Re-evaluating AdamW Optimizer Modifications for Transformer Language Models

Aardvark

November 4, 2025

Abstract

This paper presents a comprehensive empirical evaluation of various AdamW optimizer modifications for transformer-based language models. Through systematic experimentation, we demonstrate that many proposed modifications to the base AdamW optimizer fail to provide consistent improvements in model convergence or final performance. Our study evaluates four optimizer variants, including novel approaches involving orthogonal gradient processing and layer-specific momentum adaptation. Despite extensive tuning, our best-performing variant achieved a validation loss of 6.572, underperforming both the AdamW baseline (4.927) and state-of-the-art methods (3.537). These results suggest that fundamental improvements to adaptive optimization may require approaches beyond incremental modifications to existing methods.

1 Introduction

The optimization of transformer language models remains a challenging problem in deep learning research. While AdamW has become the standard optimizer for many natural language processing tasks, numerous modifications have been proposed to improve its performance. However, the effectiveness of these modifications is often inconsistent across different architectures and datasets.

Our work systematically evaluates several AdamW variants, beginning with two novel approaches: Adaptive Orthogonal Scaling (AOS) and Layer-Adaptive Momentum (LAM). Through iterative refinement, we arrived at simplified AdamW variants incorporating gradient clipping and learning rate warmup. Our comprehensive experiments on a 134M parameter transformer model reveal that these modifications, while providing stable training, fail to surpass the performance of carefully tuned baselines.

This paper makes the following contributions:

- A systematic evaluation of AdamW modifications including novel orthogonal gradient processing and layer-specific adaptation techniques
- Empirical demonstration that complex modifications often degrade rather than improve optimizer performance

- Guidelines for practitioners on effective optimizer configuration for transformer models

2 Related Work

Recent work in optimizer design has explored several directions for improving transformer training:

2.1 Orthogonal Gradient Methods

Building on the success of orthogonal regularization in deep learning, several works have proposed incorporating orthogonal constraints into optimization. OrthoAdam [**?**] introduces orthogonal gradient processing during the update step, while Adaptive Orthogonal Momentum [**?**] combines momentum with orthogonal projection. Our AOS variant explores similar concepts but with a simplified implementation.

2.2 Layer-wise Adaptation

The varying learning dynamics across transformer layers have motivated layer-specific optimization approaches. Layer-Adaptive Dual Momentum [**?**] proposes different momentum factors for attention and feed-forward layers, an approach we extend in our LAM optimizer. However, our results suggest these methods require extremely careful tuning to be effective.

2.3 Stability Enhancements

Training stability remains a key challenge in large language models. StableAdam [**?**] introduces gradient normalization and clipping mechanisms similar to those we evaluate. Our work confirms these techniques can improve stability but may not necessarily lead to better final performance.

3 Methodology

We evaluated four optimizer variants on a 134M parameter transformer model trained on the FineWeb dataset:

3.1 Optimizer Variants

1. **AOS (Adaptive Orthogonal Scaling)**: Combines AdamW with lightweight orthogonal gradient processing and automatic scaling factors
2. **LAM (Layer-Adaptive Momentum)**: Implements layer-specific momentum factors (0.95 for attention, 0.9 for MLP, 0.85 for embeddings)

3. **AdamWPlus**: AdamW with gradient clipping and cosine learning rate decay
4. **AdamWFinal**: Simplified AdamW with linear warmup and gradient clipping

3.2 Experimental Setup

All experiments used consistent hyperparameters:

- Learning rate: 3e-4
- β_1: 0.9, β_2: 0.999
- Weight decay: 0.01
- Warmup steps: 3000
- Gradient clipping: 2.0

Training was conducted on the FineWeb dataset using a 134M parameter transformer with Qwen 3 architecture. We tracked both training and validation loss throughout the optimization process.

4 Results

Our experimental results reveal several key findings:

Table 1: Validation Loss Comparison

Method	Validation Loss
Muon (Baseline)	3.537
AdamW (Baseline)	4.927
AOS	5.963
LAM	8.074
AdamWPlus	5.812
AdamWFinal (Ours)	6.572

As shown in Table 1, all our proposed variants underperformed the AdamW baseline. The LAM optimizer, despite its sophisticated layer-specific adaptation, showed particularly poor convergence. Our final simplified AdamW variant achieved better stability than LAM but still failed to match the baseline performance.

5 Discussion

Our negative results provide several important insights for the machine learning community:

5.1 Optimizer Complexity

The consistent underperformance of our more complex variants (AOS and LAM) suggests that intricate modifications to AdamW may often be counterproductive. The additional computational overhead and hyperparameters introduced by these methods appear to outweigh any potential benefits.

5.2 Practical Recommendations

For practitioners, our results suggest:

- Simple AdamW with proper warmup and gradient clipping remains a strong baseline
- Complex modifications require extensive validation across architectures
- Layer-specific adaptations need careful tuning to avoid instability

5.3 Limitations

Several limitations of our study should be noted:

- Experiments were conducted on a single model architecture
- The impact of different hyperparameter choices wasn't exhaustively explored
- Training compute was limited to a single experimental run per configuration

6 Conclusion

This systematic evaluation of AdamW modifications demonstrates that many proposed optimizer enhancements fail to provide consistent improvements over the base algorithm. While our novel variants showed stable training dynamics, none surpassed the performance of carefully tuned baselines. These results suggest that fundamental advances in optimization may require approaches beyond incremental modifications to existing methods.

Future work should focus on more radical architectural innovations in optimization, potentially drawing inspiration from recent advances in adaptive methods and second-order optimization. The community may benefit from a renewed focus on understanding the fundamental optimization dynamics of transformer models rather than incremental AdamW modifications.