Model Merging in Pre-training of Large Language Models

Yunshui Li, Yiyuan Ma, Shen Yan, Chaoyi Zhang, Jing Liu, Jianqiao Lu, Ziwen Xu
Mengzhao Chen, Minrui Wang, Shiyi Zhan, Jin Ma, Xunhao Lai, Yao Luo, Xingyan Bin
Hongbin Ren, Mingji Han, Wenhao Hao, Bairen Yi, Lingjun Liu, Bole Ma, Xiaoying Jia
Xun Zhou, Liang Xiang, Yonghui Wu
ByteDance Seed
liyunshui@bytedance.com

Abstract

Model merging has emerged as a promising technique for enhancing large language models, though its application in large-scale pre-training remains relatively unexplored. In this paper, we present a comprehensive investigation of model merging techniques during the pre-training process. Through extensive experiments with both dense and Mixture-of-Experts (MoE) architectures ranging from millions to over 100 billion parameters, we demonstrate that merging checkpoints trained with constant learning rates not only achieves significant performance improvements but also enables accurate prediction of annealing behavior. These improvements lead to both more efficient model development and significantly lower training costs. Our detailed ablation studies on merging strategies and hyperparameters provide new insights into the underlying mechanisms while uncovering novel applications. Through comprehensive experimental analysis, we offer the open-source community practical pre-training guidelines for effective model merging.

1 Introduction

Modern large language models (LLMs) [Seed et al., 2025, Achiam et al., 2023, Guo et al., 2025, Team et al., 2024, Yang et al., 2024a] have demonstrated remarkable capabilities with widespread applications across diverse tasks. Despite their exceptional performance in fundamental tasks, LLMs still face several critical challenges, including the extensive pre-training costs, discounted effectiveness of domain-specific post-training, imprecisely-predictable performance scaling, as well as the instability of large-scale training. Model merging [Yang et al., 2024b], as a relatively young topic, presents a promising approach to alleviate these practical challenges.

Recently, the benefits of model merging have been primarily studied in the post-training stage, where several models fine-tuned on different downstream tasks are combined into a single but more versatile model [Ilharco et al., 2023, Zhou et al., 2024, Yu et al., 2024]. For example, using the DARE [Yu et al., 2024] method to merge WizardLM [Xu et al., 2024] with WizardMath [Luo et al., 2025] shows a significant performance enhancement on GSM8K [Cobbe et al., 2021], raising its score from 2.2 to 66.3. In contrast, research on model merging during the pre-training phase remains scarce. Such pre-training merging typically involves combining checkpoints from a single training trajectory, as explored in LAWA [Kaddour, 2022] which utilizes model merging to accelerate the LLM training. However, as the model and data scales dramatically, independent researchers struggle to evaluate model merging's impact on large-scale models, mainly due to limited access to intermediate checkpoints from extensive pre-training. Although DeepSeek [Liu et al., 2024a] and LLaMA-3 [Grattafiori et al., 2024] have both indicated their employment of model merging techniques for model development, detailed information regarding these techniques has not been publicly disclosed.

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

In this work, we mainly focus on model merging during the pre-training stage, introducing Pre-trained Model Average (PMA), a novel strategy for model-level weight merging during pre-training. To comprehensively evaluate PMA, we trained a diverse set of LLMs of varying sizes and architectures from scratch, including Dense models Grattafiori et al. [2024] with parameters spanning from 411M to 70B, as well as Mixture-of-Experts (MoE) architectures Shazeer et al. [2017] with activated/total parameters ranging from 0.7B/7B to 20B/200B. We first investigate the performance impact of PMA and establish systematic evaluations across different phases of the warmup-stable-decay (WSD) learning schedule, which lately becomes a popular choice of lr scheduler for LLM pre-training since Hu et al. [2024]. Experimental results demonstrate that model merging during the stable training phase yields consistent performance gains at different training steps. More remarkably, applying PMA at early-stage of the $cosine$-decay phase usually achieve comparable or even superior performance to their final-stage annealing counterparts. These findings suggest that during the extensively lengthy pre-training stage with constant lr, PMA can serve as a fast, reliable yet low-cost simulator for the annealed performance, enabling both faster validation cycles and significant computational savings.

Building upon our PMA framework, we first evaluate its performance with various prevalent merging strategies, including Simple Moving Average (SMA) Johnston et al. [1999], Weighted Moving Average (WMA) Perry [2010] and Exponential Moving Average (EMA) Hunter [1986]. Notably, our experiments demonstrate that the performance differences among these methods gradually become negligible. We further investigate how these important factors of PMA, namely, the interval between each merging checkpoint, the number of models involved in merging, and the size of the model, would affect merging performance. Our analysis reveals two important findings: First, the optimal merging interval exhibits a clear scaling relationship with model size. Second, incorporating more checkpoints in the merging process consistently improves performance once training is completed.

Furthermore, we also investigated whether PMA could produce more effective initialization weights for the consecutive continued training (CT) or supervised fine-tuning (SFT) Wei et al. [2022] stages to enhance the downstream model performance. We practically observed that entering CT and SFT stages with PMA applied could yield smoother *GradNorm* curves, which thus helps stabilize the training dynamics yet without harming the performance, compared to initializing these stages with the latest available checkpoint as usual. This finding inspire a novel application of model merging for training stabilization, which we dubbed as PMA-init. We demonstrate that in scenarios when the LLM training experiences severe irrecoverable loss spikes with broken training dynamics, applying PMA-init over N preceding checkpoints to resume training, enables reliable recovery from unstable training trajectories.

In summary, our paper makes the following key contributions:

- We present the Pre-trained Model Averaging (PMA) strategy, a novel framework for model merging during LLM pre-training. Through extensive experiments across model scales (from millions to over 100B parameters), we demonstrate that merging checkpoints from the stable training phase produces consistent and significant performance improvements.
- We delved into novel applications of model merging for weight initialization (PMA-init), to help stabilize training process without harming the downstream performance, especially when it suffers from irrecoverable loss spikes with broken training dynamics. Through extensive experiments, we demonstrate the effectiveness of PMA-init on both CT and SFT stages.
- We also comprehensively ablated various model merging techniques with their associated hyper-parameters. Our findings offer the research community practical pre-training guidelines with effective model merging. Nevertheless, the low cost and rapid deployment of PMA also make it a reliable and economic monitor for the pre-training process, to flexibly simulate the ultimate model performance after annealing.

2 Related Work

Model merging is an emerging field undergoing rapid development, with diverse applications across various domains. Typically, model merging is implemented during the **post-training** phase [Ilharco et al., 2023, Zhou et al., 2024, Yu et al., 2024, Yadav et al., 2024], where multiple models fine-tuned

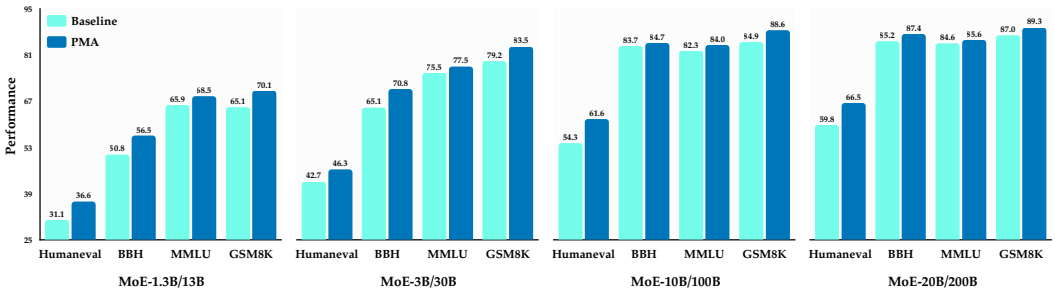

Figure 1: Comparison of downstream task performance for MoE models of varying sizes under stable training, before and after model merging.

on different downstream tasks are combined by merging their weights. This process effectively integrates the distinct capabilities of each individual model, resulting in a unified model that exhibits robust and comprehensive performance.

