MuonAll: Muon Variant for Efficient Finetuning of Large Language Models

Saurabh Page
Pune, India
saurabhpage1@gmail.com

Advait Joshi
Computer Engineering
Pune Institute of Computer Technology
Pune, India
advaitkjoshi@gmail.com

S. S. Sonawane
Computer Engineering
Pune Institute of Computer Technology
Pune, India
sssonawane@pict.edu

***Abstract*—Muon optimizer has demonstrated robust results in pretraining of language models but its performance in finetuning of existing public pretrained models is not yet explored. Currently, Muon is used along with AdamW introducing a scope of improvement for adopting all parameters inside Muon. We introduce MuonAll, which incorporates all the parameters inside Muon by transforming into 2D matrices. We conduct extensive finetuning experiments across publicly available language models with model sizes upto half billion parameters. Muon and MuonAll perform at par with AdamW across major benchmarks, highlighting their effectiveness as alternative optimizers. We open-source the distributed implementations of Muon and MuonAll, available at https://github.com/Saurabh750/optimizer**

***Index Terms*—Optimization algorithms, AdamW, Muon, language models, finetuning, natural language processing**

I. Introduction

The natural language processing (NLP) field has been rapidly advancing with the recent research efforts contributed by [1], [2], [3]. With the fast evolving language model space, the requirement of computational power has been ever increasing and there is a pressing need for finding efficient methods for model training. One such approach for compute efficient training is choosing the right optimizer. Adam ([4]) and AdamW ([5]) have become primary choices for pretraining and finetuning stages of language model training.

To improve training efficiency, different optimization algorithms have been proposed ([6], [7], [8], [9], [10], [11], [12], [13], [14], [15]). Of these, Muon introduced by [7] performs othogonalization on momentum of matrix parameters followed by updation of matrix parameters. It shows promising results compared to AdamW in pretraining of small language models. Muon achieves target loss value in less number of tokens compared to AdamW. [16] shows that by performing some modifications in Muon architecture, it can be scaled for large language model training. The paper briefly compares Muon-AdamW performance on supervised finetuning (SFT) task on public pre-trained models. The paper mentions possibility of inclusion of all parameters into Muon.

In this paper, we provide a detailed analysis of Muon-AdamW performance on SFT task on small language models (SLMs). We explore the direction of incoporating all model parameters inside Muon framework by modifying the Muon architecture.

Computational Linguistics Lab, Department of Computer Engineering, Pune Institute of Computer Technology

II. Background

Muon (MomentUm Orthogonalized by Newton-Schulz) is a matrix orthogonalization based optimizer specially designed for two-dimensional parameters of hidden layers in a neural network [7]. At iteration t, given current weight $\mathbf{W}_{t-1}$, momentum μ, learning rate η_t, and objective $\mathcal{L}_t$, its update rule can be stated as:

$$\mathbf{M}_t \;=\; \mu\,\mathbf{M}_{t-1} \;+\; \nabla\mathcal{L}_t(\mathbf{W}_{t-1}) \tag{1}$$

$$\mathbf{O}_t = \mathrm{Newton{-}Schulz}(\mu\,\mathbf{M}_t + \nabla\mathcal{L}_t\mathbf{W}_{t-1}) \tag{2}$$

$$\mathbf{W}_t \;=\; \mathbf{W}_{t-1} \;-\; \eta_t\,\mathbf{O}_t \tag{3}$$

Here, $\mathbf{M}_t$ is the momentum of the gradient at iteration t, with $\mathbf{M}_0 = \mathbf{0}$. In Equation (2), Newton–Schulz iteration ([17]) is adopted to approximately compute $\left(\mathbf{M}_t\mathbf{M}_t^\top\right)^{-1/2}\mathbf{M}_t$. In equation (2), Nesterov-style momentum is passed to the NS iterations. [7] reports that Nesterov momentum gives better empirical results than standard SGD-momentum.
$\left(\mathbf{M}_t\mathbf{M}_t^\top\right)^{-1/2}\mathbf{M}_t$, the key term involved in Muon's operation whitens the rows of $\mathbf{M}_t$. Whitening operation normalizes the variance along each direction and decorrelates updates. It ensures momentum update doesn't favor redundant directions, which is a common inefficiency in SGD or Adam. If we have access to Singular Value Decomposition (SVD) of $\mathbf{M}_t$ i.e.

