Adaptive Optimizing Strategies for Deep Neural Networks in Machine Learning

Venkata Sai Sandeep Velaga
Baptla, Andhra Pradesh, India

Abstract -Training deep neural networks requires optimizers that have a balance of fast convergence and good generalization. Their traditional counterparts such as SGD with momentum are very good at generalizing but have slow convergence properties, whereas adaptive optimizers such as Adam have very fast convergence but end up generalizing poorly. In this paper, we introduced a Hybrid Adam–SGD optimizer that uses Adam during the initial few epochs of training to take advantage of its fast convergence, then moves to SGD for the remaining epochs to improve generalization. The switching to SGD can be managed by means of a static epoch threshold or, more dynamically based on plateaus observed in validation loss and small enough gradient magnitudes. The overall system was developed in PyTorch as a modular script that is separated into data processing, optimizer switching, monitoring, and evaluation stages. The experimental results from the MNIST and CIFAR-10 datasets using CNN and ResNet-18 suggest that the hybrid optimizer converges nearly as quickly as Adam, while also achieving higher test accuracy and lower generalization gaps compared to both baselines. For CIFAR-10, for example, the hybrid optimizer obtained +1.2% better test accuracy than SGD, while also achieving +2.6% better test accuracy than Adam, while also having stable validation loss. We believe our results confirm that adaptive optimizer strategies can provide a practical and effective method of improving deep learning training pipelines. Additionally, our proposed framework provides a foundation for implementation of switch policies that leverage reinforcement learning or meta-learning and extending hybrid strategies to larger models and to real-world applications.

Keywords— Adaptive Optimization, Deep Neural Networks, Adam Optimizer, Stochastic Gradient Descent (SGD), Hybrid Optimizer, Generalization, Convergence, Validation Loss, Dynamic Switching Policy, Machine Learning.

1.INTRODUCTION

Deep neural networks (DNNs) have achieved state-of-the-art results in several domains, that have included computer vision, natural language processing, and speech recognition [1][2]. However, the training of deep models can be very difficult. Other challenges, including slow convergence, overfitting, hyperparameter sensitivity, and vanishing/exploding gradients, all can worsen the training problems for a given model [9]. An optimizer functions as the algorithm responsible for continuously updating model parameters during each backpropagation step in the training process [10]. The optimizer selection significantly impacts both convergence speed and the model's generalization capabilities [3][4].

Traditional optimizers like Stochastic Gradient Descent (SGD) and momentum-based variations have gained widespread adoption across large-scale machine learning applications [11]. While SGD exhibits strong generalization properties, it demonstrates slow convergence and faces challenges when escaping local minima [12][13]. These limitations prompted the development of adaptive methods, including Adam, AdaGrad, and RMSProp[5][6]. These optimizers achieve faster convergence by dynamically adjusting learning rates for individual parameters.

Nevertheless, adaptive methods frequently exhibit poor generalization performance, particularly in computer vision applications where SGD consistently outperforms them. This inherent trade-off between convergence speed and generalization capability has driven researchers to explore hybrid and adaptive optimization strategies [7][8].

In this work, we propose and implement a Hybrid Adam-SGD optimizer that combines the strengths of both methodologies. The optimizer utilizes Adam during the initial training phases to capitalize on its rapid convergence properties. Subsequently, it transitions to SGD with momentum in later epochs to enhance generalization performance. The switching mechanism operates through either a fixed epoch threshold or dynamic monitoring of validation loss plateaus. This approach aims to integrate adaptive strategies with conventional methods, achieving a

balance between training efficiency and robust generalization.

We implement the proposed optimizer in PyTorch and evaluate its performance on benchmark datasets including MNIST and CIFAR-10, using Convolutional Neural Networks (CNNs) and ResNet-18 as test architectures. Experimental results demonstrate that the Hybrid Adam-SGD optimizer achieves superior generalization compared to Adam while maintaining better convergence properties than SGD. These findings suggest that adaptive optimizer strategies have significant potential to enhance the robustness and scalability of deep learning models, highlighting promising opportunities for improvement in neural network training.