Recently, several methods have advanced this field significantly. For instance, Task Arithmetic [Ilharco et al., 2023], Ties-Merging [Yadav et al., 2023], and AdaMerging [Yang et al., 2024c] integrate Vision Transformer (ViT) models [Dosovitskiy et al., 2021] trained on distinct visual classification tasks, producing a single model capable of multi-task object classification. PAPA [Jolicoeur-Martineau et al., 2023] integrates the broad applicability of ensembling with the computational efficiency of weight averaging. MetaGPT [Zhou et al., 2024] frames model merging as a multi-task learning problem, aiming to minimize the average loss between the merged model and individual task-specific models. Fisher Merging [Matena and Raffel, 2022] employs a weighted fusion of model parameters, with weights determined by the Fisher information matrix. RegMean [Jin et al., 2023] elegantly addresses the merging process by formulating it as a linear regression problem solvable through closed-form solutions. Evolutionary-model-merge [Akiba et al., 2025] efficiently optimizes merging coefficients using evolutionary algorithms. Additionally, DARE [Yu et al., 2024] merges multiple task-specific language models into a versatile unified model by randomly dropping and subsequently rescaling the delta parameters.

However, research on model merging during the **pre-training** phase remains relatively limited. Such studies typically refer to incorporating checkpoints within a single training trajectory during large language model (LLM) pre-training. For example, LAWA [Kaddour, 2022, Hägele et al., 2024, Li et al., 2022] demonstrated that merging checkpoints at intermediate stages can significantly accelerate training. Sanyal et al. [Sanyal et al., 2024] further indicated that checkpoint averaging combined with a high learning rate in pre-training trajectories contributes to faster convergence. Additionally, Checkpoint Merging [Liu et al., 2024b] provided a comprehensive evaluation of the effectiveness of merging checkpoints at different stages during the pre-training of the Baichuan2 [Yang et al., 2023] LLM. Furthermore, technical reports of large-scale models such as Deepseek V3 [Liu et al., 2024a] and LLaMA3.1 [Grattafiori et al., 2024] also mention the use of model merging techniques during pre-training, although detailed methodologies have not been publicly disclosed. This paper primarily explores techniques for model merging within the pre-training paradigm. To the best of our knowledge, this is the first study to provide detailed technical insights into scaling model merging methods to significantly larger model sizes. We also discuss practical approaches for effective model merging and analyze its potential capabilities as well as its limitations.

3 Preliminaries

In this section, we describe the fundamental experimental framework, introduce the notations and concepts used in model merging, and present multiple variants of model merging techniques.

Experimental setup. In terms of model architecture, we independently trained a series of MoE and dense models. We employ a Warmup-Stable-Decay (WSD) learning rate scheduler Hu et al. [2024], which begins with a short warmup period, followed by an extended period of stable training at a constant learning rate, and concludes with annealing to a relatively small learning rate. The learning rates are determined according to scaling law guidelines Bi et al. [2024], Kaplan et al. [2020],

employing optimal values for training on an internal pretraining corpus comprising trillions of tokens. Although specific model architectures and datasets have not yet been publicly released, we posit that our findings are not strongly tied to these particular choices, as subsequent experiments primarily focus on MoE structures. Related conclusions for dense models are provided in the Appendix A. For evaluation, we primarily report results on open-source benchmarks in both few-shot and zero-shot settings, including: ARC-Challenge Clark et al. [2018], BBH Suzgun et al. [2023], DROP Dua et al. [2019], WinoGrande Sakaguchi et al. [2021], HellaSwag Zellers et al. [2019], MMLU Hendrycks et al. [2021], C-Eval Huang et al. [2023], TriviaQA Joshi et al. [2017], Ape210K Zhao et al. [2020], GSM8K Cobbe et al. [2021], MATH Zhao et al. [2020], MBPP Austin et al. [2021], HumanEval Chen et al. [2021], AGIEval Zhong et al. [2024], GPQA Rein et al. [2024], and MMLU-Pro Wang et al. [2024]. The weighted average score across these benchmarks serves as the model's comprehensive performance metric. Unless otherwise specified, we report this score as the model's performance metric to ensure evaluation reliability.

Notions and concepts. Our main focus is on model merging during pre-training, where the merged entities are sequential checkpoints along the training trajectory. Suppose we aim to merge N models, with each model's parameters denoted as M_i (where i ranges from 1 to N). Each model has an associated weighting coefficient w_i, and the merged model M_{avg} is computed as:

$$M_{\text{avg}} = \sum_{i=1}^{N} w_i M_i. \tag{1}$$

We assume that the data consumption of these models form an arithmetic sequence with a common difference V, formulated as:

$$V = T_{i+1} - T_i, \tag{2}$$

where T_i represents the cumulative number of tokens consumed by the i-th model.

Model merging variants. Model merging techniques vary primarily in how they assign weights (w_i) to individual models. This paper examines three popular approaches for weight assignment, namely the Simple Moving Average (SMA), Exponential Moving Average (EMA), and Weighted Moving Average (WMA).

The first approach, *Simple Moving Average (SMA)*, treats all models equally. For instance, when combining 10 models, each model is assigned a weight of $w_i = 0.1$. The SMA is formulated as:

$$M_{\text{avg}} = \frac{1}{N} \sum_{i=1}^{N} M_i. \tag{3}$$

The second approach, *Exponential Moving Average (EMA)*, emphasizes later models by assigning weights that decay exponentially, making EMA more sensitive to recent changes. The EMA is expressed recursively as:

$$M_{\text{avg}}^{(i)} = \alpha \cdot M_i + (1 - \alpha) \cdot M_{\text{avg}}^{(i-1)}, \ i \in [2, N], \tag{4}$$

Here, α, the smoothing factor (typically between 0 and 1), controls the balance between the current model M_i and the previous EMA result $M_{\text{avg}}^{(i-1)}$.

The third approach, *Weighted Moving Average (WMA)*, also prioritizes recent models but uses a distinct weighting scheme. In WMA, each model is assigned a specific weight, often increasing linearly for later models (e.g., $w_i = i$). The weighted sum is then normalized to compute the average, formulated as follows:

$$M_{\text{avg}} = \sum_{i=1}^{N} \frac{w_i}{w_{\text{sum}}} M_i, \quad w_{\text{sum}} = \sum_{i=1}^{N} w_i. \tag{5}$$

These methods offer flexible ways to combine models based on their recency and relevance. Choosing the right approach depends on the specific application and desired emphasis on newer data.

4 Experiments

In this section, we delve into the experimental core of our study, systematically addressing six critical questions surrounding model merging in the context of pre-training: 1) How does model merging

affect performance? 2) How do different merging methods affect final performance? 3) How to determine the optimal interval and number of weights to merge for various model sizes? 4) Do merged pre-trained models contribute to better downstream training? 5) Does model merging improve the stability of training? 6) What processes unfold during model merging? Through these experiments, we aim to provide comprehensive insights into model merging, offering practical guidance for its application and shedding light on its theoretical underpinnings.