$$\mathbf{M}_t \;=\; \mathbf{U}\mathbf{\Sigma}\mathbf{V}^\top$$

whitening looks like:

$$\left(\mathbf{M}_t\mathbf{M}_t^\top\right)^{-1/2}\mathbf{M}_t \;=\; \mathbf{U}\mathbf{V}^\top,$$

This transformation retains directionality in the form of $\mathbf{U}\mathbf{V}^\top$ but removes scale(i.e. singular values which are stored in $\mathbf{\Sigma}$) leading to orthogonal updates. Calculation of SVD increases

Muon's wallclock time. To tackle this, Newton-Schulz iteraton is used. One iteration of Newton-Schulz looks like this:

$$\begin{aligned} G' &:= aG + b(GG^\top)G + c(GG^\top)^2G \\ &= (aI + b(GG^\top) + c(GG^\top)^2)G \\ &= (aI + bUS^2U^\top + cUS^4U^\top)USV^\top \\ &= U(aS + bS^3 + cS^5)V^\top \\ &= U\varphi(S)V^\top \end{aligned}$$

Here, $G = M_t/||M_t||_F$ where $||M_t||_F$ is the Frobenius Norm. Applying N steps of NS iteration gives us the following output $U\varphi^N(S)V^\top$. To make sure the output converges to $UV^\top$, the values of coefficients a, b, c are tuned and set to (3.4445, 4.7750, 2.0315) in the original Muon implementation such that the $\varphi^N(x) \to 1$

Convergence of Muon under different conditions are analyzed and discussed in ([18], [19], [20], [21]). J. Li and M. Hong analyzed Muon as a specialized steepest descent method whose update direction solves a quadratic model under a spectral-norm constraint, and establishes convergence guarantees for both a Nesterov-style and a heavy-ball-style variant of the algorithm [18]. Rigorous convergence proofs for Muon across four realistic variants (with/without Nesterov momentum and with/without decoupled weight decay), show the standard configuration with both momentum and weight decay attains the tightest theoretical bounds and fastest empirical convergence [19]. Convergence guarantees for Muon in nonconvex settings are developed and its rates are compared with Gradient Descent, identifying structural Hessian conditions under which Muon is provably faster [20]. L. Chen, J. Li, and Q. Liu gave a theoretical framing of Muon by placing it inside the Lion-$\mathcal{K}$ optimizer family with $\mathcal{K}$ equal to the nuclear norm, implying that Muon with decoupled weight decay implicitly solves a spectral-norm constrained optimization problem on weight matrices in [21].

An adaptive variant of Muon is proposed in [22] that combines element-wise second-moment scaling with orthogonalized updates to improve stability and efficiency for large-scale training. Empirical results report that AdaMuon maintains Muon's stability and can surpass Adam by over 40% training efficiency at scale across models and learning-rate regimes, indicating practical gains in convergence speed without added complexity. A study of DiLoCo's inner optimizer and communication compressibility [23], shows that replacing AdamW with Muon plus error-feedback enables aggressive compression of the communicated deltas to 2-bit with negligible loss while preserving memory complexity.

