{-3}, and use a sequence length of 4096 with a batch size of 512, which we found provides a good trade-off between throughput and training stability. For LR decay schedules in the experiment, we set the final learning rate to 1×10^{-5}, which aligns with optimal settings found in prior work (Li et al., 2024; Luo et al., 2025; Li et al., 2025b). To ensure reproducibility of our findings, we provide a detailed list of the model and optimizer hyperparameters in Table 3.

Evaluation Setting. To compare methods, we track validation loss during training and evaluate performance on a suite of downstream tasks. Since the data distribution shifts throughout a curriculum, a fixed validation set drawn randomly from the entire dataset may not be representative. To ensure a consistent and meaningful measure of progress, we created a dedicated high-quality validation set. This set consists of 100k documents with the highest scores, drawn from a disjoint partition of the DCLM-Baseline dataset from our training data. For downstream evaluation, we use the OLMES benchmark (Gu et al., 2025), which is well-suited for evaluating models at our scale. For evaluation, we report performance on MMLU (Hendrycks et al., 2021), ARC-easy/challenge (Clark et al., 2018), CommonSenseQA (CSQA) (Talmor et al., 2019), OpenBookQA (Mihaylov et al., 2018), PIQA (Bisk et al., 2019), Social IQa (Sap et al., 2019), and WinoGrande (Sakaguchi et al., 2019), covering the evaluation of world knowledge, common sense, and understanding capabilities. Within them, we select MMLU (Hendrycks et al., 2021), ARC-easy/challenge (Clark et al., 2018), and CSQA (Talmor et al., 2019) as *Core* benchmarks. According to recent work (Heineman et al., 2025), these tasks feature a higher signal-to-noise ratio to distinguish performances of different models (Heineman et al., 2025). Moreover, as shown in Figure 7, the average downstream scores show a strong correlation with the validation loss, especially the experiments on DCLM Baseline dataset with data ordered by DCLM fasttext scores.

Data Scoring. Raw web data must pass through a processing pipeline before it is used for pre-training. The raw data first go through heuristic-filtering rules. Afterwards, in model-based filtering phase, a scorer model assigns a quality score to each data sample. For example, the DCLM Baseline dataset uses scores from a fasttext model (Joulin et al., 2016) measuring similarity to high-quality sources like OpenHermes 2.5 (Teknium, 2023) and top posts from the ELI5 subreddit. Another approach, PreSelect (Shum et al., 2025), scores data based on its similarity to downstream tasks. Typically, these scores are used to filter the dataset by removing samples below a certain quality threshold. In contrast, our work does not discard data; instead, we use these quality scores to define the data ordering for curriculum learning.

Table 3: Model and optimizer hyperparameters for our Qwen2.5-1.5B experiments.

Hyperparameter	**Value**
Model Configuration	
Sequence Length	4096
Hidden Size	1536
FFN Intermediate Size	8960
Number of Layers	28
Number of Attention Heads	12
Number of Key-Value Heads (GQA)	2
Vocabulary Size	151936
Optimizer Configuration	
Optimizer	AdamW (FP32 State)
Weight Decay	0.1
Adam β_1	0.9
Adam β_2	0.95
Adam ϵ	1.0×10^{-8}
Gradient Clipping	1.0

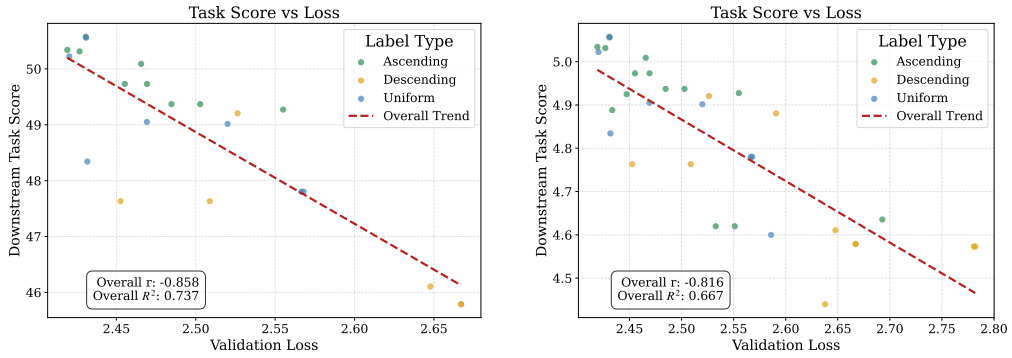
LR schedule Choice Ablation. The decay phase of WSD schedule can use various functions. Adapted from prior work (Hägele et al., 2024; Luo et al., 2025), we find that the 1-sqrt decay function, $f(t) = \eta_0 \left(1 - \sqrt{r(t)}\right) + \eta_T \sqrt{r(t)}$, and the sqrt-cube function, $f(t) = \eta_0 \left(1 - r(t)\right)^{1.5}$, produce strong, comparable results, as shown in Table 4. Here, $r(t) = \frac{t - t_{\text{decay}}}{T - t_{\text{decay}}}$ represents the progress through the decay phase, which starts at step t_{decay}. Both functions outperform simpler alternatives like linear decay. In this work, we use the 1-sqrt function due to its wide adoption (Hägele et al., 2024; Tian et al., 2025). In addition, Figure 1 adopts 37% decay ratio, and all other experiments set decay ratio between 15% to 20% aligning with optimal decay ratios reported in prior work (Hägele et al., 2024; Hu et al., 2024).

Table 4: models trained under WSD schedules under 1-sqrt and sqrt-cube decay functions produce similar results.

