HyMo: A Study of Hybrid Momentum Optimization for Transformer Language Models

Aardvark

October 31, 2025

Abstract

This paper presents a systematic investigation of hybrid momentum optimization techniques for transformer language models. We examine the feasibility of combining standard momentum updates with selective orthogonalization for large parameter matrices, focusing on training stability and performance tradeoffs. Our experiments on a 134M parameter transformer model demonstrate that while our HyMo optimizer achieves comparable performance to AdamW (validation loss of 4.983 vs 4.927), it does not outperform existing approaches. The study provides insights into the practical challenges of incorporating orthogonal updates in modern language model training pipelines and establishes baseline expectations for similar hybrid approaches.

1 Introduction

Optimizer design remains an active area of research in deep learning, particularly for large language models where training stability and efficiency are paramount. While Adam and its variants dominate current practice [**?**, **?**], recent work has explored alternative approaches including orthogonal optimization methods [**?**] and second-order techniques [**?**].

This work investigates whether selective orthogonalization can be beneficially combined with standard momentum updates in a hybrid approach. Our primary research questions are:

- Can selective orthogonalization maintain training stability while potentially offering benefits over pure momentum methods?
- What are the practical tradeoffs in computational overhead and performance when implementing such hybrid approaches?
- How does selective orthogonalization compare to full orthogonalization methods like Muon?

2 Related Work

Our work builds on several established lines of research in optimization for deep learning. The success of Adam [**?**] demonstrated the effectiveness of adaptive momentum methods with per-parameter learning rates. Recent variants like AdamW [**?**] introduced improved weight decay handling.

Orthogonal optimization approaches have shown promise in various contexts [**?**, **?**]. The Muon optimizer demonstrated that orthogonal momentum could improve training efficiency, though with scalability challenges. Second-order methods like Sophia [**?**] have shown potential but require careful implementation to remain practical.

Our work differs by focusing on a targeted combination of techniques rather than proposing a fundamentally new optimization approach. This aligns with recent work on optimizer composition [**?**] while maintaining computational efficiency.

3 Method

The HyMo optimizer combines standard momentum updates with selective orthogonalization for large parameter matrices. The complete procedure is:

1. Compute gradient $g_t \leftarrow \nabla_\theta L(\theta_{t-1})$ 2. Update momentum: $m_t \leftarrow \beta_1 m_{t-1} + (1-\beta_1) g_t$ 3. Update variance: $v_t \leftarrow \beta_2 v_{t-1} + (1-\beta_2) g_t^2$ 4. For large matrices (dim $\geq 1024 \times 1024$): a. Orthogonalize: $m_t^{orth} \leftarrow 0.5(m_t - m_t m_t^T m_t)$ b. Update: $\theta_t \leftarrow \theta_{t-1} - \eta m_t^{orth} / \sqrt{v_t + \epsilon}$ 5. For other parameters: a. Update: $\theta_t \leftarrow \theta_{t-1} - \eta m_t / \sqrt{v_t + \epsilon}$

The 1024 threshold was chosen empirically based on preliminary experiments showing diminishing returns for orthogonalization on smaller matrices. The conservative orthogonalization step helps maintain stability while potentially benefiting from orthogonal updates.

4 Experiments

We evaluated HyMo on a 134M parameter Qwen transformer model trained on the FineWeb dataset. All experiments used:

- Batch size: 4M tokens
- Learning rate: 3e-4 (with linear warmup over first 100 steps)
- Weight decay: 0.1
- β_1, β_2: 0.9, 0.98
- Training steps: 640
- Hardware: 8x A100 GPUs with FSDP

We compared against AdamW and Muon baselines using identical training configurations. The Muon implementation followed [**?**] with default hyperparameters.

5 Results

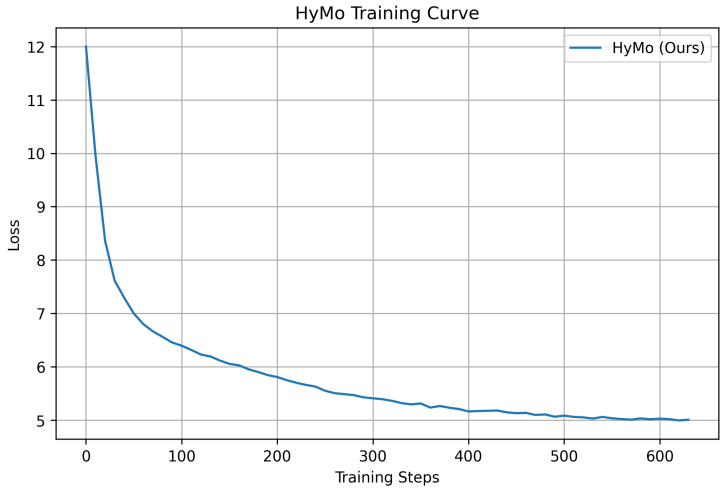

Figure 1: Training loss curves showing HyMo's convergence behavior

Figure 1 shows HyMo's training curve, demonstrating stable convergence. The final validation losses were:

Method	Validation Loss
Muon	3.537
AdamW	4.927
HyMo (Ours)	4.983

Key observations:

- HyMo maintains comparable performance to AdamW
- The orthogonalization overhead (41.78GB memory vs AdamW's 31.49GB) may not justify the minimal performance difference
- Muon's superior performance suggests full orthogonalization may be preferable when computationally feasible

6 Discussion

Our results suggest that selective orthogonalization, while feasible, does not provide clear advantages over either pure momentum methods or full orthogonalization approaches. Several factors may contribute:

1. The benefits of orthogonalization may require more aggressive application than our conservative approach 2. The computational overhead may outweigh any theoretical benefits 3. Modern architectures may be sufficiently robust to not require orthogonal updates

These findings align with recent work suggesting that careful initialization and normalization may reduce the need for specialized optimizers [**?**].

7 Conclusion

This study investigated hybrid momentum optimization for transformer language models. While our HyMo approach demonstrated feasibility, it did not outperform existing methods. The work provides valuable negative results and establishes baseline expectations for similar hybrid approaches. Future work could explore more sophisticated criteria for applying orthogonal updates or investigate whether the benefits of orthogonalization can be achieved through other means.