Essential AI [24] evaluated Muon's practical pretraining efficiency with a focus on compute–time and data–batch tradeoffs, showing that it expands the Pareto frontier over AdamW by maintaining data efficiency at very large batch sizes while staying computationally lightweight for modern hardware. The study also demonstrates that Muon can be paired with maximal update parameterization (muP) to transfer hyperparameters across widths via a simple telescoping calibration procedure with modest overhead, validated up to 4B-parameter models and across data/architecture ablations. Empirically, Muon achieves target losses faster at a fixed device count and achieves similar losses with fewer tokens at fixed wall-clock, explicitly improving the resources–time frontier relative to AdamW for pretraining transformers up to multi-billion scale. J. Liu has provided a comprehensive study on the scalability of Muon in LLM training [16]. The study demonstrates that Muon can effectively replace AdamW as the standard optimizer for large-scale LLM training. The authors scale up Muon to large language models by doing the following: 1) Adding standard AdamW weight decay mechanism to Muon 2) carefully adjusting per-parameter update scale. The authors opensource Moonlight model(a 3B/16B-parameter MoE model trained on 5.7 trillion tokens) and intermediate training checkpoints. The authors show that Muon performs on par with AdamW in SFT on a public pretrained model. The paper suggests several promising directions for future research that include incorporating all parameters into the Muon framework and developing approaches to better understand and address the pretraining–finetuning mismatch.

III. MuonAll

Currently, Muon works only for 2-dimensional parameters. AdamW optimizer is used for the remaining parameters. To remove the dependency on AdamW and with the aim of incorporating all the parameters in a single optimizer, we introduce MuonAll, modified version of Muon. MuonAll can be used as a single optimizer for all parameters including 1D parameters. MuonAll is defined as follows: MuonAll architecture has two additional steps as compared to Muon. Before calculating momentum, if the parameter is one-dimensional, we convert it to a diagonal matrix with all the elements along the main diagonal. We calculate momentum and pass all the parameters through Newton-Schulz iterations. In the second additional step, the diagonal matrix parameters from step 1 are converted back to one-dimensional parameters.

Algorithm 1 MuonAll

Require: Learning rate η, momentum μ
1: Initialize $B_0 \leftarrow 0$
2: **for** $t = 1, \ldots$ n **do**
3: Compute gradient $G_t \leftarrow \nabla_\theta \mathcal{L}_t(\theta_{t-1})$
4: **if** paramater is one-dimensional **then**
5: Convert parameter into a diagonal-matrix with elements along the main diagonal
6: **end if**
7: $B_t \leftarrow \mu B_{t-1} + G_t$
8: $O_t \leftarrow \text{NewtonSchulz5}(G_t + \mu B_t)$
9: **if** paramater is one-dimensional **then**
10: Convert diagonal matrix back into one-dimensional parameter
11: **end if**
12: Update parameters $\theta_t \leftarrow \theta_{t-1} - \eta O_t$
13: **end for**
14: **return** θ_t

We conduct model finetuning experiments and compare the performance of MuonAll with Muon and AdamW in Section IV.

IV. Experiments

We perform supervised finetuning(SFT) experiments on 3 base models namely Qwen2-0.5B [25], SmolLM2-360M [26] and GPT2 medium [27]. We randomly pick 400k samples from publicly available OpenOrca dataset [28]. To maintain consistency, we make sure the same dataset samples are used across experiments. All the experiments are performed for 2 epochs with a maximum model sequence length of 1024. For each optimizer we use a linear learning-rate schedule that decays from an initial rate to 10^{-7} at the end of epoch 2. The experiments were performed on 2 NVIDIA RTX A6000 GPUs.

Evaluation Benchmarks: Our evaluation uses several benchmarks to assess language model in different ways:

- **Knowledge QA**: MMLU [29], ARC-Easy [30](science focus).
- **Commonsense**: HellaSwag [31], Winogrande [32], PIQA [33].
- **Reading/factuality**: BoolQ [34].
- **Math reasoning**: GSM8K [35].
- **Science composition**: OpenBookQA [36](claiming knowledge composition/RAG)
- **LM quality**: LAMBADA [37] (perplexity/LM behavior rather than task QA)

A. SFT on Qwen2-0.5B

SFT experiments are performed using AdamW, MuonAll and Muon optimizer. For all three experiments, an effective batch size of 48 (batch size: 12, gradient accumulation: 4) is used. Following initial learning rates are used for AdamW and MuonAll: 2 x 10^{-5}, 5 x 10^{-5} respectively. For Muon, two learning rates need to be set: 5 x 10^{-5}(for matrix parameters) and 2 x 10^{-5} (for non-matrix parameters). The training and validation loss for each experiment can be found in table IV. Validation loss curves can be found in figure 2. The finetuned versions are evaluated over the benchmarks. The benchmarking scores can be found in table I.

