Practical Guidelines for Model Merging in LLMs Pre-Training

Giuseppe Curci[1] **Stefano Simonazzi**[2] **Andrea Molinari**[3] **Andrea Zugarini**[2]

[1]Okkam, Italy [2]Villanova.ai, Italy [3]University of Trento, Italy

curci@okkam.it, andrea.molinari@unitn.it,
{ssimonazzi, azugarini}@villanova.ai

Abstract

Model merging is widely used to combine fine-tuned models trained with different data distributions, tasks, or hyperparameters, yet its role during LLM pre-training remains under-explored. We systematically study checkpoint merging across training phases, focusing on the transition from stable to decaying learning rates. Across multiple scales, we find that simple averaging methods consistently improve performance during stable learning rate regimes, but gains sharply diminish during decay. We link this effect to reduced checkpoint diversity and show that merging effectiveness correlates with parameter-space variation. Strategies such as synthetic variability, task-vector merging, and cross-run merging yield only modest improvements. Our results provide practical insights on when merging is most effective in large-scale pre-training.

1 Introduction

Model merging - combining multiple model checkpoints into a single model - has become a common technique to improve performance in fine-tuned language models. Methods such as linear averaging (Wortsman et al., 2022), exponential moving averages, and task-vector approaches (Ilharco et al., 2022; Yu et al., 2024; Yadav et al., 2023) have been applied across different tasks showing consistent gains. However, the effectiveness of these methods during Large Language Model (LLM) pre-training remains mostly unexplored, with prior works primarily focusing on stable learning rate regimes (Li et al., 2025). In this work, we study checkpoint merging in the context of developing a new family of LLMs, denoted VILLANOVA 2B, systematically investigating merging across training phases, especially the transition from stable to decaying learning rates. We find that simple averaging improves performance during stable learning rate phases but provides diminishing returns under decay, a phenomenon that we link to reduced checkpoint diversity and lower parameter-space variability. We further explore strategies such as synthetic checkpoint perturbations, task-vector merging, and cross-run merging, showing that practical gains are modest in low-variability regimes, and extend our findings to additional models to provide guidance on when and how merging is most effective during pre-training. Code[1] and checkpoints[2] are publicly available.

2 Related Works

Model merging is a broad line of research centered around weight interpolation techniques, most notably popularized by model soups (Wortsman et al., 2022). A variety of strategies have been proposed to merge models trained under different conditions: some approaches combine independently fine-tuned models that differ in training data or hyperparameters (Wortsman et al., 2022; Jang et al., 2024), while others rely on task vectors computed relative to a shared pre-trained initialization (Ilharco et al., 2022; Yadav et al., 2023; Yu et al., 2024). While much of the early literature focuses on relatively small models, subsequent work has shown that model merging remains effective at larger scales (Yadav et al., 2024; Hitit et al., 2025; Liu et al., 2024b).

Despite this progress, the application of model merging during language model pre-training has received comparatively limited attention, even though weight averaging techniques such as stochastic weight averaging (Izmailov et al., 2018) and exponential moving average have been shown to be effective in other training settings. LAWA (Kaddour, 2022) demonstrated that averaging checkpoints along a single training trajectory can accelerate training for models such as RoBERTa-

[1]https://github.com/giuseppecurci/Merge-LLM-Pre-Training/tree/main.

[2]https://huggingface.co/VillanovaAI/Villanova-2B-checkpoints.

Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 6: Industry Track), pages 1519–1532

Base (Liu et al., 2019). Follow-up work further suggests that checkpoint averaging is most effective under high learning rates (Sanyal et al., 2023). Several recent LLM training reports—both at large scale (Grattafiori et al., 2024; Liu et al., 2024a) and smaller scale (OLMo et al., 2024; Lozhkov et al., 2025)—mention the use of checkpoint merging during pre-training or mid-training, but do not systematically analyze its benefits or limitations.

Closest to our work, Li et al. (2025) provide the first focused study of model merging during LLM pre-training, analyzing its effects mostly under a stable learning rate regime using a limited set of methods. In contrast, we study model merging across different phases of pre-training, with particular emphasis on learning rate annealing. We show that the diminishing gains from merging during later training stages are closely tied to reduced checkpoint diversity, and we explore a broader range of merging strategies to mitigate this effect.

3 Methodology

3.1 Models

Most of our experiments are conducted on two models, denoted VILLANOVADEEP and VILLANOVAWIDE. As no meaningful differences are observed between them within the scope of our merging experiments, we refer to both collectively as VILLANOVA 2B. All the checkpoints were developed during the pre-training of VILLANOVA 2B models family[3]. The stable learning rate is set to 3×10^{-4}. Additional details on architecture and training are provided in Appendix A. We extend our analysis in Section 4.3 to additional model families, including the PYTHIA suite (Biderman et al., 2023), OLMO 3 7B (Olmo et al., 2025), and SMOLLM3 (Lozhkov et al., 2025).

3.2 Evaluation

We evaluate all models on a diverse suite of reasoning and commonsense understanding benchmarks using the `lm-evaluation-harness` library (Biderman et al., 2024). The evaluation set includes ARC-Easy (Clark et al., 2018), HellaSwag (Zellers et al., 2019), PIQA (Bisk et al., 2020), SciQ (Welbl et al., 2017), and Winogrande (Sakaguchi et al., 2021), covering multiple-choice scientific, physical, and commonsense reasoning tasks. To assess multilingual generalization, we additionally report results on XCOPA (Italian) (Ponti et al., 2020) and XNLI (Conneau et al., 2018) to cover also Italian, German, Spanish and French.

All evaluations are conducted in a 5-shot setting and results are reported as the average accuracy across benchmarks. This evaluation protocol is chosen to provide a broad view of model performance across reasoning and multilingual tasks.

3.3 Model Merging

We evaluate different merging strategies that have been previously studied in the context of pre-training (Li et al., 2025), namely linear averaging (*Linear*), exponential moving average (*EMA*), and weighted moving average (*WMA*). In addition, we assess the effectiveness of task vector–based merging using *TIES* (Yadav et al., 2023) and non-linear methods using *SLERP* (Shoemake, 1985).

Based on the findings of Li et al. (2025), we merge $N = 15$ checkpoints sampled at 4-billion-token intervals. This choice reflects our focus on the intermediate and late stages of pre-training, where the benefits of merging plateau at this scale.

Standard Linear Interpolation. Given checkpoints $\{\boldsymbol{\theta}_i\}_{i=1}^{N}$, linear merging defines

$$w_i = i, \qquad \boldsymbol{\theta}_{\text{merge}} = \frac{\sum_{i=1}^{N} w_i \boldsymbol{\theta}_i}{\sum_{i=1}^{N} w_i}. \tag{1}$$

Methods differ only in the choice of weights. Linear uses uniform weights $w_i = 1$; WMA assigns increasing importance to later checkpoints with $w_i = i$; and EMA emphasizes recent checkpoints via exponentially decaying weights $w_i = \alpha(1-\alpha)^{N-i}$, where $\alpha \in (0, 1]$ controls the decay. Li et al. (2025) evaluate $\alpha \in \{0.2, 0.3\}$, observe negligible differences and thus we fix $\alpha = 0.2$.

