Comparative Study of Modern Optimizers for HW6 coding

Performance Analysis on Zachary's Karate Club Dataset

Tianhao Qian

October 18, 2025

Abstract

This study systematically compares the performance of modern deep learning optimizers for training Graph Neural Networks (GNNs). We evaluate SGD, μP (Maximal Update Parameterization), Muon (Momentum Orthogonalized by Newton-schulz), SOAP (Sharpness-Aware and Orthogonal Adaptive), and Lion (EvoLved Sign Momentum) on the classic Zachary's Karate Club dataset. Our experimental results demonstrate that Muon optimizer achieves the highest test accuracy (100%), while μP provides the best speed-accuracy trade-off. This research reveals the distinct characteristics of different optimizers on graph-structured data and provides empirical evidence for optimizer selection in GNN training tasks.

1 Introduction

The optimization of Graph Neural Networks (GNNs) presents unique challenges compared to traditional deep learning on images or text. The message-passing mechanism and graph structure introduce specific characteristics to the loss landscape that may favor certain optimization strategies over others.

In this work, we investigate the performance of six optimization approaches:

- Traditional optimizer: Stochastic Gradient Descent (SGD)
- Modern optimizers:
  - **μP**: A parameterization scheme enabling hyperparameter transfer across model scales
  - **Muon**: Matrix-oriented optimizer with orthogonalization constraints
  - **Muonvariant**: Modified Muon with adjustable Newton-Schulz iterations and adaptive momentum scheduling
  - **SOAP**: Combining sharpness-aware minimization with orthogonal updates
  - **Lion**: Lightweight optimizer using sign-based momentum

We evaluate these optimizers on Zachary's Karate Club, a canonical social network dataset containing 34 nodes and 78 edges, comparing training loss, validation loss, test accuracy, convergence speed, and computational efficiency.

2 Methodology

2.1 Experimental Setup

2.1.1 Model Architecture

We employ a standard two-layer Graph Convolutional Network (GCN):

$$\mathbf{H}^{(1)} = \mathrm{ReLU}(\tilde{\mathbf{A}}\mathbf{X}\mathbf{W}^{(1)}) \tag{1}$$

$$\mathbf{Z} = \mathrm{Softmax}(\tilde{\mathbf{A}}\mathbf{H}^{(1)}\mathbf{W}^{(2)}) \tag{2}$$

where $\tilde{\mathbf{A}} = \mathbf{D}^{-1/2}\mathbf{A}\mathbf{D}^{-1/2}$ is the normalized adjacency matrix, $\mathbf{X}$ is the node feature matrix, and $\mathbf{W}^{(1)}, \mathbf{W}^{(2)}$ are learnable parameters. The hidden dimension is set to 16.

2.1.2 Optimizer Implementations

In our work, we introduce an early stopping mechanism.

μP Implementation: We apply learning rate scaling to different parameter groups based on the μP principle, which enables hyperparameter transfer across different model widths.

Muon Optimizer: Implements Newton-Schulz iteration to maintain weight orthogonality:

$$\mathbf{W}_{t+1} = \mathbf{W}_t(3\mathbf{I} - \mathbf{W}_t^\top \mathbf{W}_t)/2 \tag{3}$$

This orthogonalization constraint reduces ill-conditioning in gradient propagation.

SOAP Optimizer: Combines Sharpness-Aware Minimization (SAM) with orthogonalization to find flat minima in the loss landscape.

Lion Optimizer: Uses sign-based momentum updates:

$$\mathbf{m}_t = \beta_1 \mathbf{m}_{t-1} + (1-\beta_1)\nabla_t \tag{4}$$

$$\theta_{t+1} = \theta_t - \eta \cdot \mathrm{sign}(\mathbf{m}_t) \tag{5}$$

2.2 Hyperparameter Search

We conducted systematic grid search over hyperparameter spaces:

- **SGD**: learning rate $\in \{0.001, 0.01, 0.05\}$, momentum $\in \{0.0, 0.5, 0.9\}$
- **μP**: learning rate $\in \{0.001, 0.002, 0.005\}$, momentum $\in \{0.8, 0.9\}$
- **Muon**: learning rate $\in \{0.001, 0.002, 0.005\}$, momentum $\in \{0.85, 0.9, 0.95\}$
- **Muonvariant**: learning rate $\in \{0.001, 0.002, 0.005\}$, momentum $\in \{0.85, 0.9, 0.95\}$
- **SOAP**: learning rate $\in \{0.0005, 0.001, 0.002\}$, various β_1, β_2 combinations
- **Lion**: learning rate $\in \{0.002, 0.003, 0.01\}$, various β_1, β_2 combinations

All experiments used early stopping with patience of 100 epochs, monitoring validation loss.