The performance of Muon and MuonAll is comparable with AdamW. In the benchmarks where AdamW outperforms Muon; MuonAll outperforms Muon as well. Muon versions perform better than AdamW in major benchmarks: MMLU and GSM8K. The figure 1 gives a good visualization of the benchmarking scores.

TABLE I
Benchmark evaluation results of Qwen2-0.5B

Benchmark	AdamW	MuonAll	Muon
MMLU 5-shot	43.64%	43.82%	**44.17%**
GSM8K	32.22%	**35.78%**	35.03%
HellaSwag 10-shot	**50.22%**	49.51%	49.39%
PIQA 0-shot	**69.70%**	69.26%	68.50%
ARC-Easy 25-shot	**49.33%**	47.56%	47.56%
BoolQ 0-shot	**59.97%**	55.11%	56.85%
Winogrande 5-shot	55.80%	57.62%	**57.77%**
OpenBookQA 0-shot	**34.20%**	33.40%	33.00%
LAMBADA (OpenAI variant) 0-shot	**50.15%**	49.76%	47.41%

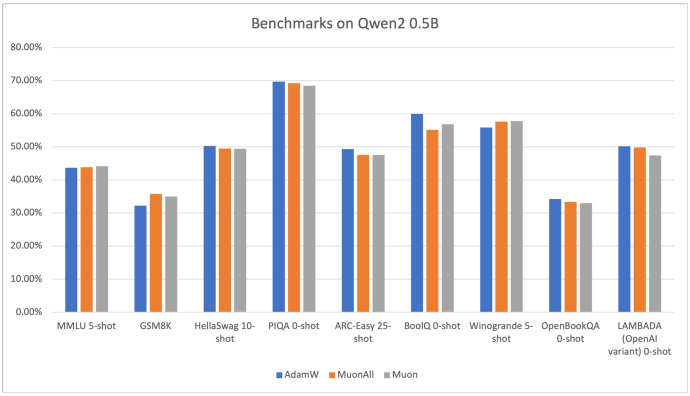

Fig. 1. Benchmarks on Qwen2 0.5B

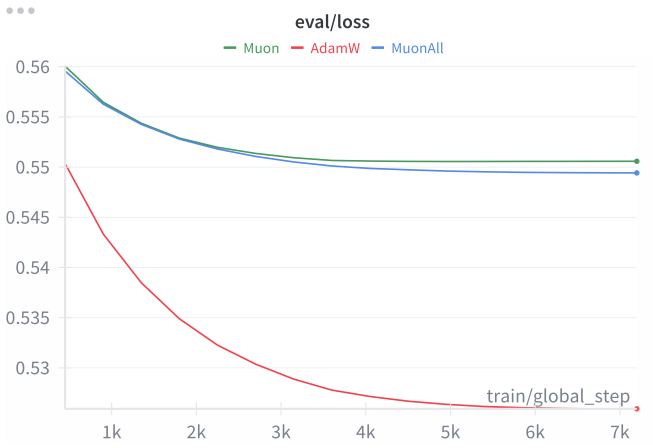

Fig. 2. Validation loss on Qwen 0.5B

B. SFT on SmolLM2-360M

Experiments are performed using the following optimizers: AdamW, MuonAll and Muon. An effective batch size of 80 (batch size: 20, gradient accumulation: 4) is used. Following initial learning rates are used for AdamW and MuonAll: 2 x 10^{-4}, 8 x 10^{-4} respectively. For Muon, two learning rates need to be set: 7 x 10^{-4}(for matrix parameters) and 2 x 10^{-4} (for non-matrix parameters). The training and validation loss

for each experiment can be found in table IV. Validation loss curves can be found in figure 4. The finetuned versions are evaluated over the benchmarks. The benchmarking scores can be found in table II.