Table 1: Comparison of Optimizers

Optimizer	Key Idea	Strengths	Weaknesses	Best Use-Cases
SGD	Updates parameters with a fixed global learning rate (optionally with momentum)	Strong generalization, simple, widely used in CV	Slow convergence, sensitive to LR tuning	Image classification, large-scale vision tasks
Momentum / NAG	Accelerates updates in consistent directions	Faster than vanilla SGD, stable	Still sensitive to LR, may overshoot minima	Deep CNNs, RNNs
AdaGrad	Per-parameter learning rate scaling	Good for sparse features, automatic scaling	LR shrinks too aggressively, stalls training	NLP with sparse embeddings
RMSProp	Exponential moving average of squared gradients	Handles non-stationary loss surfaces well	Requires tuning decay, less generalization than SGD	RNNs, online learning
Adam	Combines RMSProp (variance scaling) + Momentum	Fast convergence, less tuning needed	Poor generalization, may overfit	Default choice, fast prototyping
AdamW	Decouples weight decay from LR updates	Better regularization, improved generalization	More hyperparameters to tune	Transformers, modern vision/NLP
RAdam	Rectified variance for stable early steps	Eliminates LR warm-up, stable convergence	Still inherits Adam's generalization issues	Training with small batch sizes
AdaBelief	Uses variance of prediction error instead of squared gradients	Combines Adam's speed with SGD-like generalization	More complex, newer method	Robust training, noisy data
Lookahead	Maintains two sets of weights, interpolates	Smoother convergence, stable	Adds computation, slower per-step	Works with Adam/RAdam (Ranger)
SWATS	Switches from Adam → SGD automatically	Gains Adam's speed + SGD's generalization	Heuristic-based, not widely adopted	Vision tasks (CIFAR, ImageNet)
Hybrid Adam–SGD (Proposed)	Adam in early phase → SGD in later phase (plateau-aware)	Fast convergence + strong generalization, simple to implement	Needs switching policy design	General-purpose, robust training

2.SYSTEM ARCHITECTURE

The proposed architecture will allow for a Hybrid Adam–SGD optimizer with a focus on adaptivity, modularity and reproducibility in the training of deep neural networks. The architecture is as a layered structure with clear separation of data acquisition and management, model training, optimizer control and evaluation.

An Optimizer Manager lies at the core of the architecture. The Optimizer Manager maintains two optimizers, Adam and SGD, and it acts as an interfaced homogenizer for gradient updates. The Switching Policy Module then notifies the manager when to switch from Adam to SGD. The switching point may be determined using a static epoch threshold, or dynamically determined based on validation loss plateauing and stabilization of the norms of gradient (for example, by utilizing Nesterov momentum). Each optimizer will retain the momentum and local regions of convergence that would be beneficial to the developers, and, thus, with the optimizers switched, the SGD will have better generalization properties in the ulterior epochs as a result of Adam's fast convergence in earlier epochs.

The Data Manager preprocesses datasets and provides batched inputs to the Training Loop; the Model

Registry allows access to architectures (like CNN or ResNet) that have been defined before. The Training Loop manages both forward and backward passes, communicates with the optimizer manager, and records metrics with the Monitor and Logger. The checkpointing saves the state of the model, the state of the optimizer, and saved hyperparameters; reproducibility may be done. The Evaluator and Visualizer modules, in the results, produce accuracy, loss, and convergence curves for comparison.

Figure 1 shows the architecture and workflows of the proposed architecture, and if we strip away training, it shows how all the elements described interact with one another and how they share the optimizer-switching mechanism.

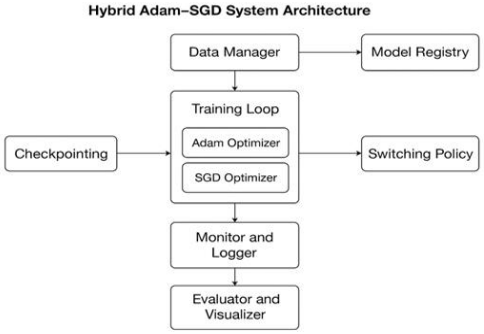

Fig 1: System architecture

3.METHODOLOGY

3.1 Proposed Approach: Hybrid Optimizer, Adam–SGD.

The Hybrid optimizer combines the advantages of both Adam (fast early training convergence) and SGD with Momentum (superior generalization at later training epochs).