Non-linear Interpolation. SLERP performs interpolation on the hypersphere rather than in Euclidean space. Given two checkpoints $\boldsymbol{\theta}_1$ and $\boldsymbol{\theta}_2$, SLERP interpolates as

$$\text{SLERP}(t) = \frac{\sin((1-t)\Omega)}{\sin \Omega}\, \boldsymbol{\theta}_1 + \frac{\sin(t\Omega)}{\sin \Omega}\, \boldsymbol{\theta}_2,$$

where $t \in [0, 1]$ and $\Omega = \arccos\left(\frac{\langle \boldsymbol{\theta}_1, \boldsymbol{\theta}_2 \rangle}{\|\boldsymbol{\theta}_1\| \|\boldsymbol{\theta}_2\|}\right)$ is the angle between them. Since SLERP operates on pairs of models, we iteratively merge checkpoints along the training trajectory, starting from the first two and then merging each subsequent checkpoint with the previously merged model. We fix $t = 0.4$

[3] https://huggingface.co/VillanovaAI/Villanova-2B-Base-2603.

to assign comparable weight to both checkpoints and show in Appendix B.1 that results are relatively insensitive to t.

Task vector. Task vector methods represent each model as an update from a shared initialization:

$$\boldsymbol{\theta}_{\text{TV}} = \boldsymbol{\theta}_{\text{init}} + \lambda \sum_{i=1}^{N} w_i \boldsymbol{\tau}_i, \tag{2}$$

where $\boldsymbol{\tau}_i$ is the task vector, w_i are normalized weights such that $\sum_{i=1}^{N} w_i = 1$ and λ is a scaling coefficient. In typical fine-tuning settings, $\boldsymbol{\theta}_{\text{init}}$ corresponds to a common pre-trained model. In our setup, however, checkpoints lie along a single training trajectory. We therefore interpret task vectors as *update vectors* between successive checkpoints and set $\boldsymbol{\theta}_{\text{init}} = \boldsymbol{\theta}_1$, the first checkpoint in the merging range. This yields $N - 1$ task vectors, $\boldsymbol{\tau}_i = \boldsymbol{\theta}_i - \boldsymbol{\theta}_1$, each capturing the parameter update accrued during training. We set $\lambda = 1$ and use uniform weights $w_i = \frac{1}{N-1}$, following standard practice in task arithmetic (Ilharco et al., 2022) when no held-out dataset is available. Under this formulation, task arithmetic reduces to standard linear weight averaging over the considered checkpoints. We therefore focus on TIES (Yadav et al., 2023), which combines task vectors by first applying magnitude pruning, then a sign-consensus operation that keeps only parameter updates with the largest direction and finally computes a disjoint mean that averages each parameter over the number of non-zero contributions rather than the total number of task vectors. Although TIES typically relies on strong magnitude pruning to remove noisy updates, Appendix B.2 shows that it degrades performance in our pre-training setting. Consequently, we use a high density (0.8), retaining most update components. Overall, this procedure is intended to emphasize consistent update directions across checkpoints and strengthen the signal of late-stage training updates.

4 Results

4.1 When model merging helps

Table 2 summarizes average benchmark accuracy across different merging methods and training stages of VILLANOVA 2B. All merging methods improve over the base model during the stable learning rate phase, with gains reaching up to +1.7 points. However, when the learning rate is decayed, the improvements shrink substantially to +0.1–0.5, even though the absolute performance remains higher.

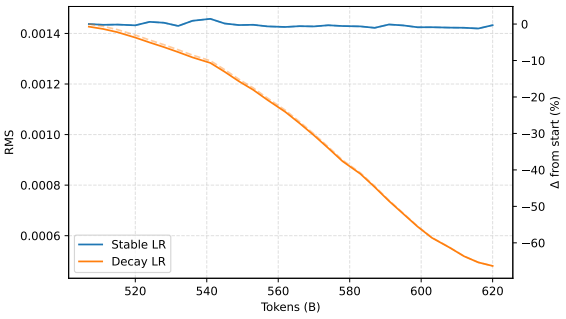

Figure 1: Checkpoint diversity over training. Solid: absolute RMS distance. Dashed: percentage change from the first checkpoint.

To understand why this is the case, we consider the concept of *Effective Noise Scale* (Zhang et al., 2025) which suggests that model merging is particularly beneficial when checkpoints have at least some level of diversity which can naturally be induced by a high learning rate. Thus, we consider the impact of the decaying learning rate on the diversity between checkpoints. Figure 1 shows the average Root Mean Square (RMS) distance between weights of checkpoints separated by 4k steps from 508 to 620 billion tokens as a proxy of diversity. During the stable phase the diversity remains roughly constant, while as the learning rate decays it approaches zero. We empirically validate our conjecture by plotting the accuracy over steps for both the merged and base decayed model with $N = 10$ and the correlation between RMS and accuracy improvement over the same range as long as the steps from 1088 to 1269 billion tokens. Figure 2 clearly shows how the improvement that merging provides gradually shrinks and Figure 3 shows a significant positive correlation between diversity and accuracy improvement. Appendix C.2 presents results with other training ranges, diversity measured in the activations, logits and probability space and additional details.

4.2 Improving performance in low learning rate regimes

We now focus mainly on checkpoints obtained during the learning rate decay phase of VILLANOVA 2B, where merging gains are consistently smaller but model performance remains higher, to investigate whether additional variability can be introduced to further improve results. Sections 4.2.1 and 4.2.2 study alternative merging strategies

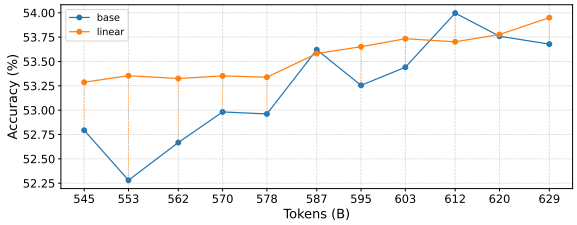

Figure 2: Average accuracy over tokens during learning rate decay. The gap between the merged and base model progressively shrinks as the learning rate decays.

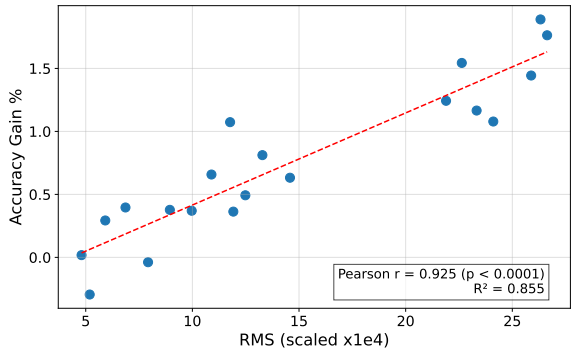

Figure 3: Correlation between checkpoint diversity and accuracy gain over the base model.

that incur no additional training cost, while Sections 4.2.3, 4.2.4 and 4.2.5 consider approaches that require additional computational resources.

4.2.1 Does synthetic variability help?

If our conjecture that reduced checkpoint variability underlies the diminishing gains from model merging is correct, a natural question arises: *can variability be reintroduced in low learning rate regimes to recover some of these gains?* To this end, we explore simple strategies aimed at artificially increasing diversity among checkpoints prior to merging.

We first introduce checkpoint-dropout (*Dropout*), which constructs M merged models by randomly selecting a fraction $1-p$ of the available checkpoints for each merge. This procedure induces variability by exposing each merged model to a slightly different subset of the training trajectory.