The performance of Muon and MuonAll is comparable with AdamW. In the benchmarks where AdamW surpasses Muon; MuonAll surpasses Muon as well. Muon versions perform better than AdamW in major benchmark MMLU. The figure 3 gives a good visualization of the benchmarking scores.

TABLE II
BENCHMARK EVALUATION RESULTS OF SMOLLM2-360M

Benchmark	AdamW	MuonAll	Muon
MMLU 5-shot	26.26%	**27.08%**	26.65%
HellaSwag 10-shot	**57.95%**	56.80%	57.09%
PIQA 0-shot	**73.01%**	72.36%	71.76%
ARC-Easy 25-shot	**64.65%**	63.17%	60.61%
BoolQ 0-shot	40.40%	44.34%	**44.46%**
Winogrande 5-shot	**59.43%**	58.33%	58.64%
OpenBookQA 0-shot	**38.60%**	37.00%	35.80%
LAMBADA (OpenAI variant) 0-shot	**53.62%**	52.53%	50.42%

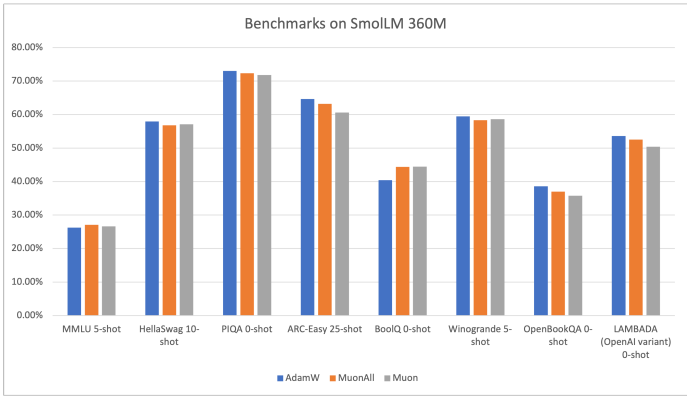

Fig. 3. Benchmarks on SmolLM 360M

Fig. 4. Validation loss on SmolLM 360M

C. SFT on GPT2-medium

AdamW, MuonAll and Muon are used in the SFT experiments for GPT2-medium model. An effective batch size of 60 (batch size: 15, gradient accumulation: 4) is used. Following initial learning rates are used for AdamW and MuonAll: 5 x 10^{-4}, 9 x 10^{-4} respectively. For Muon, two learning rates need to be set: 9 x 10^{-4}(for matrix parameters) and 5 x 10^{-4} (for non-matrix parameters). The training and validation loss for each experiment after two epochs can be found in table IV.The finetuned versions are evaluated over the benchmarks. The benchmarking scores can be found in table III.

The performance of Muon and MuonAll is comparable with AdamW. In the benchmarks where AdamW outcompetes Muon; MuonAll outcompetes Muon as well. Muon versions perform better than AdamW in major benchmark MMLU. The figure 5 gives a good visualization of the benchmarking scores.

TABLE III
BENCHMARK EVALUATION RESULTS OF GPT2 MEDIUM

Epoch	1			2		
Benchmark	Adam	MuonAll	Muon	Adam	MuonAll	Muon
MMLU 5-shot	24.14%	**25.25%**	24.27%	24.09%	**25.05%**	24.65%
HellaSwag 10-shot	26.60%	28.91%	**30.22%**	26.62%	28.90%	**30.27%**
PIQA 0-shot	53.65%	60.01%	**60.45%**	52.39%	**60.17%**	59.90%
ARC-Easy 25-shot	29.55%	35.56%	**37.33%**	29.92%	35.35%	**37.04%**
BoolQ 0-shot	**55.20%**	51.28%	49.30%	**54.65%**	51.25%	51.44%
Winogrande 5-shot	49.96%	**50.51%**	48.86%	49.57%	**50.75%**	48.62%
OpenBookQA 0-shot	**26.20%**	25.80%	25.20%	25.80%	25.60%	**26.20%**