Dataset	**Schedule**	**MMLU**	**ARC-c**	**ARC-e**	**CSQA**	**OBQA**	**PIQA**	**SIQA**	**Wino.**	**Avg.**
Random	1-sqrt	26.60	27.42	42.28	42.26	37.20	67.85	42.02	51.30	42.12
Filtered	1-sqrt	26.97	30.10	44.04	44.72	36.20	69.15	42.58	51.78	43.19
Random	sqrt-cube	26.62	27.42	42.28	42.42	35.00	68.28	43.65	51.70	42.17
Filtered	sqrt-cube	26.70	31.10	44.04	42.59	36.60	68.44	42.89	52.17	43.07

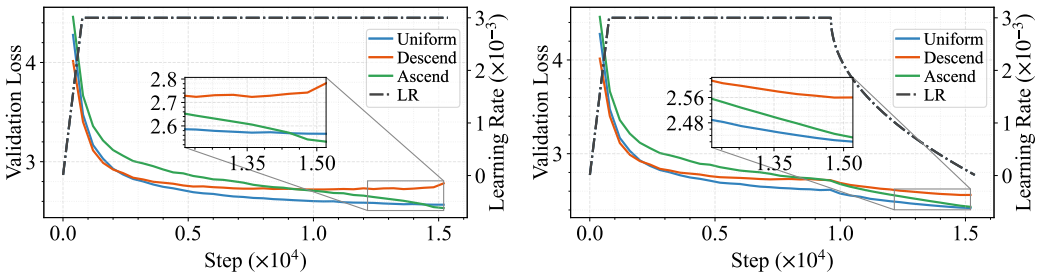
Table 5: Model Checkpoint Weights

Checkpoint Index	3725	3750	3775	3800	3825	3843
Weight	0.4249	0.1760	0.1350	0.1138	0.1003	0.0500



(a) DCLM scores on DCLM Baseline dataset. (b) Different scores on different dataset.

Figure 7: Pearson correlation coefficient (r) and R-square value (R^2) between downstream task scores and validation losses. (a) Select experiments that ordering data samples from DCLM Baseline dataset by DCLM fasttext scores. (b) Include experiments that uses PreSelect scores and use WebOrganizer Dataset.



(a) Constant LR schedule. (b) WSD LR schedule.

Figure 8: The benefits of data curriculum by PreSelect score also diminish. We show the validation loss curves of constant and WSD LR schedules, under different data schedules, including uniform, ascending, and descending order by PreSelect scores. The ascending curriculum can win over uniform data under constant schedule but can not match the constant schedule in WSD LR schedule.

Table 6: The *folding* strategy may be effective when peak LR is relatively low, but the benefit can vanish in a high peak LR regime, which is supposed to closer to an optimal setting.

Order	Strategy	Peak LR	MMLU	ARC-c	ARC-e	CSQA	OBQA	PIQA	SIQA	Wino.	Avg.
Uniform	–	1×10^{-4}	25.70	28.43	37.72	34.32	30.20	61.81	40.99	50.83	38.75
Ascend	Sorting	1×10^{-4}	26.57	28.76	38.42	35.63	28.80	61.97	41.40	50.36	38.99
Ascend	Folding	1×10^{-4}	25.69	29.43	38.77	35.22	32.20	61.43	40.99	50.04	39.22
Uniform	–	3×10^{-3}	28.68	33.78	50.35	45.95	36.60	68.66	43.65	53.35	45.13
Ascend	Sorting	3×10^{-3}	27.78	37.12	47.89	44.47	37.40	67.85	43.30	55.80	45.20
Ascend	Folding	3×10^{-3}	28.33	33.11	48.25	43.82	38.80	69.21	43.76	52.88	44.77

The Mid-Training Experiment Setting. Mid-training is a recently emerging practice in LLM pretraining (Yang et al., 2025; OLMo et al., 2025; Hu et al., 2024). Mid-training incorporates high-quality data in the decay stage of WSD schedule. In our mid-training experiments, first stable phase includes 29B tokens of data from the WebOrganizer (Wettig et al., 2025), as the low-quality data, and the second decay phase selects roughly 5B tokens from DCLM-Baseline (Li et al., 2024), as the high-quality data. The WebOrganizer data has not gone through model-filtering process to sieve the low-quality data inside dataset while the DCLM-Baseline keeps top 10% high-quality data from base dataset, whose distribution is similar to the WebOrganizer. The LR decays to to 1×10^{-5}

within the the high-quality regime. This experiment serves as a more practical setting: where most pretraining data is limited, while a small portion of data is of high quality. The experiments include ablations on 3 variables: (1) we consider the data schedules where data order is phase-wise, like uniform in the first phase and in ascending-order in the second phase (that is, (*U,A*) in Table 2), and ascending all-together (*A-T*)) experiment, sort data samples from both phases in a whole; (2) we conduct experiments on both standard WSD schedules and CMA, over different data schedule settings; (3) we try both SMA and EMA to validate the robustness of weight averaging strategy.

Score Sanity Check: Reverse Data Curriculum Results in Poor Performance. We also examine the results of reverse data curriculum that sort data samples in the descending-order of quality metric. As shown in Table 7, we find that the descending order results consistently get worse than uniform order or ascending order. It indicates that the data scoring is self-consistent, that memorizing high-score data samples helps and memorizing low-score data samples hurts. As a contrast, some prior work (Wettig et al., 2024) reports improvement from both forward and reverse data curriculum, which may indicates inconsistency between quality metrics and evaluation benchmarks.

Table 7: Comparison of Learning Rate Schedules (Constant, Cosine, WSD) and Data Orders (Uniform, Ascend, Descend) on various benchmark tasks. All scores are reported in percentage points.

LR Schedule	Order	MMLU	ARC-c	ARC-e	CSQA	OBQA	PIQA	SIQA	Wino.	Avg.
Constant	Uniform	30.30	30.43	55.61	49.80	44.60	70.29	45.19	56.20	47.80
	Ascend	31.14	39.46	61.23	49.96	43.00	70.51	43.14	56.51	49.37
	Descend	29.43	33.11	45.96	45.21	41.00	69.86	44.98	56.75	45.79
Cosine	Uniform	30.49	38.13	59.47	49.14	42.20	71.87	45.19	56.51	49.13
	Ascend	30.80	39.80	59.12	51.27	42.60	71.55	45.65	57.06	49.73
	Descend	29.51	34.11	52.98	48.81	42.60	72.42	45.45	55.17	47.63
WSD	Uniform	30.77	42.14	61.05	50.86	45.20	72.42	45.75	56.27	50.56
	Ascend	31.58	38.80	61.05	50.37	45.80	71.82	46.01	57.30	50.34
	Descend	29.56	40.13	54.39	50.70	43.20	72.96	45.96	56.75	49.20

Table 8: Downstream performance for experiments with pre-selected ascending data. **WA**: Weight Averaging (EMA: Exponential, SMA: Simple). **LRS**: Learning Rate Schedule (WSD: decay to 1×10^{-5}, Const: Constant LR, WSMD: WSD with moderate decay to 1×10^{-3}). **Core**: Average score on the first four, high signal-to-noise tasks (MMLU, ARC-c, ARC-e, CSQA). Both Core and Avg. scores are annotated with a subscript indicating the performance change relative to the baseline (first row). Subscripts in **bold green** indicate an improvement of ≥ 0.5, light green an improvement of > 0, and red a decrease. Our proposed methods (using WA) are highlighted in gray.