A second approach injects Gaussian noise (*Noise*) into each checkpoint:

$$\tilde{\mathbf{w}}_k = \mathbf{w}_k + \boldsymbol{\epsilon}_k, \qquad \boldsymbol{\epsilon}_k \sim \mathcal{N}\left(\mathbf{0}, \sigma_k^2 \mathbf{I}\right), \\ \sigma_k = \alpha \frac{\|\mathbf{w}_k\|_2}{\sqrt{|\mathbf{w}_k|}}. \tag{3}$$

where $\mathbf{w}_k$ denotes the parameters of the checkpoint at step k, and α is a small constant controlling the noise magnitude. This normalization ensures that the injected noise is scale-invariant across checkpoints and can be interpreted as creating synthetic perturbations of the original training trajectory.

For both strategies, we obtain $M = 3$ intermediate merged models differing by seed, which are subsequently merged into a single final model. We fix $p = 0.2$ for *Dropout* and $\alpha = 10^{-3}$ for *Noise*, introducing mild variability intended to increase diversity without significantly perturbing the underlying training trajectory.

Lastly, even though as observed in Table 2 EMA, Linear and WMA have a similar performance we attempt to merge these merged models (*Meta*) as well to see if it is beneficial.

As shown in Table 3, these techniques yield small but consistent improvements in low-variability regimes. In contrast, when the learning rate is stable and checkpoint diversity is already naturally sufficient, these methods provide no benefit, supporting our hypothesis that checkpoint variability is a key driver of successful merging.

4.2.2 Do other merging methods help?

We next examine whether alternative merging strategies mitigate the reduced effectiveness of model averaging during the learning rate decay phase. We consider task vector-based merging (TIES) and non-linear interpolation (SLERP). TIES aim to isolate the directions of change induced by training and selectively combine them, potentially filtering redundant or conflicting updates when checkpoints become highly correlated. In contrast, SLERP interpolates along a geodesic on the hypersphere, preserving parameter norms and avoiding purely affine combinations.

Table 3 shows that TIES consistently outperforms linear merging during decay. While the gains remain modest, they are comparable to those obtained via synthetic variability strategies, suggesting that task vector merging can partially compensate for reduced checkpoint diversity in low learning-rate regimes. We conjecture that TIES emphasizes informative parameter directions while discarding low-variance components that dominate late-stage training.

SLERP exhibits similar behavior, although less consistently than TIES. Overall, more structured merging strategies provide small but repeatable gains when naive averaging begins to saturate.

Tokens(B)	Base	Linear	Dropout	Noise	Meta	SLERP	TIES
629	53.7†	53.8(+0.1)	54.0(+0.3)	54.0(+0.3)	54.1(+0.4)	54.1(+0.4)	54.1(+0.4)
	53.9‡	53.7(-0.2)	53.8(-0.1)	53.8(-0.1)	53.9(+0.0)	53.9(+0.0)	53.7(-0.2)

Table 1: † Last checkpoint before LR decay. ‡ Linearly merged model followed by decay. Both models are decayed using the same schedule and data.

LR stage	Tokens (B)	Base	Linear	EMA	WMA
Stable	629	52.0	53.7(+1.7)	53.7(+1.7)	53.7(+1.7)
Decay		53.7	53.8(+0.1)	53.9(+0.2)	53.9 (+0.2)
Stable	1342	53.0	54.7(+1.7)	54.7(+1.7)	54.8(+1.8)
Decay		54.9	55.3(+0.4)	55.4(+0.5)	55.3(+0.4)
Stable	2306	54.8	56.4(+1.6)	56.4(+1.6)	56.4(+1.6)
Decay		57.1	57.3(+0.2)	57.2(+0.1)	57.4(+0.3)

Table 2: Average accuracy for different model merging methods and training stages with different learning schedules. The improvement in the decay is much smaller than in the stable phase.

4.2.3 Do different data distributions help?

We next investigate whether increasing diversity through different data distributions can improve the effectiveness of model merging during the learning rate decay phase. To this end, we consider three training runs that share the same architecture and optimization setup but differ in the underlying data distribution. For each run, we report the performance of the final checkpoint, the merged model obtained by averaging checkpoints along the same training trajectory, and a merged model obtained by combining checkpoints across runs trained on different data distributions. Refer to Appendix A.2 for additional information about the data distributions.

Table 4 shows that merging checkpoints across training runs with different data distributions leads to consistent, albeit modest, improvements over both the base model and within-run merging. In contrast, merging checkpoints along a single trajectory yields little to no gain in this low-variability regime. While observed improvements remain limited in magnitude, these results indicate that data-level diversity can partially compensate for the lack of variability induced by learning rate decay.

4.2.4 Do different decay schedulers help?

We consider three annealed models trained with different cosine decay schedules, varying the learning rate and the annealing interval between 503 and 629 billion tokens. We iteratively merge these models and compare the resulting checkpoints against both the corresponding base models and the merged models obtained from checkpoints along each individual training trajectory. As shown in Table 5, despite differences in the annealing schedules, the resulting gains remain modest. This suggests that, in low learning rate regimes, diversity induced by moderate scheduler variations is insufficient to substantially improve merging effectiveness. Moreover, Appendix C.1 shows that merging models trained under stable and decaying learning rate schedules can even degrade performance.

4.2.5 Do merged models decay better?

Merging provides clear gains during the stable learning-rate phase, but we ask whether these carry over to the decay phase. At 503B tokens, we compare (i) the last checkpoint before decay and (ii) a linearly merged model at the same point, both decayed with identical data and schedules. Table 1 shows that initializing from a merged model does not consistently improve performance and often slightly underperforms the last checkpoint. These results suggest that merging benefits are largely confined to the stable phase, and decay is sensitive to the training trajectory which suggests that applying merging prior to decay is a reliable substitute for continued training from a single checkpoint.

4.3 Extending results to different models

We extend our study to larger and different models: PYTHIA, SMOLLM3, and OLMO 3 7B. These models introduce additional practical constraints that make controlled merging experiments harder to conduct, *highlighting the relative uniqueness and experimental flexibility of* VILLANOVA 2B.

All PYTHIA and OLMO 3 7B models are trained with a short warmup followed by a decay stage,

LR stage	Tokens (B)	Base	Linear	Dropout	Noise	Meta	SLERP	TIES
Stable	629	52.0	53.7(+1.7)	53.6(+1.6)	53.6(+1.6)	53.6(+1.6)	53.3(+1.3)	53.3(+1.3)
Decay		53.7	53.8(+0.1)	54.0(+0.3)	54.0(+0.3)	54.1(+0.4)	54.1(+0.4)	54.1(+0.4)
Stable	1342	53.0	54.7(+1.7)	54.7(+1.7)	54.7(+1.7)	54.7(+1.7)	54.6(+1.6)	54.4(+1.4)
Decay		54.9	55.3(+0.4)	55.6(+0.7)	55.3(+0.4)	55.6(+0.7)	55.2(-0.1)	55.7(+0.8)
Stable	2306	54.8	56.4(+1.6)	56.4(+1.6)	56.4(+1.6)	56.4(+1.6)	56.3(+1.5)	56.3(+1.5)
Decay		57.1	57.2(+0.1)	57.6(+0.5)	57.6(+0.5)	57.6(+0.5)	57.6(+0.5)	57.7(+0.6)

Table 3: Injecting variability in the checkpoints provides small but consistent improvements. TIES and SLERP provide comparable improvements to the synthetic strategies.