4.1 How does model merging affect model performance?

Current learning rate schedule methods mainly involve constant learning rates or cosine annealing. In our model pre-training, we employed the Warmup-Stable-Decay (WSD) strategy Hu et al. [2024], which combines a constant learning rate phase with a subsequent cosine decay phase Loshchilov and Hutter [2017]. To explore the effects of model merging under different learning rate schedules, we conducted experiments during both constant learning rate phase and cosine dacay phase.

In the constant learning rate phase, we merged fully trained models of various sizes. As shown in Figure 1, the merged models exhibited significant performance improvements across multiple downstream tasks. For example, on the Humaneval benchmark, Seed-MoE-1.3B/13B improved from 31.1 to 36.6 points, and Seed-MoE-10B/100B increased from 54.3 to 61.6 points. While larger models showed less pronounced gains on certain benchmarks, such as BBH, this was likely due to the near-saturation of these metrics. Overall, the improvements were robust and consistent across model sizes.

Next, we performed model merging in the cosine annealing phase by collecting weights from the annealing stages of Seed-MoE-1.3B/13B, Seed-MoE-10B/100B, and Seed-MoE-15B/150B. As depicted in Figure 2, as the learning rate gradually decreased, the models converged steadily, with performance continuing to improve. Interestingly, at the early annealing stage, the results of `PMA` were comparable to those at the end of the annealing process. In some cases, particularly for larger models, the merged models even surpassed those naturally annealed.

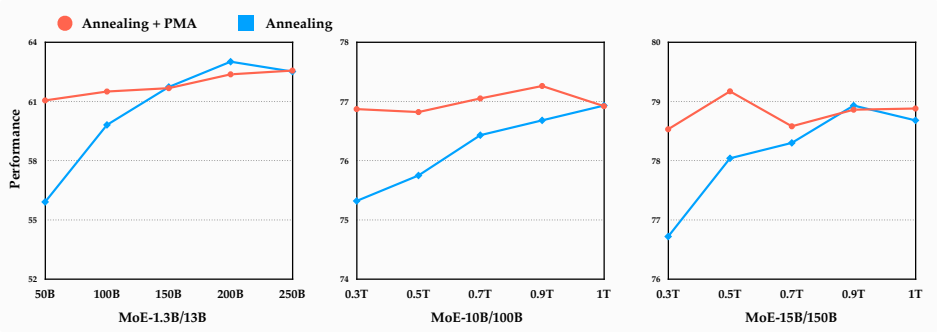

Figure 2: Comparison of overall performance for MoE models of varying sizes under annealing training, before and after model merging. The learning rate follows a cosine schedule during the annealing process. The x-axis shows the count of training tokens.

These findings raised a question: could we simplify the training process by using only the Warmup-Stable phases alongside `PMA`, skipping the decay phase, and avoiding learning rate adjustments? To investigate, we forked two training runs from the stable phase of Seed-MoE-1.3B/13B at 1.4T tokens. One continued with a constant learning rate, while another underwent annealing, each training for an additional 250B tokens. We then merged the models trained with the constant learning rate. As shown in Figure 3, early in training, the merged models significantly outperformed both the constant learning rate and annealed models. Even later, their performance was comparable to the annealed models.

This suggests that **pre-training with a constant learning rate, combined with model merging, can effectively match the performance of an annealed model at any point in the training process without the need for learning rate annealing**. This approach accelerates model validation and significantly reduces computational resource demands.

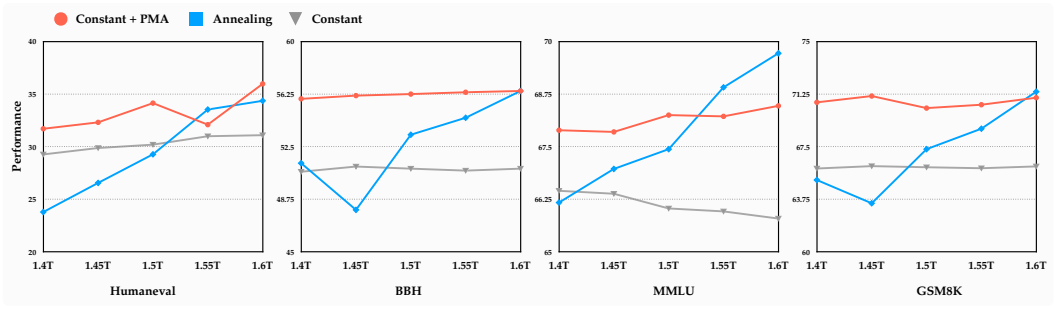

Figure 3: Comparison of downstream task performance between model merging results under stable training and the real annealed model. The x-axis shows the count of training tokens.

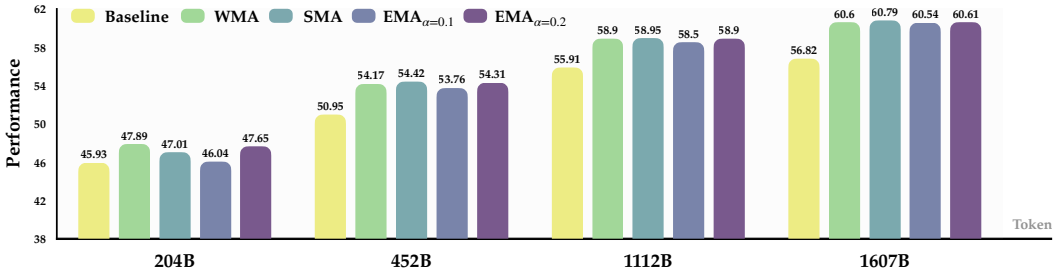

Figure 4: Impact of different model merging methods on final model performance.

4.2 How do different merging methods affect final performance?

In this section, we systematically evaluate how different merging strategies affect the performance of merged models. Specifically, we focus on three distinct approaches: EMA, WMA, and SMA. The EMA method employs exponentially decaying weights $w_i = \alpha(1-\alpha)^{N-i}$, giving higher importance to more recent checkpoints. WMA assigns linearly increasing weights $w_i = i$, also prioritizing more recent checkpoints. In contrast, SMA applies uniform weighting, treating all checkpoints equally regardless of their position in the training sequence.

We conducted experiments on Seed-MoE-1.3/13B and showed the results in Figure 4. At 204B training tokens, all merging methods enhanced model performance compared to the pre-merged model, but WMA delivered the best results. This suggests that in the early phases of training, when model weights undergo significant changes, assigning higher weights to checkpoints with more training tokens produces superior models. This is further supported by the fact that EMA$_{\alpha=0.2}$ outperforms EMA$_{\alpha=0.1}$. However, as training advances to later stages and model weights stabilize, the performance differences between merging methods diminish. **For its simplicity and stability, we primarily use SMA for model merging in subsequent experiments.**

4.3 How to determine the optimal interval and number of weights to merge for various model sizes?

Beyond the merging technique itself, two other factors may also affect the effectiveness of model merging: the interval V between selected models and the number of models N. We performed ablation studies on the Seed-MoE-1.3/13B model to investigate these effects, starting with the impact of the interval. As illustrated in the upper part of Figure 5, we fixed $N = 10$ and tested intervals of $V = 4$B, 8B, 16B, and 32B. Notably, at 204B with $V = 32$B, we reduced N to 6 due to insufficient models. In the early stage of training, at 204B tokens, merged results with $V = 16$B and $V = 32$B underperformed the baseline. This is likely because large intervals incorporated unstable weights from the initial training phase, leading to significant weight disparities and suboptimal outcomes. As training progressed and weights stabilized, the performance gap across different V settings gradually narrowed. **In practice, the optimal interval scales with model size, following these observed patterns: an interval of around 8B tokens for 1.3B/13B models, 4B tokens for 0.7B/7B models,**

and approximately 80B tokens for 10B/100B models. This aligns with the tendency of larger models to use larger batch sizes McCandlish et al. [2018].