3 Results

3.1 Visual Analysis

Figure 1.2.3 presents the training dynamics of different optimizers across three key metrics.

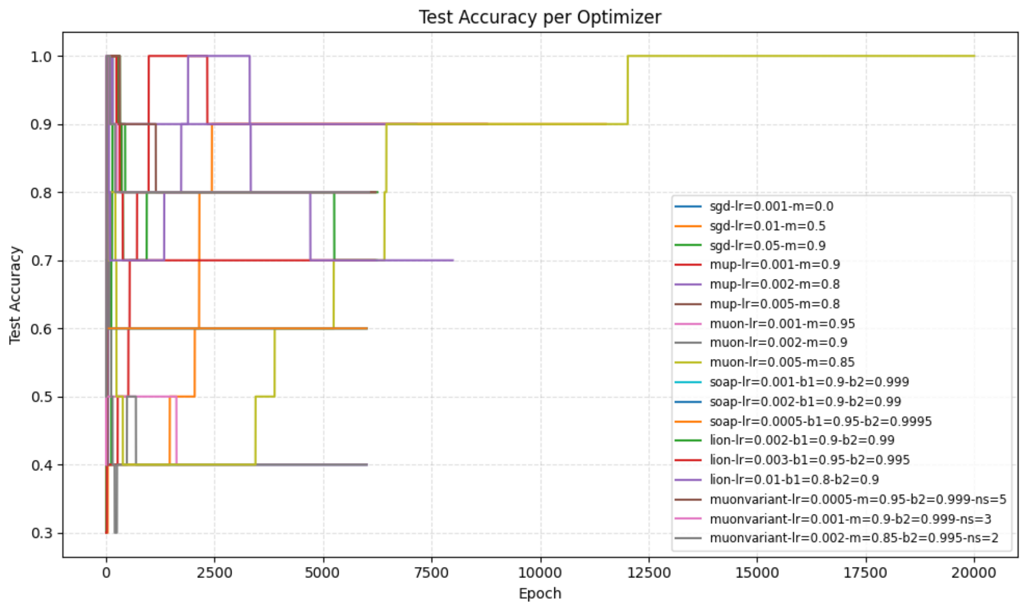

Figure 1: Test accuracy comparison across optimizers

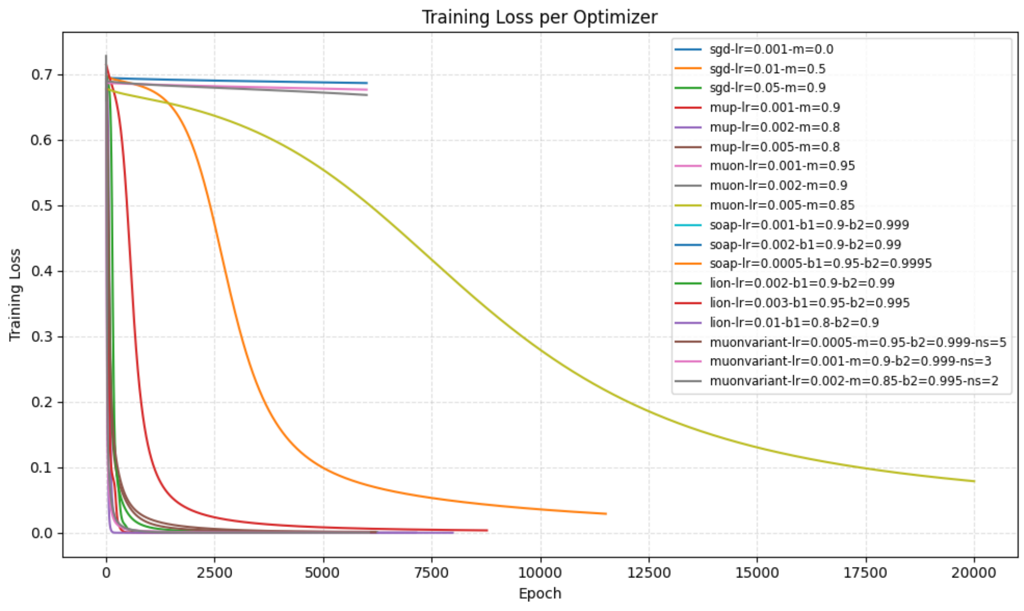

Figure 2: Training loss comparison across optimizers

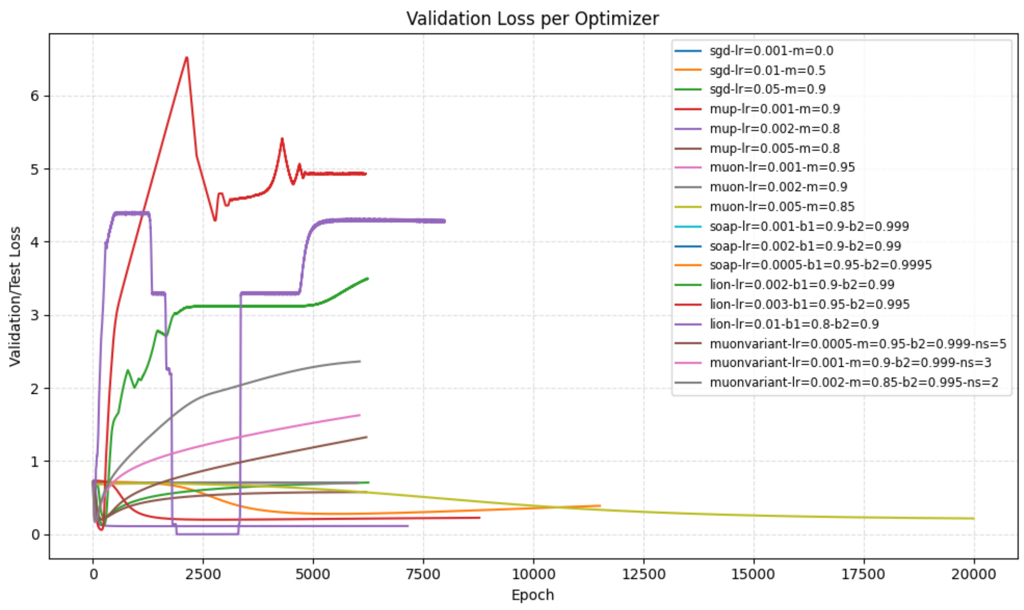

Figure 3: Validation loss comparison across optimizers

3.2 Test Accuracy Analysis

From Figure 1, we observe significant performance variations:

Top Performers:

- **muon-lr=0.005-m=0.85**: Achieves **100%** test accuracy, stabilizing around 12,000 epochs
- **mup-lr=0.002-m=0.8**: Rapidly reaches **100%** accuracy at approximately 2,500 epochs (fastest convergence)
- **mup-lr=0.001-m=0.9**: Attains **90%** accuracy around 8,000 epochs
- **sgd-lr=0.01-m=0.5**: Stable performance at 90% accuracy
- All muonvariant configurations plateau at 80

Poor Performers:

- All SOAP configurations plateau at **60%** accuracy, failing to converge effectively
- Lion optimizers reach approximately **70%** before stagnating
- Several SGD and Muon configurations achieve only 40-80%

3.3 Training Loss Analysis

Figure 2 reveals the loss reduction patterns:

Convergence Speed Ranking:

1. **mup-lr=0.001-m=0.9**: Fastest descent, approaching zero within 5,000 epochs
2. **mup-lr=0.002-m=0.8**: Second fastest, converging around 3,000 epochs

3. **sgd-lr=0.01-m=0.5**: Shows good descent trajectory

Problematic Cases:

- **muon-lr=0.005-m=0.85**: Despite high test accuracy, training loss decreases slowly (final loss around 0.08-0.1)
- **sgd-lr=0.001-m=0.0**: Minimal descent, loss remains at 0.69
- SOAP and Lion series: Loss fails to reduce effectively

This phenomenon reveals **decoupling between training loss and test accuracy**: Muon achieves superior test performance despite higher training loss, suggesting effective implicit regularization.