Distribution	Base	Linear	Multi-Data ($\leq i$)
v1	56.7	56.8	56.7
v2	57.1	57.1	57.4(+0.7)
v3	57.1	57.1	57.4(+0.7)

Table 4: Average accuracy for different data distributions with decay between 1207-1342 billion tokens: last checkpoint, merge across a training trajectory and between runs up the ith row.

Tokens (B)	Base	Linear	Multi-Dcos ($\leq i$)
503–629 (3×10^{-5})	53.7	53.8	53.7
566–629 (1×10^{-5})	53.6	53.8	53.9(+0.2)
600–629 (3×10^{-5})	53.4	53.5	53.9(+0.2)

Table 5: Average accuracy for different decay schemes (range and `final lr`): last checkpoint, merge across a training trajectory and between runs up the ith row.

preventing a direct comparison of model merging between stable and decay phases. We therefore approximate the stable phase using the early portion of decay where the learning rate is still high. Following Li et al. (2025), we avoid using the very first checkpoints, as their differences are too large for effective merging. Regarding data, PYTHIA was trained on the Pile (Gao et al., 2020) and OLMO 3 7B on an English-only filtered Common Crawl (Foundation, 2024); consequently, non-English benchmarks are excluded for their evaluation. While all other models release checkpoints every 4B tokens, SMOLLM3 does so every 94.4B tokens ($\approx \times 24$ the spacing of the other models). As a result, using $N = 15$ results in a huge span of 1416B tokens. For the decay phase, we use only 13 checkpoints, as no additional ones are available. We report results for $N = 5$, 10, and 15, noting that large N early in training (PYTHIA) or between widely spaced models (SMOLLM3) is often detrimental (Li et al., 2025). Table 6 shows how not all models support the same merging and evaluation experiments: some are not multilingual, some

Multilingual		✓		✓
WSD		✓		✓
Closeness	✓		✓	✓

Table 6: Comparison of checkpoint characteristics for model families; : PYTHIA, 🤗: SMOLLM3, : OLMO 3 7B; : VILLANOVA 2B.

Model	Size(B)	Tokens(B)	Base	N=5	N=10	N=15
	2.8	125	63.0	+0.7	+0.6	+0.2
		300	65.0	+0.2	+0.1	+0.2
	6.9	125	64.9	+0.5	+0.2	-0.1
		300	67.1	+0.3	+0.5	+0.4
	12	125	65.8	+0.9	+0.6	+0.2
		300	68.3	+0.0	-0.1	-0.2
🤗	3.0	8116	58.2	+1.3	+1.0	+0.1
		11136	60.2	-0.1	-0.9	-1.4
	7.0	1200	71.2	+1.1	+1.0	+1.4
		5927	73.9	+0.0	+0.0	+0.0

Table 7: Average accuracy of different model families across training stages with linear merging. Merged results are reported as deltas relative to the base model. : PYTHIA, 🤗: SMOLLM3, : OLMO 3 7B.

lack closely spaced checkpoints and some don't have a stable LR phase making experiments in Sections 4.2.4 and C.1 harder. Appendix D describes additional information about these models.

Table 7 presents merging results. As expected, stable-phase merges generally yield larger improvements than decay-phase merges. For PYTHIA and SMOLLM3, smaller N values provide better results, whereas OLMO 3 7B performs consistently since stable merges occur between closely spaced checkpoints after extensive training.

5 Conclusion

Our study demonstrates that model merging is most effective during pre-training when learning rates

are stable and checkpoint diversity is high. Gains from simple averaging methods sharply decrease once learning rate decay begins, reflecting the reduced parameter-space variation at late stages of training. Introducing synthetic variability or using task-vector methods can partially recover these gains, but improvements remain modest, suggesting intrinsic limitations in low-variability regimes. Cross-run merging and data-level diversity offer further but limited benefits, and initializing decay from a merged checkpoint does not consistently improve final performance. Thus, by systematically characterizing the interactions between learning rate schedules, checkpoint diversity, and merging strategies, this work clarifies when and how model merging should be applied to maximize pretraining efficiency and model quality.

Limitations

The main limitation of our work is that it focuses primarily on a single model. While we did conduct experiments on larger and different models as shown in Section 4.3, this remain somewhat limited and a more thorough exploration is probably required. Future research should also focus on strategies to increase gains of model merging during training stages where checkpoints variability is very small.

Acknowledgments

This work is part of the Villanova project, financed by IPICEI-CIS, Prog. n. SA. 102519 – CUP B29J24000850005.

References

Loubna Ben Allal, Anton Lozhkov, Elie Bakouch, Gabriel Martín Blázquez, Guilherme Penedo, Lewis Tunstall, Andrés Marafioti, Hynek Kydlíček, Agustín Piqueres Lajarín, Vaibhav Srivastav, Joshua Lochner, Caleb Fahlgren, Xuan-Son Nguyen, Clémentine Fourrier, Ben Burtenshaw, Hugo Larcher, Haojun Zhao, Cyril Zakka, Mathieu Morlon, and 3 others. 2025. Smollm2: When smol goes big – data-centric training of a small language model. *Preprint*, arXiv:2502.02737.

Stella Biderman, Hailey Schoelkopf, Quentin Gregory Anthony, Herbie Bradley, Kyle O'Brien, Eric Hallahan, Mohammad Aflah Khan, Shivanshu Purohit, USVSN Sai Prashanth, Edward Raff, and 1 others. 2023. Pythia: A suite for analyzing large language models across training and scaling. In *International Conference on Machine Learning*, pages 2397–2430. PMLR.

Stella Biderman, Hailey Schoelkopf, Lintang Sutawika, Leo Gao, Jonathan Tow, Baber Abbasi, Alham Fikri Aji, Pawan Sasanka Ammanamanchi, Sidney Black, Jordan Clive, and 1 others. 2024. Lessons from the trenches on reproducible evaluation of language models. *arXiv preprint arXiv:2405.14782*.

Yonatan Bisk, Rowan Zellers, Jianfeng Gao, Yejin Choi, and 1 others. 2020. Piqa: Reasoning about physical commonsense in natural language. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 7432–7439.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*.

Alexis Conneau, Ruty Rinott, Guillaume Lample, Adina Williams, Samuel Bowman, Holger Schwenk, and Veselin Stoyanov. 2018. Xnli: Evaluating cross-lingual sentence representations. In *Proceedings of the 2018 conference on empirical methods in natural language processing*, pages 2475–2485.

Common Crawl Foundation. 2024. Common crawl dataset. https://commoncrawl.org/. Accessed: December 31, 2024.

Leo Gao, Stella Biderman, Sid Black, Laurence Golding, Travis Hoppe, Charles Foster, Jason Phang, Horace He, Anish Thite, Noa Nabeshima, and 1 others. 2020. The pile: An 800gb dataset of diverse text for language modeling. *arXiv preprint arXiv:2101.00027*.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, and 1 others. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Oğuz Kağan Hitit, Leander Girrbach, and Zeynep Akata. 2025. A systematic study of model merging techniques in large language models. *arXiv preprint arXiv:2511.21437*.