Next, we set $V = 8$B and explored how the number of merged models N affects performance, testing $N = 3$, 6, 10, and 15. As shown in the lower part of Figure 5, early in training, incorporating more models introduced unstable weights, which reduced the performance of merged models. However, once training was complete, merging a larger number of models led to significant performance improvements. Notably, the overall performance for $N = 3$ was nearly 1 point lower than for $N = 15$. To strike a balance between computational cost and performance gains, we opted for $N = 10$ in further experiments.

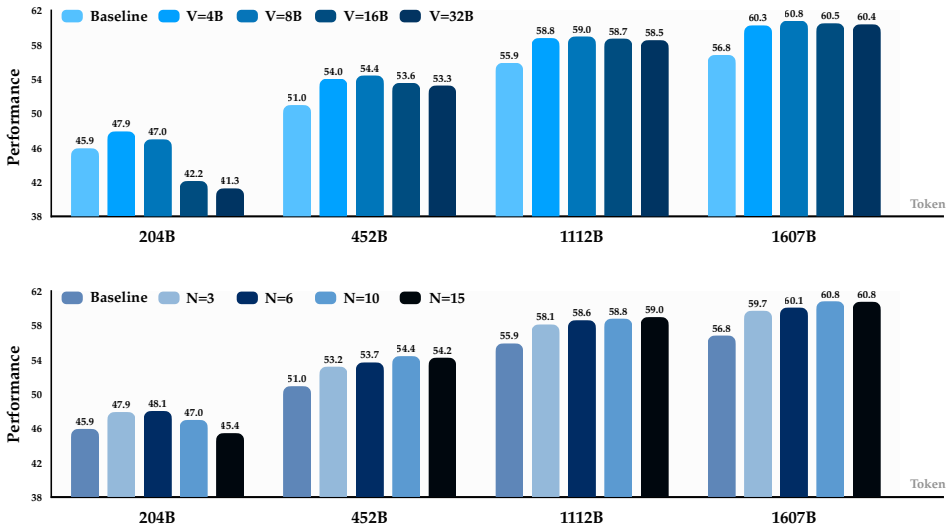

Figure 5: Impact of different model merging hyper-parameters on final model performance of the Seed-MoE-1.3/13B Model.

4.4 Do merged pre-trained models contribute to better downstream training?

A complete LLM training process typically involves multiple stages, which are pretraining, continual training (CT), supervised fine-tuning (SFT) and reinforcement learning (RL) in sequence. In light of the capacity of `PMA` to improve pretraining performance, we conjecture that merged pretrained models may similarly prove beneficial for downstream stages. To verify this hypothesis, we initialized downstream training with `PMA`, which we dubbed as `PMA-init`, and investigated its impacts over the baselines (which are initialized from their original checkpoints) for both CT and SFT stages.

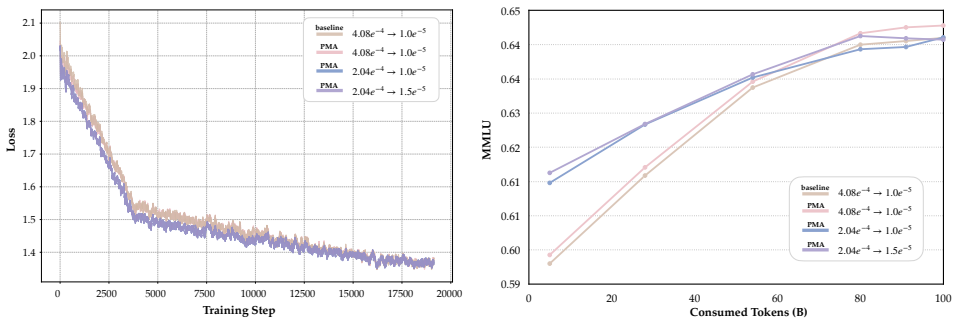

Figure 6: Comparisons of loss curves (left) and performance metrics (right) during CT stage with varying lr schedules, where a cosine scheduler is adopted to decay learning rate from lr_{peak} to lr_{end} (denoted as $lr_{peak} \rightarrow lr_{end}$). `PMA` and `baseline`, stand for whether our `PMA-init` technique is employed or not, respectively.

CT stage. We first conducted an ablation study to assess the sensitivity of the `PMA-init` of the CT stage with varying learning rate schedules. Specifically, we experimented with Seed-MoE-0.7B/7B models merged after stable training on approximately 1 trillion tokens. As illustrated in Figure 6 (left), the initialization weights obtained via `PMA` consistently achieved marginally lower loss at the initial training phase, against the baseline with the same training configuration. As training progresses, the loss values for models with different initialization weights converge to comparable levels. It's worth noting that in the loss curve, the purple line significantly overlaps with the blue line, and the brown line significantly overlaps with the pink line. Another observation is made in the Figure 6 (right), where evaluation on the MMLU benchmark reveals that the `PMA-init` models outperform the baseline early in training. While these models tend to retain a slight performance edge in later stages, their results on other tasks may be slightly suboptimal, leading to overall performance parity with the baseline. Experiments across varied learning rate schedules corroborate these findings, indicating that **models converge to similar performance levels by the end of training, and no extensive learning rate tuning is required for** `PMA-init`.

SFT stage. We next analyzed the impact of `PMA-init` on the SFT stage, where the detailed results can be found in the Appendix B. Although initialization with merged weights occasionally yields performance improvements, such gains are not consistently observed. Nonetheless, this approach does not adversely affect downstream training outcomes and may be a viable strategy for researchers seeking to enhance model performance.

4.5 Does model merging improve the training stability?

In large-scale LLM training, infrastructure issues are almost inevitable and often lead to training instability phenomena such as loss spikes or diverging. Specifically, a loss spike occurs when, at a specific point during the multi-stage training, the model's predictions deteriorate significantly compared to previous iterations. This phenomenon is often observed alongside gradient norm (GradNorm) explosion during backpropagation, which causes large weight updates and eventually lead to a irrecoverable spike in its loss function Cohen et al. [2021]. In the experiments detailed in Section 4.4, as illustrated in Figure 7 (left), we observed that a model initialized with `PMA-init` for SFT stage demonstrated a notably more stable GradNorm metric compared to the baseline. This stability is also evident in the reduced frequency of loss spikes relative to the baseline. Since applying `PMA-init` for downstream training does not impact the model's final performance and remains robust across different learning rates, we established a series of experiments to explore whether model merging could enhance training stability.

Given the extremely high expenses associated, it is unfeasible to conduct a direct analysis of training instability in LLM pre-training. Experiments Wortsman et al. [2024] show that small models using a relatively large learning rate will exhibit unstable training characteristics similar to those of large models. We thus reproduce the instability phenomena on small models to study the influence of our `PMA-init` on training stability. In one such experiment, we trained a 330M/3.3B MoE model from scratch using an exceptionally high learning rate of 6e-3. As shown in Figure 7 (right), the model overshot the optimal weights, resulting in unstable training and abrupt loss spikes as expected, and was irreversible to its original trajectory. To address this, we adopted `PMA-init` with three checkpoints saved before the training collapse happened, to resume the pre-training process. As depicted by the red line in Figure 7 (right), the resumed training process stabilized, successfully navigating past the point of the loss spike and continuing along its original training trajectory.

