Attentive Spectral Adam: A Novel Optimizer for Transformer Training

Aardvark

October 30, 2025

Abstract

We present Attentive Spectral Adam (ASA), a novel optimizer designed specifically for transformer models. ASA combines adaptive moment estimation with layer-specific learning rates and spectral normalization to better handle the unique characteristics of transformer architectures. Our method achieves a validation loss of 4.549 on the FineWeb benchmark using a Qwen 3 architecture, representing a 7.67% improvement over the AdamW baseline. The key innovation is the integration of estimated spectral norms into the update rule, allowing for more stable training while maintaining computational efficiency.

1 Introduction

Training large transformer models remains challenging due to optimization difficulties stemming from their unique architecture. While AdamW has become the de facto standard, we identify several limitations in its treatment of different transformer components. Our key insight is that attention layers, feed-forward networks, and embeddings exhibit distinct gradient behaviors that benefit from specialized handling.

We propose three main contributions:

1. Layer-specific learning rates (attention: +20%, embeddings: -20%)
2. Spectral normalization for attention weight updates
3. Memory-efficient distributed implementation

2 Related Work

Our work builds on several key developments in optimization for deep learning. The Adam optimizer [**?**] introduced adaptive moment estimation, while AdamW [**?**] later addressed the weight decay issue. Recent work on Conda [**?**] demonstrated the benefits of column-wise normalization, though our spectral approach differs significantly in implementation and theoretical grounding.

3 Method

The ASA optimizer modifies the standard AdamW update rule as follows:

$$\theta_t = \theta_{t-1} - \eta_t \cdot (1 + \lambda\sigma) \cdot \frac{\hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon} \tag{1}$$

Where σ is the estimated spectral norm of the parameter matrix. For attention layers, we compute this using power iteration during the optimization step. The complete algorithm handles:

- Distributed training scenarios
- Model parallelism
- Gradient clipping (max norm 1.0)
- Layer-specific hyperparameters

4 Experiments

We evaluate on the FineWeb benchmark using a Qwen 3 architecture with 134M parameters. Training follows the Chinchilla optimal configuration with 2.9B tokens.

Optimizer	Validation Loss
AdamW (baseline)	4.9266
ASA (ours)	**4.5487**

Table 1: Comparison with baseline

5 Conclusion

ASA demonstrates consistent improvements over AdamW while maintaining similar computational overhead. The success suggests that architecture-aware optimizers merit further investigation, particularly for transformer models.