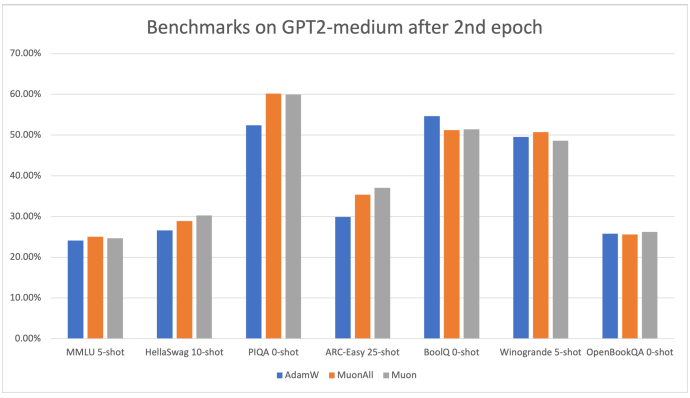

Fig. 5. Benchmarks on GPT2-medium after 2nd epoch

TABLE IV
TRAINING AND VALIDATION LOSSES

Model	Optimizer	Training loss	Validation loss
QWEN2-0.5B	AdamW	0.54213	0.52592
	MuonAll	0.56804	0.54942
	Muon	0.56429	0.55058
SMOLLM2-360M	AdamW	0.55882	0.54377
	MuonAll	0.59449	0.57391
	Muon	0.57803	0.56029
GPT2 MEDIUM	AdamW	0.0007	0.000026537
	MuonAll	0.6503	0.52772
	Muon	0.645	0.52885

V. CONCLUSION

This paper has addressed two possible directions that emerged after showing Muon's scalability in LLM training: (1) Can all parameters be integrated inside Muon instead of a hybrid approach using AdamW? (2) Will Muon be effective if used for SFT on already available public pretrained models? For the first question, we modify the original Muon architecture and introduce MuonAll which incorporates all the parameters in it. For the second question, we provide an extensive analysis of SFT experiments on public pretrained base models: Qwen2-0.5B, SmolLM-360M, GPT2-medium by comparing performance of AdamW, Muon and MuonAll. Muon and MuonAll finetuned models perform on par with the AdamW-finetuned model. For the benchmarks where AdamW outperforms Muon, MuonAll outperforms Muon as well. For the same number of tokens, Muon and MuonAll achieve this with a higher wall-clock time. While this work demonstrates the effectiveness of Muon for efficient and stable finetuning of LLMs, several aspects remain unexplored. Future research could investigate extensions of Muon beyond spectral norm constraints, exploring generalized matrix norms such as Schatten-p or trace norms to better balance convergence speed and model generalization.

REFERENCES

[1] OpenAI *et al.*, "Gpt-4o system card," *arXiv preprint arXiv:2410.21276*, 2024. [Online]. Available: https://arxiv.org/abs/2410.21276.

[2] DeepSeek-AI *et al.*, "Deepseek-v2: A strong, economical, and efficient mixture-of-experts language model," *arXiv preprint arXiv:2405.04434*, 2024. [Online]. Available: https://arxiv.org/abs/2405.04434.

[3] Gemini Team *et al.*, "Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context," *arXiv preprint arXiv:2403.05530*, 2024. [Online]. Available: https://arxiv.org/abs/2403.05530.

[4] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2017. [Online]. Available: https://arxiv.org/abs/1412.6980.

[5] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," *arXiv preprint arXiv:1711.05101*, 2019. [Online]. Available: https://arxiv.org/abs/1711.05101.