WA	Order	LRS	MMLU	ARC-c	ARC-e	CSQA	Core	OBQA	PIQA	SIQA	Wino.	Avg.
✗	Ascend	WSD	31.12	35.79	57.89	48.81	43.40	41.00	71.82	46.21	58.41	48.88
EMA	Ascend	Const	31.85	37.46	61.05	49.39	$44.94_{+1.54}$	38.40	70.51	45.34	55.33	$48.67_{-0.21}$
EMA	Ascend	WSMD	31.98	39.80	61.93	49.39	$45.77_{+2.37}$	39.60	70.78	45.60	55.96	$49.38_{+0.50}$
SMA	Ascend	WSMD	31.99	39.46	62.11	50.04	$45.90_{+2.50}$	39.40	71.06	46.06	55.96	$49.51_{+0.63}$

Weight Computation of WMA. The computation of WMA follows Tian et al. (2025). We first assume the equivalent LR schedule is WSD schedule, with ending LR as 0.05 of peak LR, with 1-sqrt decay function. We suppose $\eta_1, \ldots, \eta_N$ are normalized LR values, thus $\eta_1 = 1$. We compute the weights by $w_i = \eta_i - \eta_{i+1}$ and $w_N = \eta_N$. In this way, $\sum_{i=1}^{N} w_i = \eta_1 = 1$. The resulting weights are shown in Table 5.

Curriculum Model Averaging (CMA) Pipeline. First, in the **data scheduling** stage, the entire training dataset is sorted in ascending order based on a data quality score. Second, during the **training** stage, we employ a warmup-constant LR schedule. This schedule consists of a standard linear warmup phase followed by a high, constant learning rate for the remainder of training, with no subsequent decay. Finally, instead of reducing the learning rate for stabilization, we perform **model averaging**. We compute a weighted average of model weights from several checkpoints saved during the latter stages of training to produce the final model.

Table 9: Downstream performance for experiments on WebOrganizer dataset (Wettig et al., 2025). **WA**: Weight Averaging (EMA: Exponential, SMA: Simple). **LRS**: Learning Rate Schedule (WSD: decay to 1×10^{-5}, Const: Constant LR, WSMD: WSD with moderate decay to 1×10^{-3}). **Core**: Average score on the first four, high signal-to-noise tasks (MMLU, ARC-c, ARC-e, CSQA). Both Core and Avg. scores are annotated with a subscript indicating the performance change relative to the baseline (first row). Performance changes are color-coded: **bold green** (≥ 0.5 improvement), light green (> 0 improvement), and red (decrease). Our proposed methods (using WA) are highlighted in gray.

WA	Order	LRS	MMLU	ARC-c	ARC-e	CSQA	Core	OBQA	PIQA	SIQA	Wino.	Avg.
✗	Uniform	WSD	28.92	34.45	47.72	47.83	39.73	36.60	72.03	43.76	56.67	46.00
✗	Ascend	WSD	29.09	32.78	51.58	47.42	$40.22_{+0.49}$	38.00	72.14	44.73	55.09	$46.35_{+0.35}$
EMA	Uniform	WSMD	28.28	34.11	47.89	48.81	$39.77_{+0.04}$	39.20	71.65	43.76	55.56	$46.16_{+0.16}$
EMA	Ascend	WSMD	28.56	31.10	50.88	48.89	$39.86_{+0.13}$	39.60	71.44	44.11	56.75	$46.42_{+0.42}$
EMA	Uniform	Const	28.03	33.44	47.72	47.42	$39.15_{-0.58}$	40.60	70.78	43.65	55.72	$45.92_{-0.08}$
EMA	Ascend	Const	29.32	33.11	55.09	48.89	$41.60_{\mathbf{+1.87}}$	38.40	71.00	45.29	55.41	$47.06_{\mathbf{+1.06}}$

Additional Interpretation of CMA. This approach allows the model to learn effectively from high-quality data with a consistent learning rate, while model averaging ensures the convergence and stability of the final parameters. The constant learning rate ensures that high-quality data has a significant impact, preventing the information loss caused by LR decay. Meanwhile, model averaging improves stability and mitigates the risks associated with the absence of learning rate decay. These two components have a synergistic relationship, producing a combined effect that is greater than the sum of its parts.

Algorithm 1 Curriculum Model Averaging (CMA)

Input: Unsorted dataset D, quality scoring function $Q(\cdot)$, training steps T, warmup steps T_w, peak learning rate η_{peak}, number of checkpoints to average k, averaging decay hyperparameter α, checkpointing interval s.
Output: Final model parameters $\bar{\boldsymbol{\theta}}_{\text{final}}$.

\# Stage 1: Data Scheduling
Sort dataset D to create D_{sorted} where for any samples x_i, x_j, if $i < j$, then $Q(x_i) \leq Q(x_j)$.