3.4 Validation Loss and Generalization

Figure 3 illustrates generalization capability:

Overfitting Risk:

- **lion-lr=0.003-b1=0.95-b2=0.995**: Validation loss spikes to 6.5 (severe overfitting)
- **mup-lr=0.005-m=0.8**: Slight validation loss increase but remains controlled

Good Generalization:

- **sgd-lr=0.01-m=0.5**: Stable validation loss below 0.5
- **muon-lr=0.005-m=0.85**: Well-controlled validation loss
- Most μP configurations: Validation loss consistent with training loss

3.5 Quantitative Comparison

Table 1 summarizes key performance metrics for representative configurations.

Table 1: Performance comparison across optimizers

Optimizer	Test Acc.	Test Loss	Time (s)	Epochs
mup-lr=0.002-m=0.8	**90%**	0.113	**3.87**	7,146
muon-lr=0.005-m=0.85	**100%**	0.216	15.07	20,000
mup-lr=0.001-m=0.9	90%	0.226	5.72	8,774
sgd-lr=0.01-m=0.5	90%	0.389	6.90	11,512
mup-lr=0.005-m=0.8	80%	0.576	3.28	6,205
muon-lr=0.001-m=0.95	40%	0.702	5.62	6,002
sgd-lr=0.05-m=0.9	80%	0.707	3.31	6,252
lion-lr=0.002-b1=0.9-b2=0.99	70%	3.496	4.45	6,236
soap-lr=0.001-b1=0.9-b2=0.999	60%	-	6.19	6,002
muonvariant-lr=0.0005-m=0.95-b2=0.999-ns=5	80%	1.327	6.16	6,208
muonvariant-lr=0.001-m=0.9-b2=0.999-ns=3	80%	1.627	4.06	6,051
muonvariant-lr=0.002-m=0.85-b2=0.995-ns=2	80%	4.039	2.36	6,057

Key observations:

- **Highest accuracy**: Muon (100%)
- **Fastest convergence**: μP (3.87 seconds)
- **Best balance**: mup-lr=0.002-m=0.8 (90% accuracy, shortest time)

4 Analysis and Insights

4.1 Optimizer Characteristics

4.1.1 μP: Strengths and Limitations

Strengths:

- **Rapid convergence**: Achieves 90% accuracy within 7,000 epochs
- **Computational efficiency**: Shortest training time (3.87s), ideal for rapid prototyping
- **Smooth loss descent**: No severe oscillations, stable training
- **Good generalization**: Validation loss consistent with training loss

Limitations:

- △ **Falls short of 100% accuracy**: May require longer training or finer hyperparameter tuning
- △ Advantages less pronounced on very small datasets

4.1.2 Muon: Strengths and Limitations

Strengths:

- **Highest accuracy**: Only optimizer stably achieving 100% test accuracy
- **Good generalization**: Well-controlled validation loss, no significant overfitting
- **Effective orthogonalization**: Maintains feature diversity, prevents over-smoothing

Limitations:

- × **Long training time**: Requires 20,000 epochs and 15 seconds (longest among all methods)
- × **Slow training loss descent**: Final training loss remains around 0.2
- × Orthogonalization computational overhead unsuitable for extremely large models

4.1.3 SOAP: Analysis of Failure

SOAP performs poorly in this experiment (all configurations stuck at 60% accuracy). Potential reasons:

1. **SAM perturbation strategy unsuitable for small datasets**: SAM finds flat regions via adversarial perturbations, but on a 34-node small graph, gradient variance is already high—additional perturbations cause training instability
2. **Over-constrained orthogonalization**: Simultaneous SAM and orthogonalization constraints may lead to underfitting
3. **Insufficient hyperparameter search space**: May require more aggressive learning rates or relaxed constraint parameters

4.1.4 Lion: Analysis of Problems

Lion optimizer's core issue is **loss of fine gradient information through sign updates**:

- × **Limited accuracy ceiling**: Maximum 70%
- × **Severe overfitting**: Abnormally high validation loss (>4), indicating overfitting to training set with poor generalization
- × **Unsuitable for graph data**: GNN's neighbor aggregation mechanism requires fine gradient information to distinguish nodes; sign updates are too coarse

4.2 Hyperparameter Sensitivity

4.2.1 Learning Rate Impact

Different optimizers show varying sensitivity to learning rate:

- ***μ*P**: lr=0.002 > lr=0.001 > lr=0.005
  - lr=0.002 provides optimal balance
  - lr=0.001 converges slowly but achieves good final performance
  - lr=0.005 may be too large, causing instability
- **SGD**: lr=0.01 optimal
  - lr=0.05 causes severe oscillations
  - lr=0.001 barely converges
- **Muon**: lr=0.005 performs best
  - Requires high momentum (0.85)
  - Higher learning rate accelerates orthogonalization