Gabriel Ilharco, Marco Tulio Ribeiro, Mitchell Wortsman, Suchin Gururangan, Ludwig Schmidt, Hannaneh Hajishirzi, and Ali Farhadi. 2022. Editing models with task arithmetic. *arXiv preprint arXiv:2212.04089*.

Pavel Izmailov, Dmitrii Podoprikhin, Timur Garipov, Dmitry Vetrov, and Andrew Gordon Wilson. 2018. Averaging weights leads to wider optima and better generalization. *arXiv preprint arXiv:1803.05407*.

Dong-Hwan Jang, Sangdoo Yun, and Dongyoon Han. 2024. Model stock: All we need is just a few fine-tuned models. In *European Conference on Computer Vision*, pages 207–223. Springer.

Jean Kaddour. 2022. Stop wasting my time! saving days of imagenet and bert training with latest weight averaging. *arXiv preprint arXiv:2209.14981*.

Nikhil Kandpal, Brian Lester, Colin Raffel, Sebastian Majstorovic, Stella Biderman, Baber Abbasi, Luca Soldaini, Enrico Shippole, A. Feder Cooper, Aviya Skowron, Shayne Longpre, Lintang Sutawika, Alon Albalak, Zhenlin Xu, Guilherme Penedo, Loubna Ben, Elie Bakouch, John David, Honglu Fan, and 8 others. 2025. The Common Pile v0.1: An 8TB Dataset of Public Domain and Openly Licensed Text. *arXiv preprint*.

Hynek Kydlíček, Guilherme Penedo, and Leandro von Werra. 2025. Finepdfs. https://huggingface.co/datasets/HuggingFaceFW/finepdfs.

Yunshui Li, Yiyuan Ma, Shen Yan, Chaoyi Zhang, Jing Liu, Jianqiao Lu, Ziwen Xu, Mengzhao Chen, Minrui Wang, Shiyi Zhan, and 1 others. 2025. Model merging in pre-training of large language models. *arXiv preprint arXiv:2505.12082*.

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

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

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

Shayne Longpre, Le Hou, Tu Vu, Albert Webson, Hyung Won Chung, Yi Tay, Denny Zhou, Quoc V Le, Barret Zoph, Jason Wei, and 1 others. 2023. The flan collection: Designing data and methods for effective instruction tuning. In *International Conference on Machine Learning*, pages 22631–22648. PMLR.

Anton Lozhkov, Edward Beeching, Aymeric Roucher, Nouamane Tazi, Aksel Joonas Reedi, Guilherme Penedo, Hynek Kydlicek, Clémentine Fourrier, Nathan Habib, Kashif Rasul, Quentin Gallouédec, Hugo Larcher, Mathieu Morlon, Joshua, Vaibhav Srivastav, Xuan-Son Nguyen, Colin Raffel, Lewis Tunstall, Loubna Ben Allal, and 2 others. 2025. Smollm3: Smol, multilingual, long-context reasoner. https://huggingface.co/blog/smollm3. Hugging Face Blog.

Anton Lozhkov, Loubna Ben Allal, Leandro von Werra, and Thomas Wolf. 2024a. Fineweb-edu: the finest collection of educational content.

Anton Lozhkov, Raymond Li, Loubna Ben Allal, Federico Cassano, Joel Lamy-Poirier, Nouamane Tazi, Ao Tang, Dmytro Pykhtar, Jiawei Liu, Yuxiang Wei, and 1 others. 2024b. Starcoder 2 and the stack v2: The next generation. *arXiv preprint arXiv:2402.19173*.

Team Olmo, Allyson Ettinger, Amanda Bertsch, Bailey Kuehl, David Graham, David Heineman, Dirk Groeneveld, Faeze Brahman, Finbarr Timbers, Hamish Ivison, and 1 others. 2025. Olmo 3. *arXiv preprint arXiv:2512.13961*.

Team OLMo, Pete Walsh, Luca Soldaini, Dirk Groeneveld, Kyle Lo, Shane Arora, Akshita Bhagia, Yuling Gu, Shengyi Huang, Matt Jordan, and 1 others. 2024. 2 olmo 2 furious. *arXiv preprint arXiv:2501.00656*.

Guilherme Penedo. 2025. Finewiki. https://huggingface.co/datasets/HuggingFaceFW/finewiki. Hugging Face Datasets. Source: Wikimedia Enterprise Snapshot API. Text licensed under CC BY-SA 4.0. Accessed 2025-10-20.

Guilherme Penedo, Hynek Kydlíček, Loubna Ben allal, Anton Lozhkov, Margaret Mitchell, Colin Raffel, Leandro Von Werra, and Thomas Wolf. 2024. The fineweb datasets: Decanting the web for the finest text data at scale. In *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

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

Edoardo Maria Ponti, Goran Glavaš, Olga Majewska, Qianchu Liu, Ivan Vulić, and Anna Korhonen. 2020. Xcopa: A multilingual dataset for causal commonsense reasoning. *arXiv preprint arXiv:2005.00333*.

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

Sunny Sanyal, Atula Neerkaje, Jean Kaddour, Abhishek Kumar, and Sujay Sanghavi. 2023. Early weight averaging meets high learning rates for llm pre-training. *arXiv preprint arXiv:2306.03241*.

Ken Shoemake. 1985. Animating rotation with quaternion curves. In *Proceedings of the 12th annual conference on Computer graphics and interactive techniques*, pages 245–254.

Jianlin Su, Yu Lu, Shengfeng Pan, Ahmed Murtadha, Bo Wen, and Yunfeng Liu. 2023. Roformer: Enhanced transformer with rotary position embedding. *Preprint*, arXiv:2104.09864.

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

Johannes Welbl, Nelson F Liu, and Matt Gardner. 2017. Crowdsourcing multiple choice science questions. *arXiv preprint arXiv:1707.06209*.

Mitchell Wortsman, Gabriel Ilharco, Samir Ya Gadre, Rebecca Roelofs, Raphael Gontijo-Lopes, Ari S Morcos, Hongseok Namkoong, Ali Farhadi, Yair Carmon, Simon Kornblith, and 1 others. 2022. Model soups: averaging weights of multiple fine-tuned models improves accuracy without increasing inference time. In *International conference on machine learning*, pages 23965–23998. PMLR.

Prateek Yadav, Derek Tam, Leshem Choshen, Colin A Raffel, and Mohit Bansal. 2023. Ties-merging: Resolving interference when merging models. *Advances in Neural Information Processing Systems*, 36:7093–7115.

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

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

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

Chenxiang Zhang, Alexander Theus, Damien Teney, Antonio Orvieto, Jun Pang, and Sjouke Mauw. 2025. How does the optimizer implicitly bias the model merging loss landscape? *arXiv preprint arXiv:2510.04686*.

A Models and training

A.1 Model architecture

In most experiments, we use two proprietary models, collectively referred to as VILLANOVA 2B. Both are based on the LLaMA architecture (Touvron et al., 2023) and incorporate RMSNorm, SwiGLU activations, and Rotary Positional Embeddings (RoPE) (Su et al., 2023). Despite these shared foundations, the models differ substantially in depth, width, and attention design.

The first model, VILLANOVADEEP, employs standard Multi-Head Attention (MHA) with untied input and output embeddings. The second, VILLANOVAWIDE, follows a shallower yet wider configuration, uses Grouped-Query Attention (GQA) to improve inference efficiency, and shares weights between the input and output embeddings. Table 8 summarizes the architectural details of both models.