\# Stage 2: Warmup-Constant LR Training
Initialize model parameters $\boldsymbol{\theta}_0$.
for $t = 1$ **to** T **do**
  if $t \leq T_w$ **then**
    $\eta_t \leftarrow \eta_{peak} \cdot (t/T_w)$ ▷ Linear warmup
  else
    $\eta_t \leftarrow \eta_{peak}$ ▷ Constant LR
  end if
  Fetch next data batch B_t from D_{sorted}.
  $\boldsymbol{\theta}_t \leftarrow \text{OptimizerUpdate}(\boldsymbol{\theta}_{t-1}, \eta_t, B_t)$ ▷ e.g., Adam
  if $t \in \{T - (k-1)s, \ldots, T - s, T\}$ **then**
    Save checkpoint $\boldsymbol{\theta}_t$.
  end if
end for

\# Stage 3: Model Averaging, e.g., EMA or SMA.
Let the set of saved checkpoints be $\{\boldsymbol{\theta}_{T-is}\}_{i=0}^{k-1}$.
EMA: $\bar{\boldsymbol{\theta}}_{\text{final}} \leftarrow \frac{\sum_{i=0}^{k-1} \alpha^i \boldsymbol{\theta}_{T-is}}{\sum_{i=0}^{k-1} \alpha^i}$ ▷ SMA: $\bar{\boldsymbol{\theta}}_{\text{final}} \leftarrow \frac{\sum_{i=0}^{k-1} \boldsymbol{\theta}_{T-is}}{k}$
return $\bar{\boldsymbol{\theta}}_{\text{final}}$

Quality Metric Choice. We use the DCLM fasttext score (Li et al., 2024) as our quality metric, as it is a well-validated indicator of data quality for filtering and selection. A significant practical ad-

vantage is that these scores are often pre-computed during data preprocessing, allowing our method to be integrated into existing pipelines without additional computational cost. We also use PreSelect score (Shum et al., 2025) in our ablation experiments.

Model Averaging Technique. As formalized in Algorithm 1, we employ a Exponential Moving Average (EMA) over the final k checkpoints, which are saved at a regular interval s. The averaged parameters $\bar{\boldsymbol{\theta}}_{\text{final}}$ are computed as:

$$\bar{\boldsymbol{\theta}}_{\text{final}} = \frac{\sum_{i=0}^{k-1} \alpha^i \boldsymbol{\theta}_{T-is}}{\sum_{i=0}^{k-1} \alpha^i}$$

where $\boldsymbol{\theta}_{T-is}$ is the checkpoint saved at step $T - is$, k is the number of checkpoints, s is the checkpointing interval, and $\alpha \in (0, 1]$ is a decay hyperparameter. This formulation assigns exponentially higher weights to more recent checkpoints, which are trained on higher-quality data. Thus, this technique focuses the final model on high-quality signals while smoothing the parameter variance that can result from training with a high, constant learning rate. Additionally, we also run experiments with Simple Moving Average (SMA) as

$$\bar{\boldsymbol{\theta}}_{\text{final}} \leftarrow \frac{\sum_{i=0}^{k-1} \boldsymbol{\theta}_{T-is}}{k},$$

for its simplicity and effectiveness (Izmailov et al., 2019).

Practical Implementation. The initial data sorting is a one-time, offline process. We perform this step efficiently using Apache Spark, whose sorting algorithms are highly optimized for large-scale datasets. The computational overhead of this step is therefore minimal compared to the overall cost of pretraining, making CMA a practical approach. Additionally, the approach can share the quality metric scores with the model-filtering process, saving the cost of another scoring process.

B Additional Related Work

B.1 Curriculum Learning

Curriculum learning, which guides a model to learn from easy to hard samples, is a widely used technique in deep learning (Bengio et al., 2009). In language modeling, this principle is adapted to arrange data from simple to complex to stabilize training and improve convergence (Campos, 2021). For LLM pretraining, this often translates to ordering data by quality, from low to high. The goal is to leverage the model's tendency to better remember later data, ensuring it retains high-quality information while mitigating catastrophic forgetting (Wettig et al., 2024; Dai et al., 2025; Tirumala et al., 2022; Liao et al., 2025).

Research in this area has followed two main directions: (1) developing better quality metrics to ensure high-quality data is placed at the end of training (Wettig et al., 2024; Dai et al., 2025), and (2) exploring alternative data orderings, such as folded or interleaved curricula that sort data within distinct stages (Dai et al., 2025; Zhang et al., 2025). However, these instance-level curricula have produced only negligible improvements and have not been validated at a sufficient scale (Dai et al., 2025; Zhang et al., 2025; Wettig et al., 2024; Campos, 2021). Wettig et al. (2024) proposes to sort data with LLM-annotated scores but reports a limited improvement and shows benefits from both ascending and descending quality orders, lacking a clear interpretation on the underlying mechanisms. Campos (2021); Kim & Lee (2024) lacks validation experiments for curriculum benefits. Zhang et al. (2025); Dai et al. (2025) finds very marginal improvement of vanilla data curriculum, and propose folding (Section 4) or interleaved curricula to sort data within consecutive stages. However, as shown in Section C, the benefit of folding shows only on a smaller scale experiments with a low LR, and the benefit diminishes and even get worse in a scaled-up and high LR regime. More detailed discussions are left in Section C. Consequently, fine-grained curricula are rarely used in practice (Yang et al., 2025; DeepSeek-AI et al., 2025; Grattafiori et al., 2024). Instead, some recent models adopt a coarse, two-stage curriculum: they first train on a large, low-quality dataset and then refine the model on a smaller, high-quality corpus (Hu et al., 2024; OLMo et al., 2025).

Our work diagnoses a key reason for these limited results: the detrimental interaction between data schedules and learning rate schedules commonly used in prior work. Commonly used LR

schedules typically include an aggressive decay phase, such as the cosine schedule (Wettig et al., 2024; Dai et al., 2025; Zhang et al., 2025). We show that a data curriculum can be beneficial when this interaction is removed. We then investigate this coupling and demonstrate that it leads to insufficient learning from high-quality data, and we explain the benefit of folding strategy in previous studies through this lens.

B.2 Learning Rate Schedule

The learning rate (LR) schedule is a critical component of model training. Traditional schedules like cosine decay require a fixed number of training steps, which limits their flexibility (Singh et al., 2025). The recently proposed Warmup-Stable-Decay (WSD) schedule offers superior performance without needing a predefined training budget (Hu et al., 2024; Hägele et al., 2024). Subsequent work has validated the effectiveness of WSD through the lens of scaling laws (Tissue et al., 2024; Luo et al., 2025; Li et al., 2025a).

