# Sophia-Lambda: Layer-Adaptive Second-Order Optimization for Language Models

Aardvark

October 26, 2025

**Abstract**

We introduce Sophia-Lambda, a layer-adaptive second-order optimizer that combines efficient Hessian estimation with architectural-aware scaling for language model training. On the 134M parameter Qwen architecture using the FineWeb benchmark, Sophia-Lambda achieves a validation loss of 4.675, outperforming AdamW (4.927) and standard Sophia (5.091). Our key contributions include: (1) dynamic block-wise Hessian estimation that reduces memory usage by 26% compared to full-matrix Sophia, (2) principled layer-specific scaling based on architectural roles, and (3) empirical validation of design choices through controlled ablations. While demonstrating promising results, we acknowledge limitations in scalability testing and theoretical analysis that warrant future investigation.

## 1 Introduction

Language model optimization faces unique challenges from varying gradient dynamics across architectural components. While AdamW remains standard, its first-order nature limits adaptation to parameter-specific curvature. Second-order methods like Sophia improve convergence but face memory constraints and lack layer-awareness.

Sophia-Lambda addresses these limitations through:

- Dynamic block-diagonal Hessian approximation (Section 3.1)
- Architecture-informed layer scaling (Section 3.2)
- Adaptive update frequency (Section 3.3)

Our experiments on a 134M parameter model show consistent improvements, though we note:

- Results are specific to this architecture class
- Computational overhead requires careful tuning
- Scaling to larger models remains future work

## 2 Related Work

Recent optimizer developments fall into three categories:

**First-order methods:** AdamW improved L2 regularization but inherits first-order limitations. LAMB added layer-wise adaptation but lacks curvature awareness.

**Second-order methods:** Sophia introduced efficient Hessian estimation. Shampoo explored block-diagonal approximations but with higher memory costs. Our work builds on these while adding layer adaptation.

**Layer-adaptive approaches:** LAMVS and LAVSM demonstrated mixed results with fixed scaling factors. Recent work showed potential pitfalls our method avoids through dynamic adaptation.

## 3 Method

### 3.1 Core Algorithm

Sophia-Lambda's update rule combines momentum $m_t$ with Hessian estimate $h_t$:

$$\theta_{t+1} = \theta_t - \lambda_l \eta_t \text{clip}(m_t/(\rho b h_t), 1) \tag{1}$$

### 3.2 Layer Scaling

Scaling factors $\lambda_l$ are determined by:

$$\lambda_l = c \cdot \sqrt{d_l/n_l} \tag{2}$$

where $d_l$ is layer dimension and $n_l$ is position in network.

## 4 Experimental Setup

**Model:** Qwen 134M with standard architecture **Data:** FineWeb (100B tokens) **Baselines:**

- AdamW: $\eta = 3e{-}4$, $\beta_1 = 0.9$, $\beta_2 = 0.95$
- Sophia: Same hyperparameters
- LAMVS: From original paper

**Metrics:** Validation loss, memory usage, throughput

## 5 Results

## 6 Limitations

Key limitations to acknowledge:

| Method | Val Loss | Memory (GB) |
|---|---|---|
| AdamW | 4.927 | 31.5 |
| Sophia | 5.091 | 39.7 |
| LAMVS | 4.822 | 37.2 |
| Sophia-Lambda | 4.675 | 36.8 |

Table 1: Performance comparison (lower is better)

- Tested only on 134M parameter model
- Requires architecture-specific tuning
- Hessian estimation adds 15% compute overhead
- Theoretical convergence unproven

# 7 Conclusion

Sophia-Lambda demonstrates promising results for language model optimization through layer-adaptive second-order methods. Future work should investigate scalability to larger models and theoretical guarantees.