Training uses the Distributed Fused Adam optimizer with a global batch size of 512. At a sequence length of 4096 tokens, each optimization step processes approximately 2.1 million tokens. Checkpoints are saved every 2k steps, corresponding to roughly 4B processed tokens.

Hyperparameter	DEEP	WIDE
Layers (N_L)	24	18
Hidden Size (d_{model})	2048	2560
FFN Hidden Size (d_{ffn})	5440	10240
Attention Heads (H)	16	20
KV Heads (H_{kv})	16 (MHA)	4 (GQA)
Head Dimension	128	128
Tied Embeddings	FALSE	TRUE

Table 8: Architectural comparison of the two models used for pretraining experiments.

A.2 Data Distribution

Table 9 reports the data mixtures for different models and pre-training phases. We distinguish between a stable pre-training phase and a subsequent annealing phase, where the data composition is progressively shifted toward higher-quality and more specialized sources. Overall, the mixtures combine large-scale web corpora, curated PDF documents, code datasets, and math and question-answering (QA) resources. During annealing, the proportion of generic web data is reduced in favor of higher-quality web subsets, curated PDFs, code (including high-quality code), and math/QA-oriented corpora.

Web. For English web data, we rely on FineWeb-edu (Lozhkov et al., 2024a) only, with the exception of VILLANOVADEEP where we used the 350B token sample of FineWeb (Penedo et al., 2024) too. Multilingual web data is drawn from FineWeb-2 (Penedo et al., 2025), with a predominance of Spanish, French, German, and Italian languages. Higher-quality English web data (Web–English HQ) is introduced in VILLANOVAWIDE and further emphasized during annealing. This portion includes curated sources such as FineWiki (Penedo, 2025) and filtered arXiv and PubMed articles from (Kandpal et al., 2025).

Pdfs. PDF-based corpora are drawn from FinePDFs (Kydlíček et al., 2025). These are introduced in VILLANOVAWIDE during stable pre-training (7% English, 4% multilingual) and are further increased during annealing (up to 9–10%).

This shift reflects a deliberate emphasis on structurally richer and more curated long-form documents.

Code. Code data includes Stack-edu (Allal et al., 2025) and CraneCode (codeHQ) (Olmo et al., 2025). Code is incorporated only in VILLANOVAWIDE, starting from 5% in stable pre-training. During annealing, its proportion initially increases (7% in v1 and v2). In v3, high-quality code (5%), replaced most of the code data.

Math & QA. Math and instruction-styled datasets, such as Flan (Longpre et al., 2023) from Dolmino (OLMo et al., 2024), MegaMatt, Wiki To RCQA from (Olmo et al., 2025), Kaggle notebook issues (Lozhkov et al., 2024b), and FineMath (Allal et al., 2025) are included in v2 and v3 annealings.

	Stable Pre-training		Annealing		
Corpus	DEEP	WIDE	v1	v2	v3
Web - English	0.56	0.38	0.28	0.24	0.24
Web - Multilingual	0.44	0.42	0.33	0.29	0.29
Web - English HQ	–	0.04	0.10	0.10	0.10
Pdfs - English	–	0.07	0.09	0.08	0.08
Pdfs - Multilingual	–	0.04	0.10	0.09	0.09
Code	–	0.05	0.07	0.07	0.02
Code HQ	–	–	–	–	0.05
Math & QA	–	–	0.03	0.13	0.13

Table 9: Data distributions used for pre-training. The distributions labelled as *v* are used to train different versions of VILLANOVAWIDE between 1207B and 1342B tokens for the experiments reported in Table 4.

A.3 Learning Rate Schedules

All runs use a peak learning rate of 3×10^{-4} during the stable phase. Differences arise only in the decay interval and final learning rate.

Table 10 summarizes the schedules used across experiments.

B Model Merging Hyperparameters

B.1 SLERP Hyperparameters

We empirically evaluate different interpolation coefficients $t \in \{0.2, 0.4, 0.6, 0.8\}$ for SLERP. Recall that smaller t values place more weight on the previous checkpoint, while larger values favor the later one.

Figure 4 shows that performance is relatively insensitive to the precise choice of t. In the stable phase, smaller values ($t = 0.2$) tend to perform slightly better at larger token counts, whereas

Experiments	Decay (B tokens)	Final LR
Tab. 2, Tab. 3, Tab. 1	503–629	3×10^{-5}
	1073–1342	3×10^{-6}
	2097–2306	3×10^{-6}
Tab. 4	1207–1342	3×10^{-6}
Tab. 5	503–629	3×10^{-5}
	566–629	1×10^{-5}
	600–629	3×10^{-5}

Table 10: Summary of learning rate decay schedules.

performance degrades gradually as t increases toward 0.8. In the decay phase, differences are modest overall: $t \in \{0.4, 0.6\}$ consistently match or slightly outperform linear merging, while extreme values do not provide additional gains.

Given the limited sensitivity and the desire to assign comparable weight to both checkpoints, we fix $t = 0.4$ in our main experiments.

B.2 TIES Hyperparameters

In this section, we empirically justify the hyperparameter choices adopted for TIES.

We present additional results using TIES merging to motivate our choice of a high density (i.e., weak pruning). In Section 3.3, we noted that while the original TIES formulation (Yadav et al., 2023) typically uses a low density (20%), such aggressive pruning is detrimental in the pre-training setting. Figure 5 shows that, for both the stable and decay learning-rate regimes, performance remains relatively stable when using high densities, whereas low densities lead to substantial degradation and underperform simpler approaches such as `linear` merging.

We further analyze the sensitivity of TIES to checkpoint spacing. Figure 6 shows that TIES yields competitive improvements only when merging checkpoints that are sufficiently close in training. Increasing the spacing 4B to 8B tokens results in a significant drop in performance.

C Additional results

This section is meant to present additional miscellaneous experiments.

C.1 Different schedulers contribution

Section 4.2.4 investigates whether different learning rate decay schedules introduce sufficient variability to improve merged models. Here, we extend

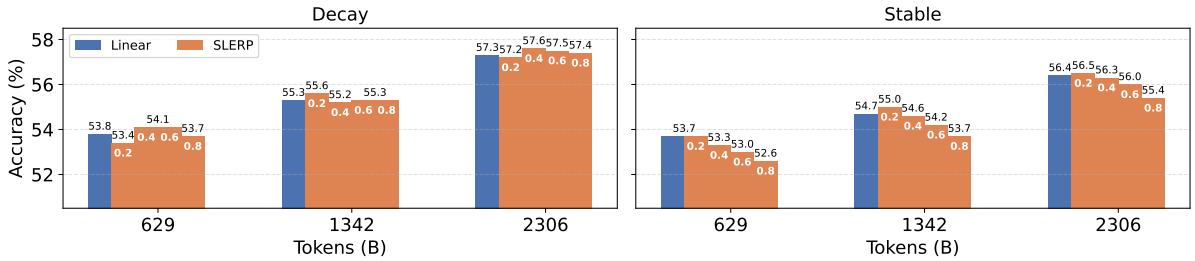

Figure 4: Effect of SLERP t on performance across pre-training stages. Lower values seem to yield slightly better results during the Stable stage, although in general results are relatively unaffected by it.