These results highlight that `PMA-init` **can reliably enhance the multi-stage training stability.** When a loss spike occurs, one can merge the model checkpoints from before the spike and resume training from that point. This approach provides an alternative solution to avoid retraining the model from scratch, thereby substantially reducing the waste of computational resources.

4.6 Investigating the Mechanisms of Model Merging

To gain deeper insight into the underlying mechanisms that enable model merging to be effective, we provide both qualitative and quantitative analyses, employing mathematical derivations and visualizations of weight distributions. Due to space limitations, this section focuses solely on the qualitative analysis. For the quantitative analysis, please refer to Appendix C for more details.

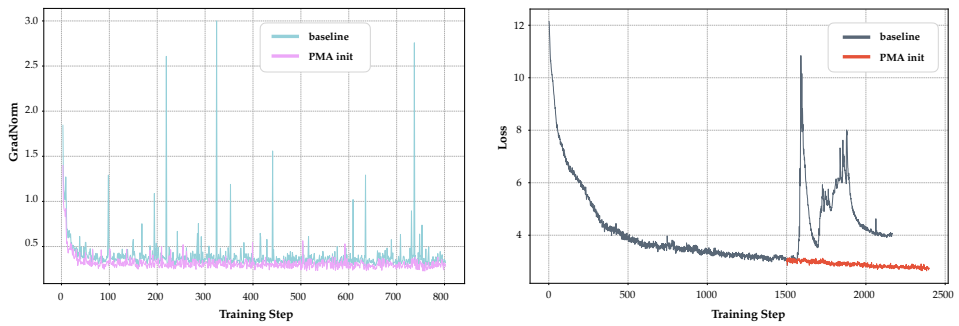

Figure 7: Left: GradNorm comparisons for SFT training initialized with `PMA-init`. Right: Comparison of pre-training loss curves between resuming with `PMA-init` and the original training.

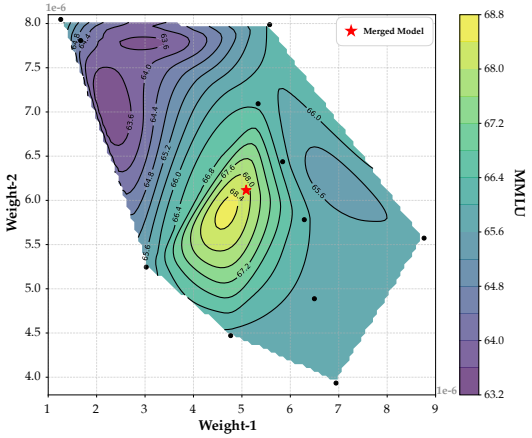

Figure 8: Visualization of MMLU score contour lines, comparing the weights of an original model with those of a merged model. Black dots represent the parameter locations of various individual model checkpoints.

We selected several checkpoints from the pre-training of Seed-MoE-1.3B/13B and visualized the average distribution of two selected parameters from a specific layer. Using these points, we generated contour lines for MMLU scores, as illustrated in Figure 8. The weight positions of various individual models are marked as black dots. These dots are distributed along the MMLU score contours, revealing a discernible "complementary" pattern. The averaged weight position (representative of the merged model) is often situated closer to a region of higher MMLU scores (a better optimum) than many individual model checkpoints. This visualization also provides an intuitive explanation for why model merging yields diminished improvements when models are annealed to a very low learning rate: at such a stage, the models to be merged are already tightly converged within a specific local optimum. Merging them essentially averages points within this already narrow basin, making it unlikely to escape to a significantly better or different optimal region.

5 Conclusion

This research pioneers a deeper exploration of model merging within the challenging pre-training stage of large-scale models. By training a spectrum of MoE and Dense models and performing rigorous ablations, we established that merging checkpoints from stable training phases not only yields significant performance gains and predicts annealing but also streamlines development and reduces costs. Our work provides concrete guidance on merging strategies, optimal parameters, and downstream applications, alongside insights into the underlying mechanisms. These contributions equip the open-source community with the knowledge and tools for more efficient model development through pre-training merging.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

Takuya Akiba, Makoto Shing, Yujin Tang, Qi Sun, and David Ha. Evolutionary optimization of model merging recipes. *Nature Machine Intelligence*, pages 1–10, 2025.

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, et al. Program synthesis with large language models. *arXiv preprint arXiv:2108.07732*, 2021.

Xiao Bi, Deli Chen, Guanting Chen, Shanhuang Chen, Damai Dai, Chengqi Deng, Honghui Ding, Kai Dong, Qiushi Du, Zhe Fu, et al. Deepseek llm: Scaling open-source language models with longtermism. *arXiv preprint arXiv:2401.02954*, 2024.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde De Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021.

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. *arXiv preprint arXiv:1803.05457*, 2018.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

Jeremy Cohen, Simran Kaur, Yuanzhi Li, J Zico Kolter, and Ameet Talwalkar. Gradient descent on neural networks typically occurs at the edge of stability. In *International Conference on Learning Representations*, 2021.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021.

Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. DROP: A reading comprehension benchmark requiring discrete reasoning over paragraphs. In Jill Burstein, Christy Doran, and Thamar Solorio, editors, *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics*, pages 2368–2378, 2019.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models. *arXiv e-prints*, pages arXiv–2407, 2024.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

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

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.

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

Yuzhen Huang, Yuzhuo Bai, Zhihao Zhu, Junlei Zhang, Jinghan Zhang, Tangjun Su, Junteng Liu, Chuancheng Lv, Yikai Zhang, jiayi lei, Yao Fu, Maosong Sun, and Junxian He. C-eval: A multi-level multi-discipline chinese evaluation suite for foundation models. In *Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2023.

J Stuart Hunter. The exponentially weighted moving average. *Journal of quality technology*, 18(4): 203–210, 1986.

Gabriel Ilharco, Marco Tulio Ribeiro, Mitchell Wortsman, Ludwig Schmidt, Hannaneh Hajishirzi, and Ali Farhadi. Editing models with task arithmetic. In *International Conference on Learning Representations*, 2023.

Xisen Jin, Xiang Ren, Daniel Preotiuc-Pietro, and Pengxiang Cheng. Dataless knowledge fusion by merging weights of language models. In *The International Conference on Learning Representations*, 2023.

FR Johnston, John E Boyland, Maureen Meadows, and E Shale. Some properties of a simple moving average when applied to forecasting a time series. *Journal of the Operational Research Society*, 50 (12):1267–1271, 1999.

Alexia Jolicoeur-Martineau, Emy Gervais, Kilian Fatras, Yan Zhang, and Simon Lacoste-Julien. Population parameter averaging (papa). *arXiv preprint arXiv:2304.03094*, 2023.

Mandar Joshi, Eunsol Choi, Daniel Weld, and Luke Zettlemoyer. TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension. In Regina Barzilay and Min-Yen Kan, editors, *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, pages 1601–1611, 2017.

Jean Kaddour. Stop wasting my time! saving days of imagenet and BERT training with latest weight averaging. In *Advances in Neural Information Processing Systems Workshop*, 2022.