In this work, we use the WSD schedule with a 1-sqrt decay function. Its clear separation between a stable-LR phase and a decay phase allows us to isolate and study the effect of LR decay on different data schedules. The WSD schedule produces a characteristic loss curve: the loss stays high during the stable phase and drops sharply during the decay phase. To explain this phenomenon, Wen et al. (2025) proposed a "river valley" loss landscape. They hypothesize that optimization involves a "river" direction corresponding to meaningful progress and a "valley" direction corresponding to noise. Our analysis builds on a similar intuition, emphasizing the role of data samples, on progress direction and on noise scale.

B.3 Model Averaging

Model averaging is a technique that combines the parameters of multiple checkpoints to create a single, improved model (Izmailov et al., 2019; Jin et al., 2023; Yang et al., 2024). This technique has been used in LLM pretraining to accelerate convergence (Kaddour, 2022) and boost performance (Li et al., 2025c), and was reportedly used in training models like LLaMA (Grattafiori et al., 2024). Notably, Tian et al. (2025) combined model averaging with a decay-free LR schedule and claimed it surpasses the standard WSD schedule (Hu et al., 2024).

Despite these advances, the interaction between data quality, model averaging, and learning schedules remains under-explored. Prior work has often shown model averaging to be only comparable to LR decay (Li et al., 2025c) or slightly better than a WSD schedule with a short decay phase (Tian et al., 2025). The latter comparison may be unfair, as it potentially underestimates the full benefit of a properly configured WSD schedule. Our work specifically investigates the interaction between model averaging and data quality, a crucial factor that these studies did not isolate.

C Additional Experiments and Discussion

Ablation Study. To validate our understanding and approaches can be generalized, we conduct experiments on other quality metrics and pretraining dataset. (1) **Quality metrics:** We ablate on PreSelect score as the data quality metric and sort data in ascending order of PreSelect score. As shown in in Table 8, we find that over the ascending-order of PreSelect score, both CMA and CDMA performs better than standard WSD schedule and WSMD performs better on average scores. In addition, we find that ordering by PreSelect score is slightly worse than that by DCLM score on average, which may need further exploring. We conjecture that it is because the base dataset is filtered with DCLM fasttext score in model-filtering process, while sorted in PreSelect, resulting in a inconsistent quality arrangement. (2) **Pretraining Dataset:** Previously, we have validated our results on DCLM-Baseline and a mixture of DCLM-Baseline and WebOrganizer. We further check our approach is applicable to WebOrganizer alone. As shown in Table 9, we find that the using model average with data curriculum, can generate better results in weborganizer dataset than a LR decay approach. In this setting, weight average without decay shows a larger benefit than that with moderate decay, possibly indicating the potential benefit of a high LR when high-quality data is extremely sparse.

Detailed Comparison with Related Works about Curriculum Learning. In previous works of curriculum learning in LLM pretraining, the impact of LR schedules are largely overlooked. Cosine schedules are common practices for these works, and typically with a peak LR at scale of 10^{-4} (Wettig et al., 2024; Dai et al., 2025; Zhang et al., 2025; Kim & Lee, 2024; Campos, 2021). Among them, we will focus on discussion about the works with positive validation experiments (Wettig et al., 2024; Dai et al., 2025; Zhang et al., 2025). QuRating (Wettig et al., 2024) mainly focus on their quality metric and test it on the curriculum learning setting. They reports 0.6% average improvement on downstream tasks for low-to-high quality ordering. As a comparison, our best result (*Const+SMA*) achieves over 2.7% improvement on the same baseline (*Cos+Uniform*). Although the downstream tasks are not exactly the same, we achieve our results on a slightly larger scale and on a high-quality dataset, which is typically more challenging to improve performance, and we do not optimize the quality metric and use the existing DCLM fasttext scores as quality metrics (Li et al., 2024; Wettig et al., 2025). Moreover, Qurating reports a 0.5% improvement on a decreasing order curriculum, which is paradoxical over the advantage of curriculum learning. In contrast, we find a consistent performance drop for reverse data curriculum in our experiments, as shown in Table 7 and discussed in Section A. Recent works (Dai et al., 2025; Zhang et al., 2025) also test the curriculum learning. Dai et al. (2025) reports a negligible benefit from curriculum when not optimizing quality metric, and a higher benefit when optimizing the quality metric, but a limited scale. Zhang et al. (2025) reports a slight benefit of curriculum at a scale of 1B model and 10B data. They concurrently propose a folding strategy, use interleaved data curriculum, that is to sort data in several consecutive stages, to improve the performance. However, in our replication experiments detailed in Section C, we find the benefit reveals when peak LR is at 1×10^{-4} to follow prior works, but the benefit diminishes in a higher and better peak LR regime, around 3×10^{-3} in our setting. These results shows that the folding strategy may not be adapted to a near-optimal hyperparameter regime, at least in our experiment setting. In our experiments, we use DCLM fasttext score and focus on LR schedule ablation, and a more sophisticated quality metric may additionally contribute to the boost in prior work.

Folding Experiment: Low Peak LR vs High Peak LR. Prior works (Zhang et al., 2025; Dai et al., 2025) report the advantage of folding on both sorting and uniform, differing from our results. We conduct experiment, and deduce the reason that our experiments use well-tuned hyperparameters on model-filtered dataset, where a high peak LR can improve data efficiency without inducing spikes. Better data efficiency can contribute to more improvement for uniform baseline than folding.

To align with the previous works (Dai et al., 2025; Zhang et al., 2025), we run experiments on models with roughly 0.6B parameters, trained on about 30B tokens. We conduct experiments on 3 kinds of data schedules: *Uniform*, *Ascend + Sorting*, *Ascend + Folding*. The introduction of folding strategy can refer to Section 4. Then we set peak LR to two scales: 1×10^{-4}, aligning with Dai et al. (2025); Zhang et al. (2025), and 3×10^{-3} used in our experiment setting. The fold number follows Dai et al. (2025), and is set to 3. The downstream results are reported in Table 6. When peak LR is 1×10^{-4}, folding strategy performs best, and sorting is slightly better than uniform. However, when peak LR is 3×10^{-3}, the order changes, that *Ascend + Sorting* outperforms the other two and *Ascend + Folding* can even not match *Uniform*. We conjecture that, the high peak LR can accelerate the convergence process, thus may outweighs the role of forgetting, especially when we set ending LR as 1×10^{-5} by default, which may not fully utilize the utility of high-quality data in folding and sorting.