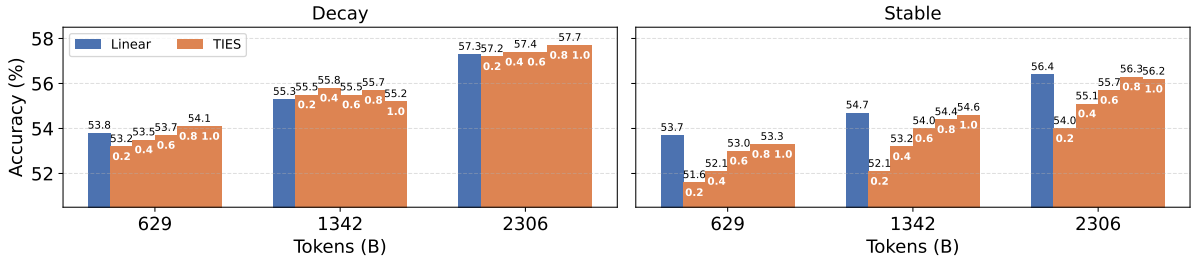

Figure 5: Effect of TIES density on performance across pre-training stages. Lower density ratios reduce performance, while TIES outperforms Linear in the Decay stage only at higher densities.

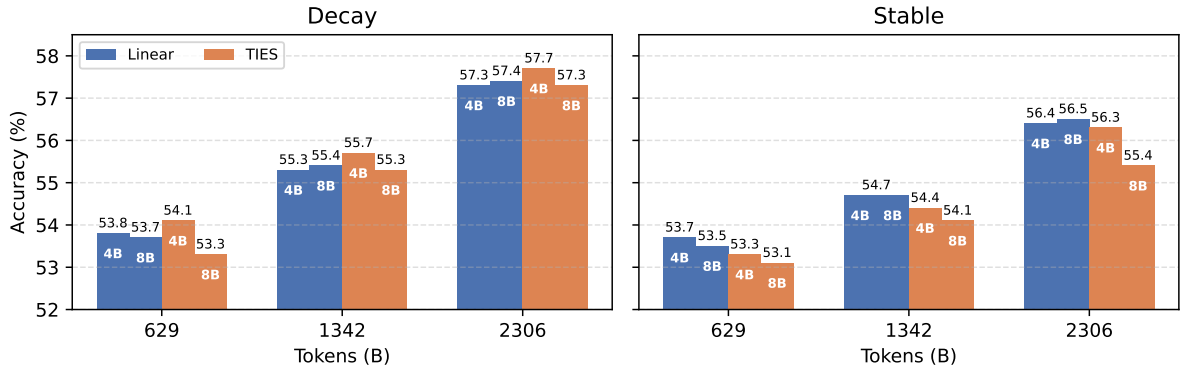

Figure 6: Effect of checkpoint spacing on TIES merging. TIES performance degrades with larger sampling distance, while Linear is mostly unaffected.

Tokens (B)	Base	Linear	Multi-Dcos ($\leq i$)
629	52.0	53.7	52.0
503–629	53.7	53.8	53.4(+1.4)
566–629	53.6	53.8	53.6(+1.6)
1207	52.6	54.8	52.6
1073-1207	53.5	55.0	54.3(+1.7)

Table 11: Average accuracy for stable and decayed models: last checkpoint, merge across a training trajectory and between runs up the ith row.

those results by additionally examining whether merging models trained with a decay schedule together with their counterparts trained under a constant learning rate provides any benefit. Table 11 reports results at two stages of training. As shown, merging a decayed model with its stable counterpart can improve the performance of the stable model. However, this approach is detrimental compared to the decayed model itself, and in particular, it underperforms the merged model obtained by combining only checkpoints collected during the decay or the stable phase.

C.2 Checkpoints diversity during training

To quantify checkpoint diversity during training, we compute the RMS distance between corresponding tensors of two checkpoints at 4k steps (≈8B tokens) distance. Given two tensors t_1 and t_2 with identical shape, we define the per-tensor RMS difference as:

$$\mathrm{RMS}(t_1, t_2) = \sqrt{\frac{1}{|t_1|}\sum_i (t_{1,i} - t_{2,i})^2}, \qquad (4)$$

where $|t_1|$ denotes the number of elements in the tensor. We then aggregate by averaging across all layers. Let $\{\mathrm{RMS}_\ell\}$ denote the RMS values for each layer ℓ; the overall divergence between two checkpoints is:

$$\mathrm{RMS}_{\text{model}} = \frac{1}{L}\sum_{\ell=1}^{L} \mathrm{RMS}_\ell, \qquad (5)$$

where L is the number of valid layers.

Figures 7 and 8 report the evolution of the checkpoint diversity across training beyond the specific windows highlighted in the main text. During phases with a constant (stable) learning rate, the divergence remains approximately constant, indicating that successive checkpoints maintain a similar degree of separation in parameter space. In contrast, during the decay phase, the divergence

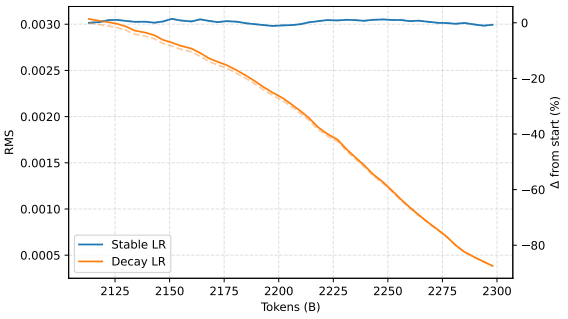

Figure 7: Checkpoint diversity over training. Solid: absolute RMS diversity. Dashed: percentage change from the first checkpoint.

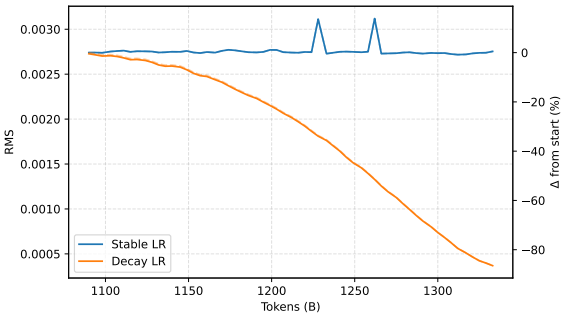

Figure 8: Checkpoint diversity over training. Solid: absolute RMS diversity. Dashed: percentage change from the first checkpoint.

progressively decreases as training advances. This reflects the shrinking update magnitude induced by the decaying learning rate: parameter updates become smaller, checkpoints grow increasingly similar, and the RMS distance correspondingly contracts. We note how the small spikes in Figure 8 are artifacts caused by a couple of missing checkpoints due to a saving error, specifically checkpoints at 590k (≈1237B tokens) and 606k (≈1270B tokens) steps. Thus, when computing RMS for steps 586k (≈1228B tokens) and 602k (≈1262B tokens) we use later checkpoints at 592k and 608k step respectively, increasing the spacing from 8B to 12B tokens. This larger token gap naturally produces higher divergence, resulting in the observed spikes. These artifacts do not affect the overall conclusion: the other diversity plots consistently show stable differences under a constant learning rate and decreasing differences during the decay phase.

