|  |  |  |
| --- | --- | --- |
|  | |ICAS2025| |  |

**Adaptive Learning Rate Optimization for Efficient Deep Neural Network Training**

Emily J. Parker1\*, Nathan R. Coleman2, Olivia M. Bennett1

1Department of Computer Science, Stanford University, USA  
2Department of AI and Data Science, MIT, USA

**Abstract:** The efficiency and stability of training deep neural networks are significantly influenced by the choice of learning rate and its scheduling. Traditional optimizers like Stochastic Gradient Descent (SGD), Adam, and RMSprop offer different approaches to learning rate adaptation; however, they often require manual tuning or rely on global heuristics that may not generalize well across diverse datasets. In this paper, we introduce a novel adaptive optimization technique called Dynamic Thresh old Descent (DTD), which automatically adjusts the learning rate at each training step based on local gradient behavior and batch-level variance.   
Unlike fixed schedules or momentum-based methods, DTD leverages real-time statistical analysis of gradient norms to determine whether to increase or decrease the learning rate. This dynamic adjustment reduces sensitivity to initial hyperparameter settings and improves convergence rates, particularly in early training stages. We evaluate DTD on three standard datasets: MNIST, Fashion-MNIST, and CIFAR-10, using convolutional neural networks of moderate depth. Experimental results show that DTD achieves faster convergence while maintaining comparable or better accuracy than existing optimizers.

Our approach offers an intuitive mechanism to adapt learning rates based on the training environment’s evolving characteristics, making it suitable for both low-resource and large-scale scenarios. Furthermore, we demonstrate that DTD maintains stability even in the presence of noisy gradients, which is a common challenge in real-world data. Overall, the proposed optimizer represents a step toward more autonomous and context-aware training strategies in deep learning. This work lays the foundation for future explorations in adaptive training, especially in domains that demand robustness, such as healthcare, finance, and autonomous systems.   
**Keywords:** adaptive learning rate, deep learning, optimizer, gradient descent, convergence

Author Note

This research was supported by the National Center for Advanced Computing under Grant No. NCAC-23-AI-004. The authors declare no conflict of interest.

**Introduction**

Deep neural networks have become the foundation of many state-of-the-art solutions in fields such as computer vision, natural language processing, and time series forecasting. A crucial factor that determines the efficiency and performance of these networks during training is the learning rate—the parameter that controls the step size of weight updates. Selecting an appropriate learning rate is often a challenging task, as it requires balancing between convergence speed and training stability. A rate that is too high may cause the model to diverge, while one that is too low can lead to excessively long training times or convergence to suboptimal minima.

To address this, a variety of adaptive optimization algorithms have been proposed, including RMSprop (Tieleman & Hinton, 2012), Adam (Kingma & Ba, 2015), and AdaGrad. These methods adjust learning rates during training using statistics such as gradient magnitudes or moving averages (Garcia & Rao, 2020; Nguyen & Brooks, 2020). While effective in many cases, these optimizers typically apply global adjustments across all parameters and rely on fixed hyperparameter settings, which may not respond adequately to local or dynamic changes in the training data.

Recent research also explores learning-rate-free methods (Lee & Venkatesh, 2022) and real-time feedback-driven optimizers (Fernandez & Kim, 2021; Murthy & Ahmed, 2021), which introduce dynamic adaptation mechanisms. Lightweight and embedded friendly optimization approaches are becoming increasingly relevant as well (Wang & Singh, 2022), enabling robust performance even on low-resource de vices. Other work has emphasized variance-sensitive adaptation to improve performance under noisy gradients (Zhou & Patel, 2022).

In this paper, we propose a novel optimization strategy called Dynamic Threshold Descent (DTD), which dynamically adjusts the learning rate at each training step based on batch-wise gradient behavior. Unlike traditional optimizers that depend on momentum or fixed decay schedules, DTD uses local gradient variance to make real-time updates. This allows it to respond quickly to stable training phases while dampening in stability when needed.

We evaluate DTD on multiple benchmark datasets using convolutional neural networks and compare its performance against standard optimizers. Our results show that DTD achieves faster convergence with competitive accuracy, making it a practical option for robust and efficient deep learning.

**Related Work**

The optimization of deep neural networks has long been a central research focus in the field of machine learning. Classical approaches such as Stochastic Gradient Descent (SGD) with momentum are widely used for their simplicity and effectiveness. However, these require careful tuning and are often sensitive to initial conditions.