Discussion: Correlation between Downstream Performance and Validation Loss. The validation loss evaluation and downstream scores may exhibits some consistency, like Figure 2 suggesting best results in an aggressive regime, but a moderate ending LR is better in Figure 5. This observations are related to Figure 7(b), where we find ascending order training can achieve a slightly higher downstream task score on average. The validation loss reveals an average prediction capability on a high-quality dataset, while the data near the end can be more informative and aligned with task, helps model be accurate on this local part of data distribution. We deduce our results from a comprehensive analysis on both evaluation results and is validated across different settings, which can reduce the misalignment between these evaluations.

D A FULL THEORETICAL DEMONSTRATION

As we reported and discussed above, the benefit of curriculum learning emerges when we apply a weight averaging manner instead of a learning rate schedule with excess decaying, such as Cosine or WSD schedule in practical pretraining. In the following, we present a simple theoretical model that recovers the above empirical insight. The main proof of this section can be found in Section E.

Problem Setup. We consider a quadratic loss function $\mathcal{L}(\boldsymbol{w}) = \frac{1}{2}\|\boldsymbol{w} - \boldsymbol{w}^*\|_2^2$, where $\boldsymbol{w} = (w_1, w_2) \in \mathbb{R}^2$ represents the trainable parameter, and $\boldsymbol{w}^*$ denotes the ground truth, which is set to the original point $(0, 0)$. We use Stochastic Gradient Descent (SGD) to optimize this problem. We denote the one-sample loss used to calculate the gradient for the t-th iteration as $\ell_t(\boldsymbol{w}) := \|\boldsymbol{w} - \boldsymbol{x}_t\|_2^2$, where data point $\boldsymbol{x}_t$ is sampled from some given dataset $\mathcal{D} = \{\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}, \ldots, \boldsymbol{x}^{(M)}\}$. Therefore, we have the SGD update rule for the t-th iteration as $\boldsymbol{w}_t = \boldsymbol{w}_{t-1} - \eta_t \nabla \ell_t(\boldsymbol{w}_{t-1})$, where the η_t denotes the learning rate in the t-th iteration. We initialize the parameter as $\boldsymbol{w}_0 = (Md, 0)$. We denote the learning rate schedule by $E := \{\eta_1, \eta_2, \ldots, \eta_M\}$. We denote $\boldsymbol{w}_t = (w_t^{(1)}, w_t^{(2)})$. We define $\mathcal{W}_{M;E}$ to be the distribution of $\boldsymbol{w}_M$. The randomness within $\boldsymbol{w}_M$ comes from the random draw of the distribution in SGD. We further define the expected loss $\bar{\mathcal{L}}(M; E) := \mathbb{E}_{\boldsymbol{w}\sim\mathcal{W}_{M;E}}[\mathcal{L}(\boldsymbol{w})]$.

In the following, we consider the training dataset $\mathcal{D}$, which consists of M different data points with varying data qualities. Specifically, data point $\boldsymbol{x}^{(i)} = (x_1^{(i)}, x_2^{(i)})$ satisfy that $x_1^{(i)} = (i-1)d$ and $x_2^{(i)} \sim \mathrm{Uniform}(-L, L)$, we further set $d = L/M$. $x^{(i)}$ provides signal in first dimension and introduce noise in the second dimension. Next, we consider two sampling strategies for each iteration of SGD: (1) We sample one data point uniformly from $\mathrm{Uniform}(\mathcal{D})$; (2) we sample one data point from $\mathcal{D}$ in an ascending order. In other word, in t-th iteration, $\boldsymbol{x}_t = \boldsymbol{x}^{(M-t+1)} \in \mathcal{D}$. The visualization of optimization trajectories in simulation experiment can refer to Figure 6.

Uniform sampling + Learning rate Schedule. SGD acts as an exponential averaging of the current parameter and the sampled data point. Once we uniformly sample data points with no ordering, then on the x-axis, the parameter would approximately oscillate from 0 to $(m-1)d$ with a large variance. We can prove that given any data schedule E starting with some $\eta_1 \leq 1$, the expected loss for uniformly sampling SGD has a lower bound

$$\min_E \bar{\mathcal{L}}(M; E) = \Omega(L^2). \tag{3}$$

This lower bound is derived from the loss on the x-axis. When we apply the uniform sampling, SGD cannot get enough signal towards the right direction; instead, the SGD optimizer would approach to the mean of $x_1^{(1)}, x_1^{(2)}, \ldots, x_1^{(M)}$ in expectation.

Ascending Data-Ordering + Practical WSD Schedule. Next, we sample data in an ascending order from $\mathcal{D}^{(M)}$ to $\mathcal{D}^{(1)}$, using the following WSD learning rate schedule $\tilde{E}$ such that $\eta_t = \frac{1}{2}$ for $1 \leq t \leq \lfloor 0.9M \rfloor$, and $\eta_t = \frac{1}{T-(M-T_0)}$ for $\lfloor 0.9M \rfloor + 2 \leq t \leq M$, where $T_0 = M - \lfloor 0.9M \rfloor$. In this learning rate schedule, we follow a practical setting, where we decay 10% of the total iterations. We then show that for this learning rate schedule $\tilde{E}$, the expected loss still cannot break the lower bound

$$\bar{\mathcal{L}}(M; \tilde{E}) = \Theta(L^2).$$

Ascending Data-Ordering + WSMD Schedule. In the above, we show that even using an ascending data-ordering, the loss lower bound does not improve if we decay too much in the learning rate schedule. Next, we show a Warmup-Stable-Moderate Decay (WSMD) schedule with less decay and a larger ending learning rate can better utilize the ascending data-ordering and get a smaller loss. Specifically, do a modification of the above WSD schedule, setting $T_0 = \Theta(M^{\frac{2}{3}})$. WSMD schedule can break through the above lower bound, denoted as E^*

$$\bar{\mathcal{L}}(M; E^*) = \Theta(M^{-\frac{2}{3}} L^2). \tag{4}$$

Ascending Data-Ordering + Stochastic Weight Averaging (SWA). Despite the failure of the practical WSD learning rate schedule, we demonstrate that with a constant learning rate, a sample SWA surpasses the aforementioned lower bound. The reason is that: (1) First, along the x-axis,

with a constant learning rate, the updated parameter gets a larger gradient accumulation towards the ground truth compared with the practical WSD with 10% decay, which is too much to get enough loss reduction. (2) Second, the SWA allows appropriate averaging along the y-axis and results in noise reduction, thus leading to smaller loss, as the WSMD schedule does.