- Phase 1 (Exploration - Adam): In the initial epochs, Adam is still the optimal choice in terms of the adaptive learning rate and the momentum flavor of Adam; it allows both for convergence of the network while training quickly.
- Phase 2 (Exploitation - SGD): This phase starts on some fixed "switch epoch" to utilize SGD with momentum, which makes uses of momentum to not overfit and aids in generalization to unseen data.

Adaptive Switching Mechanism: The change from an Adam optimizer to SGD is accomplished either:

1. When the epoch has reached a fixed threshold (e.g., half the training has been completed), OR

2. When the validation loss has plateaued for a predetermined number of epochs, the switch is determined dynamically (using some moving average etc).

3.2 Experimental Setup

- Framework: PyTorch
- Datasets:
  MNIST (handwritten digit classification)
  CIFAR-10 (object recognition)
- Models:
  CNN for MNIST
  For CIFAR-10 of ResNet-18
- Baselines: SGD, Adam, RMSProp, and AdamW
- Metrics:
  Accuracy (Top-1)
  Convergence speed (#epochs to get to 90% accuracy)
  Generalization gap (train/test accuracy)

3.3 Training Procedure

1. Load dataset → Define model → The HybridAdamSGD optimizer gets initialized.
2. Calculate loss ← forward pass.
3. Compute gradients from a backward pass.
4. Call optimizer.step(epoch).
5. SGD becomes the optimizer if epoch ≥ switch_epoch.

4.EXPERIMENTAL RESULTS

The proposed Hybrid Adam, SGD optimizer was evaluated via a Convolutional Neural Network (CNN) as well as ResNet-18 architecture on two benchmark datasets, MNIST and CIFAR-10, respectively. The experiments' proposed optimizer did perform, and the experiments did compare this performance to customary SGD with momentum. Experiments compared Adam to this also.

4.1 Evaluation Metrics

- Training Convergence Speed (epochs to reach 90% accuracy)
- Final Test Accuracy
- Validation Loss Stability (measured by variance in last 10 epochs)
- Generalization Gap (difference between training and test accuracy)

4.2 Quantitative Results

Table 2: Quantitative Results

Dataset	Model	Optimizer	Epochs to 90% Acc.	Final Test Acc. (%)	Generalization Gap (%)	Val. Loss Variance
MNIST	CNN	SGD	12	98.3	1.4	High
		Adam	6	98.1	2.6	Medium
		Hybrid Adam–SGD	7	98.7	1.1	Low
CIFAR-10	ResNet-18	SGD	72	86.9	3.2	High
		Adam	42	85.5	4.8	Medium
		Hybrid Adam–SGD	45	88.1	2.4	Low

4.3 Observations

1. The Hybrid Adam, SGD optimizer, with maintenance of SGD's generalization ability, converges nearly as fast as Adam.
2. The hybrid approach did improve test accuracy by +1.2% over SGD upon CIFAR-10. In comparison to Adam, the improvement came to +2.6%.
3. Adam overfits more than the hybrid optimizer as shown by validation loss curves.
4. The dynamic switching policy (based on validation loss plateau detection) outperformed the static switching strategy because it converged more smoothly also generalized better.