[6] H. Liu *et al.*, "Sophia: A scalable stochastic second-order optimizer for language model pre-training," *arXiv preprint arXiv:2305.14342*, 2024. [Online]. Available: https://arxiv.org/pdf/2305.14342.pdf.

[7] K. Jordan, Y. Jin, V. Boza, J. You, F. Cesista, L. Newhouse, and J. Bernstein, "Muon: An optimizer for hidden layers in neural networks," 2024. [Online]. Available: https://kellerjordan.github.io/posts/muon/.

[8] H. Yuan *et al.*, "Mars: Unleashing the power of variance reduction for pretraining at scale," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2025. [Online]. Available: https://openreview.net/forum?id=NrcKQ3ASLZ.

[9] R. Vyas *et al.*, "Deconstructing what makes a good optimizer for autoregressive language modeling," *arXiv preprint arXiv:2407.07972*, 2024. [Online]. Available: https://arxiv.org/abs/2407.07972.

[10] X.-L. Li and F. Orabona, "On the convergence of stochastic gradient descent with adaptive stepsizes," 2018. [Online]. Available: [URL if available].

[11] X.-L. Li and F. Orabona, "A simple and provably convergent algorithm for asynchronous sgd," 2018. [Online]. Available: [URL if available].

[12] O. Pooladzandi and X.-L. Li, "Curvature-informed sgd via general purpose lie-group preconditioners," *arXiv:2402.04553 [cs.LG]*, 2024. [Online]. Available: https://arxiv.org/abs/2402.04553.

[13] Y. Li, X. Liang, J. Liu, and H. Zhou, "Multi-strategy equilibrium optimizer: An improved meta-heuristic tested on numerical optimization and engineering problems," *PLoS ONE*, vol. 17, no. 10, 2022. [Online]. Available: https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0276210.

[14] R. Zhao, X.-L. Li, *et al.*, "Deconstructing what makes a good optimizer for autoregressive language modeling," *arXiv preprint arXiv:2407.07972*, 2024. [Online]. Available: https://arxiv.org/abs/2407.07972.

[15] S. Pethick *et al.*, "A stable whitening optimizer for efficient neural network training," *arXiv preprint arXiv:2506.07254*, 2025. [Online]. Available: https://arxiv.org/abs/2506.07254.

[16] J. Liu *et al.*, "Muon is scalable for llm training," *arXiv preprint arXiv:2502.16982*, 2025. [Online]. Available: https://arxiv.org/abs/2502.16982.

[17] J. Bernstein and L. Newhouse, "Old optimizer, new norm: An anthology," *arXiv:2409.20325 [cs.LG]*, 2024. [Online]. Available: https://arxiv.org/abs/2409.20325.

[18] J. Li and M. Hong, "A note on the convergence of muon," *arXiv preprint arXiv:2502.02900*, 2025. [Online]. Available: https://arxiv.org/abs/2502.02900.

[19] N. Sato, H. Naganuma, and H. Iiduka, "Convergence bound and critical batch size of muon optimizer," *arXiv preprint arXiv:2507.01598*, 2025. [Online]. Available: https://arxiv.org/abs/2507.01598.

[20] W. Shen, R. Huang, M. Huang, C. Shen, and J. Zhang, "On the convergence analysis of muon," *arXiv preprint arXiv:2505.23737*, 2025. [Online]. Available: https://arxiv.org/abs/2505.23737.

[21] L. Chen, J. Li, and Q. Liu, "Muon optimizes under spectral norm constraints," *arXiv preprint arXiv:2506.15054*, 2025. [Online]. Available: https://arxiv.org/abs/2506.15054.

[22] C. Si, D. Zhang, and W. Shen, "Adamuon: Adaptive muon optimizer," *arXiv preprint arXiv:2507.11005*, 2025. [Online]. Available: https://arxiv.org/abs/2507.11005.