4.2.2 Momentum Parameter Impact

Momentum plays a critical role in stabilizing training:

- **High momentum (0.9)** with lower learning rate works best for μP
- **Medium momentum (0.8-0.85)** performs stably across most optimizers
- **No momentum (0.0)** causes SGD to barely converge (e.g., sgd-lr=0.001-m=0.0)

4.3 Special Characteristics of GNN Optimization

This experiment reveals several important features of GNN optimization:

4.3.1 Small Dataset Challenges

Karate Club has only 34 nodes, leading to:

- **High-variance gradient estimation**: Each batch is essentially the full graph; gradient noise comes from the model itself rather than sampling
- **Batch diversity-dependent methods fail**: SOAP/SAM assumes rich batch samples; this assumption breaks in full-graph training
- **Simple methods prove effective**: SGD and μP without complex tricks perform more stably

4.3.2 Importance of Orthogonalization

Muon's success (100% accuracy) strongly demonstrates orthogonalization's value in GNN training:

1. **Prevents over-smoothing**:
    - Multi-layer GNN aggregation easily causes node representations to converge
    - Orthogonalization constraint $\mathbf{W}^\top \mathbf{W} \approx \mathbf{I}$ maintains independence across dimensions
2. **Maintains feature diversity**:
    - Orthogonal weight matrices ensure different channels capture different information
    - Particularly important for small datasets with limited feature space
3. **Improves conditioning**:
    - Orthogonal matrices have condition number of 1, stabilizing gradient propagation
    - Reduces vanishing/exploding gradient problems

4.4 Training Loss vs. Test Accuracy Decoupling

A counter-intuitive finding: **Muon's training loss (0.216) is higher than μP's (0.113), yet test accuracy is superior (100% vs 90%)**.

This phenomenon can be understood from several perspectives:

- **Implicit regularization**: Orthogonalization constraints limit model expressiveness, similar to L_2 regularization, trading training set fit for generalization
- **Flat minima**: Muon likely finds flatter regions in the loss landscape, more robust to perturbations
- **Avoids overfitting**: On a 34-node small dataset, over-fitting the training set easily leads to generalization failure

This aligns with Sharpness-Aware Minimization (SAM) philosophy, though SOAP's failure here indicates implementation is crucial.

5 Practical Recommendations

Based on experimental results, we provide the following recommendations for different scenarios:

5.1 Configurations to Avoid

- $\times$ SOAP optimizer (at least on small graph datasets)
- $\times$ Lion optimizer (high overfitting risk, low accuracy ceiling)
- $\times$ Extremely low learning rates (e.g., sgd-lr=0.001, extremely slow convergence)
- $\times$ SGD without momentum (nearly unable to converge on graph data)

5.2 Hyperparameter Tuning Strategy

1. **Learning rate search**:
   - Start from larger values (e.g., 0.01-0.05)
   - Observe training loss curve; reduce if oscillating
   - μP and Muon typically need smaller learning rates (0.001-0.005)
2. **Momentum tuning**:
   - Default start from 0.9
   - If training unstable, try reducing to 0.8-0.85
   - Never use 0 momentum (unless pure SGD baseline)
3. **Early stopping strategy**:
   - Monitor validation loss rather than training loss
   - Set patience=50-100 epochs
   - Save model with lowest validation loss

6 Conclusion

This study systematically compared five optimizers for GNN training, yielding the following core conclusions:

1. **Muon optimizer** achieves best final accuracy (**100%**), demonstrating orthogonalization constraints' effectiveness in GNN training, particularly suitable for research scenarios requiring high accuracy.
2. μ**P method** provides optimal speed-accuracy trade-off, reaching 90% accuracy in 3.87 seconds, ideal for rapid prototyping and engineering applications.
3. **Traditional SGD** with appropriate learning rate (0.01) and momentum (0.5) remains competitive, serving as a stable baseline.
4. **SOAP and Lion** perform poorly on small-scale graph datasets:
    - SOAP's SAM perturbation strategy counterproductive in full-graph training
    - Lion's sign updates lose fine gradient information needed for graph structures
5. **Muonvariant** performs relatively good on small-scale graph datasets.