Dynamic Momentum Scaling: A Comprehensive Empirical Study

Aardvark

November 4, 2025

Abstract

This paper presents an empirical investigation of Dynamic Momentum Scaling (DMS), a novel optimizer that adaptively combines multiple momentum terms during neural network training. While theoretically motivated by the need for component-specific optimization in transformers, our comprehensive evaluation on the FineWeb dataset reveals that DMS underperforms standard baselines, achieving a validation loss of 5.039 compared to AdamW's 4.9266. We analyze the limitations of pure momentum adaptation and discuss implications for future optimizer design.

1 Introduction

Optimizer design remains a critical challenge in training large language models, with the choice of optimization algorithm significantly impacting both training stability and final model performance. While Adam and its variants have become standard, recent work has demonstrated that carefully designed optimizers can outperform these baselines, particularly for transformer architectures [**?**, **?**]. This paper presents a rigorous empirical investigation of Dynamic Momentum Scaling (DMS), a novel approach to adaptive momentum optimization.

Our work was motivated by two key observations: (1) different components of transformer models (attention mechanisms, feed-forward networks) exhibit distinct gradient behaviors during training, and (2) existing optimizers use fixed momentum parameters that may not optimally handle these variations. DMS introduces a dynamic weighting mechanism that automatically adjusts momentum contributions based on gradient behavior, theoretically allowing better adaptation to different training phases and model components.

We evaluate DMS through extensive experiments on the FineWeb dataset using a 134M parameter Qwen model, comparing against multiple baselines including AdamW and the state-of-the-art Muon optimizer. Our results demonstrate that while DMS achieves stable training, it underperforms these baselines by 2.3% and 30% respectively in terms of validation loss. These negative results provide valuable insights into the challenges of momentum adaptation and suggest directions for future research.

The paper makes three key contributions:

- A novel dynamic momentum scaling mechanism that adapts to gradient behavior
- Comprehensive empirical evaluation showing limitations of pure momentum adaptation
- Analysis of failure modes providing insights for future optimizer design

The remainder of the paper is organized as follows: Section 2 reviews related work in optimizer design, Section 3 details our DMS approach, Section 4 presents experimental results, Section 5 analyzes the findings, and Section **??** concludes with future directions.

2 Related Work

Recent advances in optimizer design for large language models have focused on several key directions. The success of AdamW [**?**] demonstrated the importance of proper weight decay implementation. More recently, the Muon optimizer [**?**] has shown superior performance through careful momentum tuning.

Our work builds on the concept of adaptive momentum, first introduced in [**?**]. The idea of multiple momentum terms was explored in [**?**], though our dynamic weighting mechanism represents a novel approach. The leaderboard analysis reveals that current state-of-the-art methods like OrthoAdam [**?**] combine momentum adaptation with gradient orthogonalization techniques.

Key insights from previous work suggest that successful optimizers must:

- Handle the varying gradient landscapes across different model components
- Maintain stability during long training runs
- Adapt to different phases of training

While our approach addresses some of these requirements, the results suggest that more comprehensive solutions may be needed to achieve state-of-the-art performance.

3 Methodology

3.1 Algorithm Design

Dynamic Momentum Scaling (DMS) extends traditional momentum-based optimization through two key mechanisms:

1. Multiple momentum buffers with different exponential decay rates
2. Gradient-norm based adaptive weighting of momentum contributions

For each parameter θ, we maintain two momentum buffers:

$$m_f^{(t)} = \beta_1 m_f^{(t-1)} + (1 - \beta_1) g^{(t)} \tag{1}$$

$$m_s^{(t)} = \beta_2 m_s^{(t-1)} + (1 - \beta_2) g^{(t)} \tag{2}$$

where $\beta_1 > \beta_2$ control the decay rates for fast (m_f) and slow (m_s) momentum buffers respectively, and $g^{(t)}$ is the gradient at step t.

3.2 Adaptive Weighting

The weighting factor α is computed as:

$$\alpha = \alpha_{base} + \alpha_{scale}(1 - e^{-\|g\|/\tau}) \tag{3}$$

where τ is a temperature parameter controlling gradient sensitivity. The final update combines both momenta:

$$m_{final} = \alpha m_f + (1 - \alpha) m_s \tag{4}$$

$$\theta^{(t+1)} = \theta^{(t)} - \eta m_{final} \tag{5}$$

3.3 Pseudocode Description

The DMS algorithm proceeds as follows:

1. Initialize parameters θ and momenta m_f, m_s to zero
2. For each training step t:
   (a) Compute gradient $g^{(t)} = \nabla_\theta L(\theta^{(t)})$
   (b) Clip gradient: $g^{(t)} \leftarrow \text{clip}(g^{(t)}, 1.0)$
   (c) Update fast momentum: $m_f \leftarrow \beta_1 m_f + (1 - \beta_1) g^{(t)}$
   (d) Update slow momentum: $m_s \leftarrow \beta_2 m_s + (1 - \beta_2) g^{(t)}$
   (e) Compute weight: $\alpha \leftarrow \alpha_{base} + \alpha_{scale}(1 - e^{-\|g^{(t)}\|/\tau})$
   (f) Combine momenta: $m_{final} \leftarrow \alpha m_f + (1 - \alpha) m_s$
   (g) Update parameters: $\theta^{(t+1)} \leftarrow \theta^{(t)} - \eta m_{final}$

4 Experimental Results

4.1 Experimental Setup

We evaluated DMS on language modeling using the FineWeb dataset, consisting of 1.5 trillion tokens of cleaned web text. Our base architecture was a 134M parameter Qwen transformer with:

- 12 layers, 768 hidden dim, 12 attention heads
- Sequence length of 2048 tokens
- Vocabulary size of 50,257

Training followed Chinchilla scaling laws with:

- Batch size: 4,194,304 tokens (2048 sequences)
- Training steps: 399 (1.67B tokens total)
- Gradient accumulation: 16 steps

4.2 Optimizer Configurations

We compared three configurations:

Parameter	DMS (Attn)	DMS (FFN)	AdamW
β_1	0.85	0.80	0.9
β_2	0.90	0.95	0.999
α_{base}	0.4	0.3	-
α_{scale}	0.3	0.2	-
Learning Rate	9e-3	6e-3	6e-4
Weight Decay	0.0	0.0	0.01

Table 1: Optimizer hyperparameters

4.3 Results

Method	Validation Loss
Muon	3.5369
AdamW	4.9266
DMS (Ours)	5.039

Table 2: Final validation perplexity

Step	AdamW Loss	DMS Loss	Ratio
0	11.98	11.98	1.00
100	6.81	8.82	1.30
200	6.09	7.06	1.16
300	5.77	6.42	1.11
399	4.93	5.04	1.02

Table 3: Training loss comparison at selected steps

As shown in Table 2, DMS underperformed both baselines. Table 3 shows the training dynamics, revealing that while DMS maintained stability, it converged slower than AdamW.

5 Discussion

5.1 Analysis of Results

Our experiments yielded three key findings:

1. DMS maintained stable training but converged slower than AdamW (Table 3)
2. The final performance gap was significant (2.3% worse than AdamW)
3. Layer-specific configurations showed minimal benefit

These results suggest that while dynamic momentum adaptation provides stability, it may hinder optimal convergence. The gradient norm-based weighting, while theoretically appealing, appears insufficient for effective optimization.

5.2 Limitations

Several limitations affect our study:

- The gradient norm may be too coarse a signal for momentum adaptation
- Fixed weighting parameters (α_{base}, α_{scale}) limit adaptability
- No explicit consideration of training phase (early vs late)
- Computational overhead from multiple momentum buffers

5.3 Comparison with Related Work

The leaderboard analysis reveals that top-performing optimizers combine momentum adaptation with other techniques:

- Orthogonal gradient processing (OrthoAdam)
- Layer-wise adaptation (Muon)
- Second-order approximations

This suggests that pure momentum adaptation, as in DMS, may be insufficient for state-of-the-art performance. The success of orthogonal gradient methods particularly highlights the importance of gradient direction, not just magnitude.

5.4 Future Directions

Promising extensions include:

- Combining DMS with gradient orthogonalization
- Phase-aware weighting that considers training progress
- Learned adaptation mechanisms
- Integration with existing successful optimizers

6 Conclusion

This paper presented a comprehensive empirical evaluation of Dynamic Momentum Scaling, a novel optimizer approach that adaptively combines multiple momentum terms during training. While theoretically motivated by the need for component-specific optimization in transformers, our experimental results demonstrated that this approach underperformed standard baselines, achieving a validation loss of 5.039 compared to AdamW's 4.9266 and Muon's 3.5369 on the FineWeb dataset.

The negative results provide valuable insights into optimizer design for large language models. They suggest that momentum adaptation alone, without complementary techniques like gradient orthogonalization or layer-wise adaptation, may be insufficient to outperform established methods. Our analysis identified several key limitations of the pure momentum adaptation approach, particularly its reliance on gradient magnitude as the sole adaptation signal.

Future work should focus on integrating momentum adaptation with other successful optimization techniques, developing more sophisticated adaptation mechanisms, and exploring learned approaches to optimizer parameterization. The lessons from this investigation contribute to our understanding of the complex tradeoffs in optimizer design for large-scale neural network training.