While previous experiments suggest that RMS is already an effective metric for measuring checkpoint diversity in model merging, we also evaluate diversity using cosine distance in both the activation and logit spaces, and KL divergence in

the probability space, following the same procedure adopted for RMS. Activations, logits, and probabilities are computed on a randomly sampled batch from HellaSwag, using the final token of the "context + correct answer" sequence. The results, shown in Figure 9, indicate that the observed diversity patterns remain consistent across different representation spaces and metrics.

D Additional details of experiments on additional models

This appendix provides additional relevant details on the models used to extend our experiments beyond VILLANOVA 2B, including PYTHIA, SMOLLM3, and OLMO 3 7B. These models differ in training setups and checkpoint availability, which influences how model merging can be applied.

D.1 Pythia

The PYTHIA family was trained on English-only text with a warmup followed by a decay stage. Because they lack a stable phase, we approximate a pseudo-stable phase using the early decay region where the learning rate is still relatively high. Figure 10 shows the learning rate schedules and the checkpoint ranges selected for merging. Using earlier checkpoints would involve merging models with very little training and thus very different in performance, which can harm the effectiveness of the merge.

D.2 SmolLM3

SMOLLM3 is multilingual and releases checkpoints less frequently than PYTHIA. Specifically, we consider the ones released during the following stages:

- Stage 1 (Stable): steps 0 to 3,450,000 (86 checkpoints)
- Stage 3 (Decay): steps 4,200,000 to 4,720,000 (13 checkpoints)

Although they use a similar batch size to other models of 2,359,296 tokens, they only release checkpoints every 40,000 steps which is approximately equivalent to 94.4B tokens. The large interval between checkpoints limits merging using a high number of models ($N = 15$).

D.3 OLMo 3 7B

OLMO 3 7B is trained in English only, with longer sequences of warmup and decay:

- Checkpoints spaced every 1,000 steps (batch size 4,194,304 tokens), approximately equivalent to the 2k steps of the PYTHIA and VILLANOVA 2B models.
- The early checkpoints considered are already well-trained, allowing pseudo-stable merges even at 1,200B tokens
- Only English benchmarks are used in evaluation

Figure 11 shows the learning rate schedule and merging ranges for OLMO 3 7B.

D.4 Checkpoint characteristics

Table 6 summarizes key differences across the models:

- **Multilingual**: whether the model is trained on multiple languages.
- **WSD (Warmup, Stable, Decay)**: whether checkpoints exist across all learning rate stages, enabling merging experiments.
- **Closeness**: whether checkpoints are sufficiently close in training to allow effective merging.

Not all models support all experiment types, which makes our study of VILLANOVA 2B particularly unique.

E Computational resources

All training runs used 8×8 H100 GPUs. Model merging is performed in RAM, requiring up to 130 GB of memory.

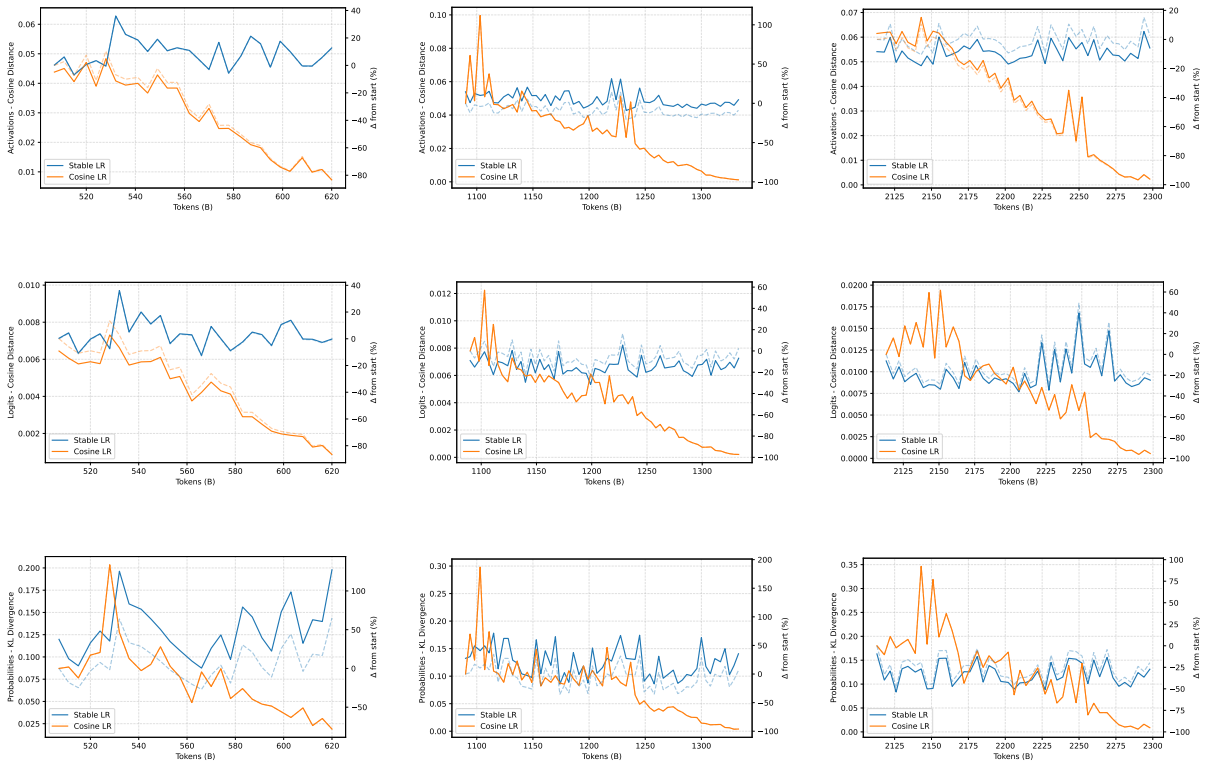

Figure 9: Model diversity over training. Top row: activations. Middle row: logits. Bottom row: probabilities. Solid lines: absolute distance measured with cosine distance/KL divergence. Dashed lines: percentage change from the first checkpoint.

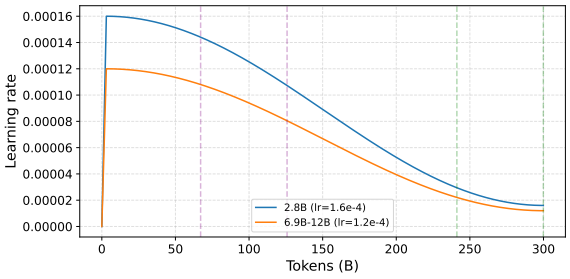

Figure 10: Learning rate schedules used for the Pythia model family. The vertical lines indicate, from left to right, the checkpoint ranges used for merging: pseudo-stable phase and late decay phase.

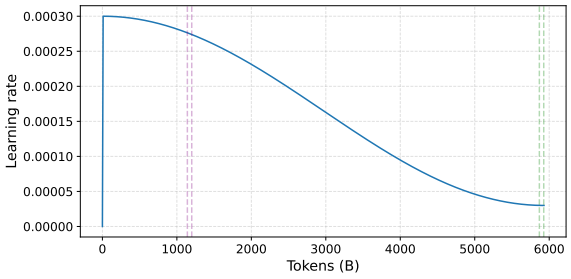

Figure 11: Learning rate schedules used for the OLMo 3 7B model. The vertical lines indicate, from left to right, the checkpoint ranges used for merging: pseudo-stable phase and late decay phase.