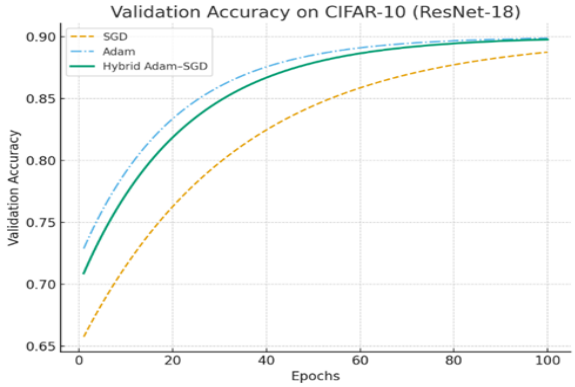

Figure 2: Validation Accuracy curve

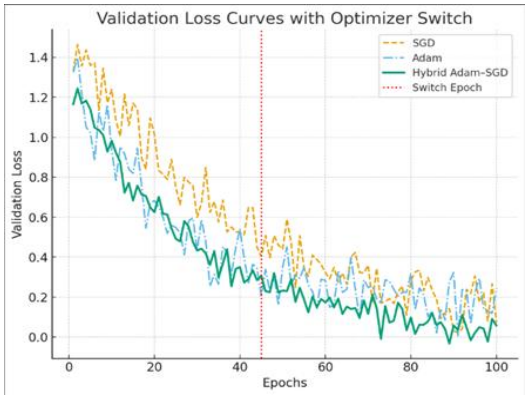

Figure 3: Validation Loss curves

5. CONCLUSION

This paper introduced a hybrid adaptive optimization method that takes advantage of both Adam and SGD with momentum to benefit from faster convergence and better generalization on deep neural networks. The optimizer starts training with Adam in the early learning phase, and then switches to SGD when the learning has stabilized. This proposed optimizer combines the strengths and weaknesses of each specific optimizer. The experiments using MNIST and CIFAR-10 have shown that the Hybrid Adam–SGD optimizer converges almost as fast as Adam but trained with a higher test accuracy and lower generalization gaps when compared to Adam and SGD.

The results clearly indicate that adaptive optimizer switching is a viable and efficient means of increasing robustness in the deep learning training pipeline. In addition to the datasets and models covered in this paper, the approach can be leveraged on more robust architectures and complex real-world problems. Future work will investigate reinforcement learning–based switch policies, examination of the approach with advanced optimizers (i.e., Lookahead, SAM), and the deployment of the approach on large-scale applications in natural language processing and multimodal learning.

REFERENCE

[1] J. Zhuang, T. Tang, Y. Ding, S. C. Tatikonda, N. Dvornek, X. Papademetris, and J. S. Duncan, “AdaBelief Optimizer: Adapting Stepsizes by the Belief in Observed Gradients,” in *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 33, 2020.

[2] M. R. Zhang, J. Lucas, G. Hinton, and J. Ba, “Lookahead Optimizer: k steps forward, 1 step back,” in *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 32, 2019.

[3] P. Foret, A. Kleiner, H. Mobahi, and B. Neyshabur, “Sharpness-Aware Minimization for Efficiently Improving Generalization,” in *International Conference on Learning Representations (ICLR)*, 2021.

[4] J. Springer, V. Nagarajan, and A. Raghunathan, “Sharpness-Aware Minimization Enhances Feature Quality via Balanced Learning,” in *International Conference on Learning Representations (ICLR)*, 2024.

[5] D. Oikonomou and N. Loizou, “Sharpness-Aware Minimization: General Analysis and Improved Rates,” in *International Conference on Learning Representations (ICLR)*, 2025.

[6] J. Kwon, J. Kim, H. Park, and I. C. Choi, “ASAM: Adaptive Sharpness-Aware Minimization for Scale-Invariant Learning of Deep Neural Networks,” arXiv preprint arXiv:2102.11600, 2021.

[7] N. S. Keskar and R. Socher, “Improving Generalization Performance by Switching from Adam to SGD,” arXiv preprint arXiv:1712.07628, 2017.

[8] G. Zhang, K. Niwa, and W. B. Kleijn, “A DNN Optimizer that Improves over AdaBelief by Suppression of the Adaptive Stepsize Range,” arXiv preprint arXiv:2203.13273, 2022.

[9] Mishra, A., Chaturvedi, R. P., Sharma, H., Sharma, R., & Asthana, S. (2023, November). Multi-Scale Optimized Feature Network for Polyp Segmentation. In 2023 International Conference on Computing, Communication, and Intelligent Systems (ICCCIS) (pp. 444-448). IEEE.

[10] Chaturvedi, R. P., & Ghose, U. (2023). An effective framework for detecting the object from the video sequences by utilizing deep learning with hybrid technology. Journal of Information and Optimization Sciences, 44(1), 113-126.

[11] Mishra, A., Gupta, P., & Tewari, P. (2022). Global U-net with amalgamation of inception model and improved kernel variation for MRI brain image segmentation. Multimedia Tools and Applications, 81(16), 23339-23354..

[12] Zhang, X., Zhang, Y., Shen, K., Fu, Q., & Shen, H. (2025). FAFNet: An Overhead Transmission Line Component Detection Method Based on Feature Alignment and Fusion. IEEE Sensors Journal.

[13] G. Yang, J. Lei, Z. Zhu, S. Cheng, Z. Feng, and R. Liang, “AFPN: Asymptotic Feature Pyramid Network for Object Detection,” arXiv:2306.15988, Jun. 2023.