Revisiting AdamW: A Comprehensive Evaluation of Optimizer Modifications for Transformer Language Models

Aardvark

November 5, 2025

Abstract

This paper presents a systematic evaluation of optimizer modifications for transformer language models, comparing them against the standard AdamW implementation. Through extensive experimentation with momentum scaling, parameter grouping, and learning rate adaptation techniques on a 134M parameter Qwen model trained on the FineWeb dataset, we find that AdamW remains remarkably robust. Our results demonstrate that several intuitive modifications either failed to improve performance or degraded training stability. While our attempts to innovate on the optimizer did not yield improvements, the negative results provide valuable insights into optimizer design for large language models and suggest that research efforts might be better directed toward other aspects of model training.

1 Introduction

The optimization of transformer language models presents unique challenges due to their scale and complex loss landscapes. While numerous optimizer variants have been proposed [?, ?, ?], AdamW remains the de facto standard in practice. This work investigates whether careful modifications to AdamW could yield improvements in training efficiency or final model performance.

Our investigation was motivated by several recent papers proposing optimizer innovations [?, ?, ?]. However, as we demonstrate through rigorous experimentation, many promising modifications fail to deliver consistent improvements over the baseline AdamW implementation. Our work contributes to a growing body of literature suggesting that simple, well-tuned optimizers often outperform more complex variants [?].

2 Related Work

Recent work on optimizers for large language models has explored various directions. The adaptive moment estimation (Adam) optimizer [?] introduced individual adaptive learning rates for parameters. Subsequent work identified issues with Adam's weight decay implementation [?], leading to AdamW. More recent innovations include:

- Orthogonal gradient processing [?]
- Layer-wise adaptation [?]
- Memory-efficient variants [?]

While these approaches show promise, they often introduce significant complexity. Our work provides empirical evidence that simpler approaches may be preferable for many applications.

3 Methods

We conducted a series of ablation studies comparing various AdamW modifications against the baseline implementation. Our experimental protocol used:

- Model: Qwen architecture with 134M parameters
- Dataset: FineWeb benchmark (2.9B tokens)
- Training: Chinchilla-optimal token count (1.67B tokens)
- Batch size: 4,194,304 tokens
- Gradient accumulation: 16 steps
- Evaluation: Validation perplexity

We evaluated four primary modifications, each carefully tuned:

1. Multi-timescale momentum adaptation (betas: 0.9, 0.95, 0.99)
2. Layer-specific learning rate scaling (attention layers scaled by 1.2x)
3. Parameter-group aware weight decay (0.1 for weights, 0.0 for biases)
4. Combined momentum-spectral approaches

All experiments used the same random seed and were run on identical hardware to ensure fair comparison.

Table 1: Validation Loss Comparison

Method	Validation Loss
AdamW Baseline	4.927
Our Final Implementation	4.954
Multi-Momentum	11.500
Layer-Specific LR	11.672
Grouped Weight Decay	11.812
Combined Approach	11.937

4 Results

As shown in Table 1, none of our modifications improved upon the AdamW baseline:

These results suggest that AdamW's default configuration is remarkably well-tuned for transformer language models. The performance degradation from our modifications indicates that careful parameter grouping and momentum handling are crucial for stable training.

5 Limitations

Our study has several important limitations:

- Evaluated on a single model size (134M parameters)
- Tested on one dataset (FineWeb)
- Limited hyperparameter search for each variant
- No downstream task evaluation

Future work should validate these findings across different model scales and datasets.

6 Conclusion

Our comprehensive evaluation demonstrates that AdamW remains a robust choice for transformer language model optimization. While the negative results may seem disappointing, they provide valuable insights:

- Simple, well-tuned optimizers can outperform more complex variants
- AdamW's default parameters work well for transformer architectures
- Future work should focus on orthogonal improvements rather than AdamW modifications

These findings suggest that research efforts might be better directed toward areas like curriculum learning or architecture modifications rather than optimizer innovations.