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

Tao Li, Zhehao Huang, Qinghua Tao, Yingwen Wu, and Xiaolin Huang. Trainable weight averaging: Efficient training by optimizing historical solutions. In *The Eleventh International Conference on Learning Representations*, 2022.

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

Deyuan Liu, Zecheng Wang, Bingning Wang, Weipeng Chen, Chunshan Li, Zhiying Tu, Dianhui Chu, Bo Li, and Dianbo Sui. Checkpoint merging via bayesian optimization in llm pretraining. *arXiv preprint arXiv:2403.19390*, 2024b.

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

Haipeng Luo, Qingfeng Sun, Can Xu, Pu Zhao, Jian-Guang Lou, Chongyang Tao, Xiubo Geng, Qingwei Lin, Shifeng Chen, Yansong Tang, and Dongmei Zhang. Wizardmath: Empowering mathematical reasoning for large language models via reinforced evol-instruct. In *International Conference on Learning Representations*, 2025.

Michael S Matena and Colin A Raffel. Merging models with fisher-weighted averaging. *Advances in Neural Information Processing Systems*, 35:17703–17716, 2022.

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

Marcus B Perry. The weighted moving average technique. *Wiley Encyclopedia of Operations Research and Management Science*, 2010.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R Bowman. Gpqa: A graduate-level google-proof q&a benchmark. In *First Conference on Language Modeling*, 2024.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. Winogrande: An adversarial winograd schema challenge at scale. *Communications of the ACM*, 64(9):99–106, 2021.

Sunny Sanyal, Atula Tejaswi Neerkaje, Jean Kaddour, Abhishek Kumar, and sujay sanghavi. Early weight averaging meets high learning rates for LLM pre-training. In *First Conference on Language Modeling*, 2024.

ByteDance Seed, Yufeng Yuan, Yu Yue, Mingxuan Wang, Xiaochen Zuo, Jiaze Chen, Lin Yan, Wenyuan Xu, Chi Zhang, Xin Liu, et al. Seed-thinking-v1. 5: Advancing superb reasoning models with reinforcement learning. *arXiv preprint arXiv:2504.13914*, 2025.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Y Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Noam Shazeer, *Azalia Mirhoseini, *Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. In *International Conference on Learning Representations*, 2017.

Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc Le, Ed Chi, Denny Zhou, and Jason Wei. Challenging BIG-bench tasks and whether chain-of-thought can solve them. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki, editors, *Findings of the Association for Computational Linguistics*, pages 13003–13051, 2023.

Gemini Team, Petko Georgiev, Ving Ian Lei, Ryan Burnell, Libin Bai, Anmol Gulati, Garrett Tanzer, Damien Vincent, Zhufeng Pan, Shibo Wang, et al. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. *arXiv preprint arXiv:2403.05530*, 2024.

Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, et al. Mmlu-pro: A more robust and challenging multi-task language understanding benchmark. In *The Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2024.

Jason Wei, Maarten Bosma, Vincent Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M. Dai, and Quoc V Le. Finetuned language models are zero-shot learners. In *International Conference on Learning Representations*, 2022.

Colin White, Samuel Dooley, Manley Roberts, Arka Pal, Benjamin Feuer, Siddhartha Jain, Ravid Shwartz-Ziv, Neel Jain, Khalid Saifullah, Sreemanti Dey, Shubh-Agrawal, Sandeep Singh Sandha, Siddartha Venkat Naidu, Chinmay Hegde, Yann LeCun, Tom Goldstein, Willie Neiswanger, and Micah Goldblum. Livebench: A challenging, contamination-limited LLM benchmark. In *International Conference on Learning Representations*, 2025.

Mitchell Wortsman, Peter J Liu, Lechao Xiao, Katie E Everett, Alexander A Alemi, Ben Adlam, John D Co-Reyes, Izzeddin Gur, Abhishek Kumar, Roman Novak, Jeffrey Pennington, Jascha Sohl-Dickstein, Kelvin Xu, Jaehoon Lee, Justin Gilmer, and Simon Kornblith. Small-scale proxies for large-scale transformer training instabilities. In *International Conference on Learning Representations*, 2024.

Can Xu, Qingfeng Sun, Kai Zheng, Xiubo Geng, Pu Zhao, Jiazhan Feng, Chongyang Tao, Qingwei Lin, and Daxin Jiang. WizardLM: Empowering large pre-trained language models to follow complex instructions. In *The International Conference on Learning Representations*, 2024.

Prateek Yadav, Derek Tam, Leshem Choshen, Colin Raffel, and Mohit Bansal. TIES-merging: Resolving interference when merging models. In *Conference on Neural Information Processing Systems*, 2023.

Prateek Yadav, Tu Vu, Jonathan Lai, Alexandra Chronopoulou, Manaal Faruqui, Mohit Bansal, and Tsendsuren Munkhdalai. What matters for model merging at scale? *arXiv preprint arXiv:2410.03617*, 2024.

Aiyuan Yang, Bin Xiao, Bingning Wang, Borong Zhang, Ce Bian, Chao Yin, Chenxu Lv, Da Pan, Dian Wang, Dong Yan, et al. Baichuan 2: Open large-scale language models. *arXiv preprint arXiv:2309.10305*, 2023.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, et al. Qwen2. 5 technical report. *arXiv preprint arXiv:2412.15115*, 2024a.

Enneng Yang, Li Shen, Guibing Guo, Xingwei Wang, Xiaochun Cao, Jie Zhang, and Dacheng Tao. Model merging in llms, mllms, and beyond: Methods, theories, applications and opportunities. *arXiv preprint arXiv:2408.07666*, 2024b.

Enneng Yang, Zhenyi Wang, Li Shen, Shiwei Liu, Guibing Guo, Xingwei Wang, and Dacheng Tao. Adamerging: Adaptive model merging for multi-task learning. In *International Conference on Learning Representations*, 2024c.

Le Yu, Bowen Yu, Haiyang Yu, Fei Huang, and Yongbin Li. Language models are super mario: Absorbing abilities from homologous models as a free lunch. In *International Conference on Machine Learning*, 2024.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Tiantian Fan, Gaohong Liu, Lingjun Liu, Xin Liu, et al. Dapo: An open-source llm reinforcement learning system at scale. *arXiv preprint arXiv:2503.14476*, 2025.

Yufeng Yuan, Qiying Yu, Xiaochen Zuo, Ruofei Zhu, Wenyuan Xu, Jiaze Chen, Chengyi Wang, TianTian Fan, Zhengyin Du, Xiangpeng Wei, et al. Vapo: Efficient and reliable reinforcement learning for advanced reasoning tasks. *arXiv preprint arXiv:2504.05118*, 2025.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. Hellaswag: Can a machine really finish your sentence? *arXiv preprint arXiv:1905.07830*, 2019.

Wei Zhao, Mingyue Shang, Yang Liu, Liang Wang, and Jingming Liu. Ape210k: A large-scale and template-rich dataset of math word problems. *arXiv preprint arXiv:2009.11506*, 2020.

Wanjun Zhong, Ruixiang Cui, Yiduo Guo, Yaobo Liang, Shuai Lu, Yanlin Wang, Amin Saied, Weizhu Chen, and Nan Duan. AGIEval: A human-centric benchmark for evaluating foundation models. In Kevin Duh, Helena Gomez, and Steven Bethard, editors, *Findings of the Association for Computational Linguistics: NAACL*, pages 2299–2314, 2024.