Theorem D.1. *Given a learning rate $\eta_0 \leq 1$, the parameter derived by the averaging on the last n weights $\bar{\boldsymbol{w}}_M = \frac{1}{n}\sum_{t=0}^{n-1} \boldsymbol{w}_{M-t}$, where $n = \Theta(M^{\frac{2}{3}})$ such that the expected loss*

$$\mathbb{E}[\mathcal{L}(\bar{\boldsymbol{w}}_M)] = \tilde{O}(M^{-\frac{2}{3}}L^2),$$

where $\tilde{O}(\cdot)$ hides log factors and constants independent of L and M.

E Proofs in Section 6

In Section 6, we analyze the bounds of expected loss under four different optimization cases:

1. **Uniform sampling + Learning rate Schedule**.
2. **Ascending data-ordering + Practical WSD schedule**.
3. **Ascending data-ordering + WSMD schedule**.
4. **Ascending data-ordering + Stochastic Weight Averaging (SWA)**.

In the following, we give the proof of their corresponding theoretical claims we mentioned in Section 6 one by one.

Lemma E.1. *Consider the uniform sampling, for any learning rate schedule E such that $0 \leq \eta_i \leq 1$, and the parameter initialized at $(L, 0)$, it holds that*

$$\min_E \bar{\mathcal{L}}(M; E) = \Omega(L^2).$$

Proof. We first consider the update rule of SGD in the optimization process on the x-axis as

$$w_t^{(1)} = w_t^{(1)} - \eta_t(w_{t-1}^{(1)} - x_t^{(1)}).$$

Then, taking the expectation over the randomness in SGD and the data generation gives

$$\begin{aligned}\mathbb{E}[w_t^{(1)}] &= (1-\eta_t)\mathbb{E}[w_{t-1}^{(1)}] + \eta_t\mathbb{E}[\boldsymbol{x}_t] \\ &= (1-\eta_t)\mathbb{E}[w_{t-1}^{(1)}] + \eta_t\frac{M(M-1)d}{2} \\ &\geq \frac{M(M-1)d}{2}.\end{aligned}$$

Thus, we write out the lower bound for the expected loss

$$\mathbb{E}[\mathcal{L}(\boldsymbol{w}_t)] = \mathbb{E}[\boldsymbol{w}_t^\top \boldsymbol{w}_t] \geq \mathbb{E}[w_t^{(1)} w_t^{(1)}] \geq \mathbb{E}[w_t^{(1)}] \geq \mathbb{E}[w_t^{(1)}] = \Theta(L^2).$$

The above equation completes the proof. □

For Case 2 and Case 3, we give a more general lemma, for which the conclusions for Case 2 and Case 3 are direct corollaries.

Lemma E.2. *Consider the Ascending data-ordering, and a class of WSD learning rate schedules with the following formula*

$$\eta_t = \begin{cases} \eta_0 & \text{for } 1 \leq t \leq M - T_0 + 1 \\ \frac{1}{T - T_0} & \text{for } M - T_0 + 2 \leq t \leq M, \end{cases}$$

where $T_0 = \omega(1)$ and $\eta_0 = \frac{1}{2}$, it holds for any learning rate schedule E with the above formula that

$$\bar{\mathcal{L}}(M; E) = \tilde{\Theta}\left(T_0^2 d^2 + \frac{L^2}{T_0}\right).$$

Proof. We write out the update rule on the x-axis in the Ascending data-ordering case

$$w_t^{(1)} = (1-\eta_t)w_{t-1}^{(1)} + \eta_t x_1^{(M-t+1)}.$$

Using the above update rule, we can get the expression of $w_M^{(1)}$ as

$$w_M^{(1)} = \prod_{t=1}^{M}(1-\eta_t)w_0^{(0)} + \sum_{i=1}^{M}\prod_{j=i+1}^{M}(1-\eta_j)\eta_i x_{M-i+1}^{(1)}.$$

Then, plugging in the formula of the learning rate schedule gives

$$w_M^{(1)} = \frac{1}{T_0}\sum_{i=1}^{T_0-1} x_{M-i+1}^{(1)} + \frac{1}{2^{T_0-1}}w_{T_0+1}.$$

The above equation uses the following fact

$$\begin{aligned}&(1-\frac{1}{T_0})\cdot(1-\frac{1}{T_0-1})\cdots(1-\frac{1}{i+1})\cdot\frac{1}{i}\\ &= \frac{T_0-1}{T_0}\cdot\frac{T_0-2}{T_0-1}\cdots\frac{i}{i+1}\cdot\frac{1}{i} = \frac{1}{T_0},\end{aligned}$$

where $2 \le i \le T_0$ Also notice that $T_0 = \omega(1)$, then we have

$$\begin{aligned}w_M^{(1)} &= \frac{1}{T_0}\sum_{i=1}^{T_0-1} x_{M-i+1}^{(1)} + \frac{1}{2^{T_0-1}}\sum_{i=1}^{T_0-1}\frac{1}{2^{T_0-i+1}}x_{M-i+1}^{(1)} + \frac{1}{2^M}w_0^{(1)}w_0^{(1)}\\ &= \frac{1}{T_0}\sum_{i=1}^{T_0-1} x_{M-i+1}^{(1)} + \frac{1}{2^{T_0-1}}\sum_{i=1}^{\log(Md)}\frac{1}{2^{T_0-i+1}}x_{M-i+1}^{(1)}\left(1+o(1)\right)\\ &= \frac{1}{T_0}\sum_{i=1}^{T_0-1} x_{M-i+1}^{(1)} + \tilde{o}\left(d\right)\\ &= \frac{1}{T_0}\sum_{i=1}^{T_0-1} x_{M-i+1}^{(1)}\left(1+o(1)\right).\end{aligned}$$