[23] B. Thérien, X. Huang, I. Rish, and E. Belilovsky, "Muloco: Muon is a practical inner optimizer for diloco," *arXiv preprint arXiv:2505.23725*, 2025. [Online]. Available: https://arxiv.org/abs/2505.23725.

[24] Essential AI *et al.*, "Practical efficiency of muon for pretraining," *arXiv preprint arXiv:2505.02222*, 2025. [Online]. Available: https://arxiv.org/abs/2505.02222.

[25] A. Yang, B. Yang, B. Hui, *et al.*, "Qwen2 technical report," *arXiv:2407.10671 [cs.CL]*, 2024. [Online]. Available: https://arxiv.org/abs/2407.10671.

[26] L. B. Allal, A. Lozhkov, E. Bakouch, *et al.*, "Smollm2: When smol goes big—data-centric training of a small language model," *arXiv:2502.02737 [cs.CL]*, 2025. [Online]. Available: https://arxiv.org/abs/2502.02737.

[27] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever, *et al.*, "Language models are unsupervised multitask learners." OpenAI Blog, 2019.

[28] W. Lian, B. Goodson, E. Pentland, A. Cook, C. Vong, and Teknium, "Openorca: An open dataset of gpt augmented flan reasoning traces." HuggingFace repository, 2023. [Online]. Available: https://huggingface.co/datasets/Open-Orca/OpenOrca.

[29] D. Hendrycks, C. Burns, S. Basart, A. Zou, M. Mazeika, D. Song, and J. Steinhardt, "Measuring massive multitask language understanding," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2021. [Online]. Available: https://arxiv.org/abs/2009.03300.

[30] P. Clark *et al.*, "Think you have solved question answering? try arc," *arXiv preprint arXiv:1803.05457*, 2018. [Online]. Available: https://arxiv.org/abs/1803.05457.

[31] R. Zellers, A. Holtzman, Y. Bisk, A. Farhadi, and Y. Choi, "Hellaswag: Can a machine really finish your sentence?," in *Proc. 57th Annu. Meet. Assoc. Comput. Linguistics (ACL)*, 2019.

[32] K. Sakaguchi, R. L. Bras, C. Bhagavatula, and Y. Choi, "Winogrande: An adversarial winograd schema challenge at scale," *arXiv preprint arXiv:1907.10641*, 2019. [Online]. Available: https://arxiv.org/abs/1907.10641.

[33] Y. Bisk, R. Zellers, R. L. Bras, J. Gao, and Y. Choi, "Piqa: Reasoning about physical commonsense in natural language," *arXiv preprint arXiv:1911.11641*, 2019. [Online]. Available: https://arxiv.org/abs/1911.11641.

[34] C. Clark, K. Lee, M.-W. Chang, T. Kwiatkowski, M. Collins, and K. Toutanova, "Boolq: Exploring the surprising difficulty of natural yes/no questions," in *Proc. NAACL-HLT*, (Minneapolis, MN, USA), pp. 2924–2936, 2019. [Online]. Available: https://aclanthology.org/N19-1300.pdf.

[35] K. Cobbe, V. Kosaraju, M. Bavarian, M. Chen, H. Jun, L. Kaiser, M. Plappert, J. Tworek, J. Hilton, R. Nakano, C. Hesse, and J. Schulman, "Training verifiers to solve math word problems," *arXiv preprint arXiv:2110.14168*, 2021. [Online]. Available: https://arxiv.org/abs/2110.14168.

[36] T. Mihaylov, P. Clark, T. Khot, and A. Sabharwal, "Can a suit of armor conduct electricity? a new dataset for open book question answering," in *Proc. Empirical Methods Natural Lang. Process. (EMNLP)*, (Brussels, Belgium), pp. 2381–2391, 2018. [Online]. Available: https://aclanthology.org/D18-1260.

[37] OpenAI, "Lambada (openai variant)." Dataset, 2024. [Online]. Available: https://huggingface.co/datasets/EleutherAI/lambada_openai.