Yuyan Zhou, Liang Song, Bingning Wang, and Weipeng Chen. MetaGPT: Merging large language models using model exclusive task arithmetic. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen, editors, *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1711–1724, 2024.

A The Effect of Model Merging in Dense Models

We also conducted model merging experiments on Dense architecture models, which mirror the exact architecture of LLaMA 3.1 [Grattafiori et al., 2024], ranging from small Dense-411M models to large Dense-70B models. Since the 411M and 2B models were not sufficiently trained, we used a configuration of N=6 for merging, with weight intervals (V) of 2B and 5B tokens, respectively. For the 8B and 70B models, which were trained more thoroughly, we used N=10, with V values of 15B and 40B for merging. As shown in Figure 9, models of different sizes achieved significant improvements on downstream tasks after model merging. Notably, the performance gains of larger models were not smaller than those of smaller models. Specifically, Dense-70B improved from 50.6 to 57.9 on Humaneval and from 85.9 to 91.3 on GSM8K. This further validates the robustness and generalization ability of PMA, demonstrating that it can work across different model architectures and sizes.

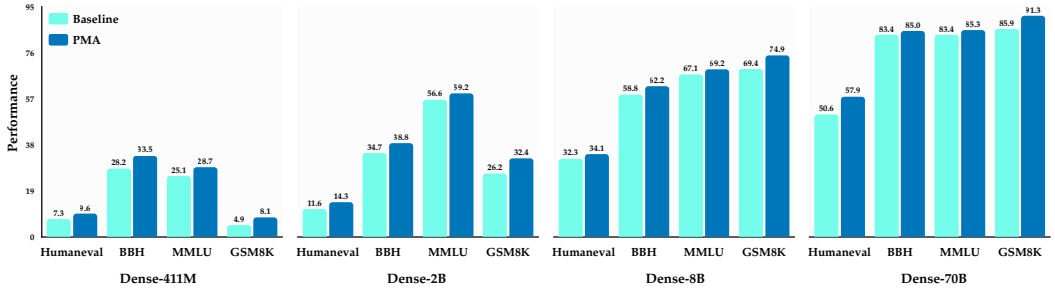

Figure 9: Comparison of downstream task performance for dense models of varying sizes under stable training, before and after model merging.

B Model Merging at the CT Stage for Supervised Fine-Tuning

PMA-init can be integrated during the CPT annealing phase, where PMA weights are merged to provide a strong starting point for fine-tuning. We conducted an ablation study to assess the sensitivity of the PMA-init during the SFT stage to varying learning rate schedules. This study included experiments on merged MoE-15B/150B models following stable training on approximately 16T tokens, as well as after further training on 1T tokens with cosine annealing. We conducted SFT training for 220M tokens using both the original weights and PMA-init weights. For the original weights, we used a cosine learning rate schedule with an initial learning rate of 2e-5 and an end learning rate of 2e-6. For the PMA-init weights, we used cosine schedules with initial learning rates of 1e-5, 2e-5, and 4e-5, all with an end learning rate of 2e-6. We evaluated the trained models using Open-Benchmark, which includes MMLU Hendrycks et al. [2021], LiveBench White et al. [2025], AMC-2023, GPQA Rein et al. [2024] and LiveCodeBench White et al. [2025], as well as our in-house evaluation set comprising OOD, Reasoning, and Instruction Following assessments. As shown in Table 1, with the same learning

Table 1: Comparisons of performance metrics during SFT stage with varying lr schedules, where a cosine scheduler is adopted to decay learning rate from lr_{peak} to lr_{end} (denoted as $lr_{peak} \rightarrow lr_{end}$). PMA and `baseline`, stand for whether our `PMA-init` technique is employed or not, respectively. IF refers to Instruction Following.

Model	Open-Benchmark					In-house Evaluation		
	MMLU	LiveBench	AMC-2023	GPQA	LiveCodeBench	OOD	Reasoning	IF
Baseline$_{2e^{-5}\rightarrow 2e^{-6}}$	86.8	50.5	61.0	**55.2**	**39.7**	32.6	32.1	36.3
PMA$_{2e^{-5}\rightarrow 2e^{-6}}$	87.1	52.0	64.0	54.0	39.4	**34.7**	**34.0**	**38.8**
PMA$_{1e^{-5}\rightarrow 2e^{-6}}$	**87.2**	**53.2**	**65.5**	54.4	**39.7**	33.8	33.2	37.3
PMA$_{4e^{-5}\rightarrow 2e^{-6}}$	87.0	51.3	61.4	54.0	39.2	31.8	32.6	37.2

rate, `PMA-init` significantly outperformed the baseline on both Open-Benchmark and our in-house evaluations. Notably, on the in-house evaluation set, we observed improvements of over two points in OOD and Instruction Following, and a 1.9-point increase in Reasoning. In the other two experiments

with different learning rates, we also saw some degree of improvement compared to the baseline, especially with $\texttt{PMA}_{1e^{-5}\to 2e^{-6}}$, which showed gains of 2.7 points on Livebench and 4.5 points on AMC-2023.

However, we were unable to replicate such significant gains in subsequent experiments with other model sizes, although it did not negatively impact the final downstream model performance. Therefore, as a low-cost approach, PMA-init is worth trying to obtain a more powerful downstream model.

C Quantitative Analysis on Model Merging

We begin with a second-order Taylor expansion of the loss function $L(\theta)$ around an optimal parameter set θ^*:

$$L(\theta) \approx L(\theta^*) + (\theta - \theta^*)^T \nabla L(\theta^*) + \frac{1}{2}(\theta - \theta^*)^T H (\theta - \theta^*), \tag{6}$$

where H is the Hessian matrix of the loss function evaluated at θ^* (the matrix of second partial derivatives), which captures curvature information. Since θ^* is an optimal point, the gradient $\nabla L(\theta^*)$ is zero. Thus, the expansion simplifies to:

$$L(\theta) \approx L(\theta^*) + \frac{1}{2}(\theta - \theta^*)^T H (\theta - \theta^*). \tag{7}$$

Consider k sets of model parameters $\theta_1, \theta_2, \ldots, \theta_k$. Let the deviation vector of each model i from the optimal parameters be $\delta_i = \theta_i - \theta^*$. The loss for each model i can then be approximated as:

$$L(\theta_i) \approx L(\theta^*) + \frac{1}{2}\delta_i^T H \delta_i. \tag{8}$$

The average loss of these k individual models is:

$$\frac{1}{k}\sum_{i=1}^{k} L(\theta_i) \approx L(\theta^*) + \frac{1}{2k}\sum_{i=1}^{k} \delta_i^T H \delta_i. \tag{9}$$

The parameters of the merged model are $\theta_{\text{avg}} = \frac{1}{k}\sum_{i=1}^{k}\theta_i$. The deviation of this merged model from the optimal parameters is $\theta_{\text{avg}} - \theta^* = \frac{1}{k}\sum_{i=1}^{k}\delta_i$. The loss for the merged model is approximated by:

$$L(\theta_{\text{avg}}) \approx L(\theta^*) + \frac{1}{2}\left(\frac{1}{k}\sum_{i=1}^{k}\delta_i\right)^T H \left(\frac{1}{k}\sum_{i=1}^{k}\delta_i\right). \tag{10}$$