Thus, the expected loss on the axis follows

$$\mathbb{E}[x_M^{(1)}x_M^{(1)}] = \frac{1}{(T_0)^2}\sum_{i=1}^{T_0-1}(M-i+1)d = \Theta((T_0)^2d^2).$$

Similarly, we write out the expected loss on the y-axis

$$\begin{aligned}\mathbb{E}[x_M^{(2)}x_M^{(2)}] &= \frac{1}{(T_0)^2}\sum_{i=1}^{T_0-1}\mathbb{E}[x_{M-i+1}^{(2)}x_{M-i+1}^{(2)}]\left(1+o(1)\right)\\ &= \frac{L^2}{T_0}(1+o(1))\\ &= \Theta\left(\frac{L^2}{T_0}\right).\end{aligned}$$

The above equation completes the proof. Specifically, taking $T_0 = \lfloor 0.9M \rfloor$ and $T_0 = \Theta(M^{\frac{2}{3}})$ gives the results in Equation (3) and Equation (4). □

In the end, we show how a simple SWA method can beat the practical WSD schedule, which is stated in Theorem D.1

Proof of Theorem D.1. We first write out the expression for the parameter after a rescaled SWA as

$$\bar{\boldsymbol{w}}_M = \sum_{t=0}^{n-1}\frac{\alpha}{n}\boldsymbol{w}_{M-t}.$$

Then, plugging in the constant learning rate schedule gives

$$\bar{\boldsymbol{w}}_M = \sum_{i=1}^{n}\sum_{j=0}^{i-1}\frac{\alpha}{n}(1-\eta_0)^j\boldsymbol{x}_i + \sum_{i=n+1}^{M}\sum_{j=0}^{n-1}\frac{\alpha}{n}(1-\eta_0)^{j+i-n}\boldsymbol{x}_i + \sum_{i=1}^{n}(1-\eta_0)^{M-i}\boldsymbol{w}_0.$$

We then decouple the parameter into the x-axis component and the y-axis component as $\bar{\boldsymbol{w}}_M = (\bar{w}_M^{(1)}, \bar{w}_M^{(2)})$. For the x-axis, we have $\boldsymbol{x}_1^{(1)} = 0$, thus we can rewrite theabove equation in the x-axis

$$\begin{aligned}\bar{w}_M^{(1)} = {} & \frac{\alpha_0}{n}\eta_0 x_1^{(1)} + \eta_0\left(\frac{\alpha_0}{n}(1-\eta_0)+\alpha\right)x_2^{(1)} + \eta_0\left(\frac{\alpha_0}{n}(1-\eta_0)^2+\alpha(1-\eta_0)+\alpha\right)x_3^{(1)} \\ & + \cdots + \left(\alpha_0(1-\eta_0)^{n-1} + \alpha\sum_{i=0}^{n-2}(1-\eta_0)^i\right)x_n^{(1)} \\ & + \sum_{i=n+1}^{M}\sum_{j=0}^{n-1}\frac{\alpha}{n}(1-\eta_0)^{j+i-n}x_i^{(1)} \\ & + \sum_{i=1}^{n}(1-\eta_0)^{M-i}w_0^{(0)}.\end{aligned}$$

We then take $\alpha_0 = \frac{1}{\eta_0}$ and $\alpha = 1$, then we recursively have

$$\begin{aligned}\frac{\alpha_0}{n}\eta_0 &= \frac{1}{n} \\ \left(\alpha_0(1-\eta_0)^j + \alpha\sum_{i=0}^{j-1}(1-\eta_0)^i\right) &= \frac{1}{n} \quad \text{for all } 1 \le j \le n-1 \\ \sum_{j=0}^{n-1}\frac{\alpha}{n}(1-\eta_0)^{j+i-n} &= \frac{1}{n}(1-\eta)^{i-n} \quad \text{for all } 1 \le j \le n-1.\end{aligned}$$

The above equations gives

$$\begin{aligned}\bar{w}_M^{(1)} &= \frac{1}{n}\sum_{i=1}^{n}x_i^{(1)} + \sum_{i=n+1}^{M}\frac{1}{n}(1-\eta_0)^{i-n}x_i^{(1)} + \sum_{i=1}^{n}(1-\eta_0)^{M-i}w_0^{(0)} \\ &= \frac{1}{n}\sum_{i=1}^{n}x_i^{(1)} + \sum_{i=n+1}^{\log(Md)}\frac{1}{n}(1-\eta_0)^{i-n}x_i^{(1)}\,(1+o(1)) \\ &\le \frac{1}{n}\sum_{i=1}^{n+\log(Md)}x_i^{(1)}\,(1+o(1))\,. \end{aligned} \tag{5}$$

Thus, we give the upper bound for the component of expected loss on the x-axis

$$\mathbb{E}[w_M^{(1)}w_M^{(1)}] = \tilde{O}\left(n^2d^2\right) = \tilde{O}\left(M^{-\frac{2}{3}}L^2\right).$$

Similarly, for the y-axis, we have

$$\bar{w}_M^{(2)} \le \frac{1}{n}\sum_{i=1}^{n+\log(md)}x_i^{(2)}(1+o(1)). \tag{6}$$

Notice that the only difference between the derivation of Equation (6) and the derivation of Equation (5) is we cannot replace α with α_0 since $x_1^{(2)}$ is not the constant 0, but the difference $(\alpha_0-\alpha)\boldsymbol{w}_M$ can be obviously merged into the main term $\frac{1}{n}\sum_{i=1}^{n+\log(md)}x_i^{(2)}$. And then we get

$$\mathbb{E}[w_M^{(2)}w_M^{(2)}] = \tilde{O}\left(\frac{L^2}{n}\right) = \tilde{O}\left(M^{-\frac{2}{3}}L^2\right).$$

Finally, notice that the rescale constant α can be merged into the $\tilde{O}$ notation, thus we complete the proof. □