Expanding the quadratic term:

$$\frac{1}{2}\left(\frac{1}{k}\sum_{i=1}^{k}\delta_i\right)^T H \left(\frac{1}{k}\sum_{i=1}^{k}\delta_i\right) = \frac{1}{2k^2}\sum_{i=1}^{k}\sum_{j=1}^{k}\delta_i^T H \delta_j. \tag{11}$$

This can be rewritten by separating diagonal and off-diagonal terms:

$$\frac{1}{2k^2}\left(\sum_{i=1}^{k}\delta_i^T H \delta_i + \sum_{i=1}^{k}\sum_{j\neq i}\delta_i^T H \delta_j\right). \tag{12}$$

For the merged model to have a lower loss than the average loss of the individual models, i.e., $L(\theta_{\text{avg}}) < \frac{1}{k}\sum_{i=1}^{k} L(\theta_i)$, the following condition must hold:

$$\frac{1}{2k^2}\left(\sum_{i=1}^{k}\delta_i^T H \delta_i + \sum_{i=1}^{k}\sum_{j\neq i}\delta_i^T H \delta_j\right) < \frac{1}{2k}\sum_{i=1}^{k}\delta_i^T H \delta_i. \tag{13}$$

Multiplying by $2k^2$ and rearranging terms, we get:

$$\sum_{i=1}^{k}\delta_i^T H \delta_i + \sum_{i=1}^{k}\sum_{j\neq i}\delta_i^T H \delta_j < k\sum_{i=1}^{k}\delta_i^T H \delta_i. \tag{14}$$

Which simplifies to:

$$\sum_{i=1}^{k}\sum_{j\neq i}\delta_i^T H\delta_j < (k-1)\sum_{i=1}^{k}\delta_i^T H\delta_i. \tag{15}$$

Assuming H is a positive definite matrix (which is generally true around a local minimum), then each term $\delta_i^T H\delta_i > 0$. The inequality is more easily satisfied if the off-diagonal terms $\delta_i^T H\delta_j$ (for $i \neq j$) are predominantly negative. This "negative correlation" in the context of the Hessian means that the deviation vectors point in somewhat opposing directions relative to the curvature of the loss landscape. This mathematical analysis can be intuitively interpreted as follows: 1. The effectiveness of model weight merging stems from the fact that different model checkpoints, representing different points in the training trajectory, have explored different local regions or directions within the parameter space. 2. When these explorations exhibit a degree of "complementarity" concerning the geometric structure of the loss function (captured by the Hessian and the cross-terms $\delta_i^T H\delta_j$), their average can position the merged model closer to an optimal point than the individual models might be on average. 3. This helps explain why merging models, particularly those from a stable yet ongoing training phase, often improves performance. The averaging process can smooth out idiosyncrasies of individual checkpoints. This analysis suggests that weight merging is not merely a simple averaging of parameters but rather a process that can leverage the geometric structure of the loss landscape and the diversity among the models being merged.

D Limitations

In our study, we thoroughly investigated the potential of model merging in the pre-training phase, offering significant advantages for teams working on large-scale model pre-training to pursue more daring explorations. This is due to the fact that model merging can replicate the benefits of simulated annealing, greatly shortening the exploration period during pre-training. While our experiments were extensive, certain aspects still remain open for deeper research.

In our experiments, we defaulted to using the optimal learning rate derived from the scaling law for model training, without extensively exploring the impact of learning rate on model merging. In our practice, we believe that training with a higher learning rate could lead to a better model through model merging, which aligns with the findings in Sanyal et al. [2024]. However, due to the high computational cost, we did not further quantify the impact of learning rate on model merging in a more detailed manner.

Additionally, this paper primarily focuses on the application of model merging in pre-training. In reality, due to innovations in RL algorithms [Yu et al., 2025, Yuan et al., 2025, Shao et al., 2024], RL training has become more stable and often involves longer training cycles, during which a series of adjacent weights can be obtained. This paper does not investigate model merging in the context of post-training scenarios, and we leave this aspect for future research.

E Societal Impact

The development of large-scale language models, while technologically promising, carries significant societal implications that necessitate careful consideration. Our research on model merging during pre-training offers advancements that not only push the technical frontier but also contribute positively to the responsible development of artificial intelligence.

E.1 Potential Benefits

The primary contribution of our work is a substantial increase in the efficiency of the LLM pre-training process. The Pre-trained Model Averaging (PMA) technique significantly reduces the computational resources and time required to develop high-performance models. This efficiency has several downstream societal benefits:

Accelerating Innovation and Accessibility: By lowering the immense costs traditionally associated with training state-of-the-art models, our methods can help democratize access to large-scale AI development. This allows smaller research labs, academic institutions, and companies to contribute to the field, fostering broader innovation.

Enhancing Applications: The accelerated development cycles enabled by our work can lead to faster improvements in critical NLP applications. This includes creating more effective and accessible educational tools, developing more nuanced and accurate communication technologies (such as real-time translation), and building more capable assistive technologies for individuals with disabilities.

E.2 Risks and Mitigation Strategies

We also recognize the potential risks associated with large-scale models and believe our work provides avenues for their mitigation.

Computational Energy Costs and Environmental Impact: The training of LLMs is an energy-intensive process with a considerable environmental footprint. Our findings directly address this concern. By demonstrating that merged checkpoints can match or exceed the performance of models that have undergone a full, lengthy annealing phase, our technique provides a method to curtail total training time. Furthermore, the ability of PMA-init to recover from training instabilities prevents the wasteful process of restarting training from scratch. These efficiencies translate directly into reduced energy consumption and a smaller carbon footprint for model development.

Algorithmic Bias: Large language models can perpetuate and amplify existing societal biases present in their training data. While our research does not introduce a new method for bias detection, the efficiency it creates is a critical enabler for more rigorous and responsible AI development. By reducing the cost of each training and evaluation cycle, our methods allow development teams to:

Iterate More Frequently: Conduct more frequent and thorough testing for biases within the model.

Reallocate Resources: Dedicate computational and financial resources saved from training towards crucial mitigation tasks, such as curating higher-quality, more diverse datasets and implementing sophisticated fairness-aware fine-tuning techniques.

In summary, the model merging techniques presented in this paper offer a practical path toward a more efficient, sustainable, and responsible ecosystem for AI development. By addressing the core challenge of computational cost, we empower the research community to not only build more powerful models but also to invest more deeply in making them safe and beneficial for society.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The main claims made in the abstract and introduction of our paper accurately reflect the paper's contributions and scope.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The paper discusses the limitations of the work, see Appendix D.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

Justification: The paper provide the full set of assumptions and a complete proof.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: The paper fully discloses all the information needed to reproduce the main experimental results, and although the code and data are not provided, it does not affect the main conclusions of the paper.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [No]

Justification: Although the code and data are not provided, it does not affect the main conclusions of the paper.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: The paper specify all the training and test details.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

   Justification: The paper does not report error bars, but experiments were conducted across multiple model architectures and sizes to validate the robustness of the method.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).

- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [No]

   Justification: We primarily used NVIDIA H-series GPUs for training and evaluation.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The research conducted in the paper conforms in every respect to the NeurIPS Code of Ethics.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [No]

    Justification: The paper does not address societal impact, but it significantly reduces training costs, facilitating the development of better models.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.

- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [No]

    Justification: The methods primarily discussed in our paper do not involve data or models with a high risk of misuse.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We have appropriately cited the papers relevant to our work.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.

- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [NA]

    Justification: The paper does not release new assets.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.

- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

    Answer: [NA]

    Justification: The core method development in this research does not involve LLMs.

    Guidelines:

    - The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.