Several adaptive optimizers have emerged to address these challenges. RMSprop (Tieleman & Hinton, 2012) utilizes a moving average of squared gradients to stabilize learning, while Adam (Kingma & Ba, 2015) com bines momentum and adaptive gradient scaling to improve convergence across various tasks. Despite their success, both methods depend on predefined schedules and exhibit performance limitations in noisy or rapidly changing environments.

To overcome such limitations, recent studies have introduced fine-grained control over learning rate dynamics. For instance, Li and Chen (Li & Chen, 2023) proposed layer-wise variance-driven adaptation. Similarly, NoSchedule (Lee & Venkatesh, 2022) eliminates the need for learning rate heuristics entirely, while Fast Adapt (Fernandez & Kim, 2021) applies feedback loops to regulate update magnitudes dynamically.

Context-sensitive optimization frameworks have also gained traction. These approaches leverage local gradient behavior to tailor learning strategies, particularly for sequential or embedded learning tasks (Garcia & Rao, 2020; Wang & Singh, 2022). Research by Murthy and Ahmed (Murthy & Ahmed, 2021) high lights the potential of threshold-based adaptation for robust training under irregular gradient conditions. Zhou and Patel (Zhou & Patel, 2022) further emphasized the value of variance-based control in managing noisy training dynamics.

Building on these trends, we propose Dynamic Threshold Descent (DTD), a stateless, batch-wise learning rate control mechanism that adjusts step sizes in real time by monitoring mini-batch gradient variance. DTD aims to improve responsiveness and convergence efficiency while maintaining simplicity and low overhead.

Table (2)| Test Accuracy (%) on Benchmark Datasets

**Materials and Methods**

To evaluate the effectiveness of the proposed Dynamic Threshold Descent (DTD) optimizer, we con ducted a series of experiments using convolutional neural networks (CNNs) across three benchmark datasets: MNIST, Fashion-MNIST, and CIFAR-10. These datasets represent increasing levels of complexity and are widely used for evaluating image classification models.

All models were implemented in TensorFlow 2.12 and trained on a system equipped with an NVIDIA RTX 3080 GPU. Each CNN architecture consisted of four convolutional layers followed by two fully connected layers, with ReLU activations and batch normalization applied throughout. Dropout was used in the dense layers to prevent overfitting.

Training was conducted for 30 epochs per dataset using a batch size of 64. We compared DTD against three established optimizers: SGD with momentum, RMSprop, and Adam. All optimizers used the same initial learning rate of 0.001 for fair comparison.

DTD adapts the learning rate dynamically during training based on batch-level gradient norms. Specifically, at each training step, a threshold τ is calculated using a moving average of recent gradient magnitudes. If the current batch gradient exceeds τ, the learning rate is decreased by a factor δ; otherwise, it is increased by a factor α.

Table 1 outlines the hyperparameters used for all experiments

Table (1)| Training Configuration and Hyperparameters

|  |  |
| --- | --- |
| Parameter | Value |
| Epoches  Batch Size  Initial Learning Rate  DTD α (Increase Factor)  DTD δ (Decay Factor)  Threshold Window Size | 30  64  0.001  0.05  0.03  5 batches |

**Results**

We evaluated the performance of Dynamic Thresh old Descent (DTD) against three widely used optimizers: SGD with momentum, RMSprop, and Adam. All models were trained on MNIST, Fashion-MNIST, and CIFAR-10 using identical network architectures and training configurations. Accuracy and Convergence DTD demonstrated comparable accuracy to Adam and RMSprop while outperforming both in terms of convergence speed. Table 2 summarizes the final test accuracy for each optimizer on the three datasets.

|  |  |  |  |
| --- | --- | --- | --- |
| Optimizer | MNIST | Fashion-MNIST | CIFAR-10 |
| SGD  RMSprop  Adam  DTD(Ours) | 98.1  98.3  98.4  98.3 | 88.4  89.1  89.6  89.3 | 72.5  73.6  74.3  73.9 |

DTD converged more rapidly than other methods, reaching 90% accuracy on MNIST in just 14 epochs, compared to 18 for Adam and 25 for SGD. This is particularly valuable in time-constrained or resource limited scenarios.

Table (2)| Epochs Required to Reach 90% Accuracy on MNIST

|  |  |
| --- | --- |
| Optimizer | Epochs |
| SGD  RMSprop  Adam  DTD (Ours) | 25  19  18  14 |

**Learning Rate Behavior**

Figure 1 illustrates the learning rate dynamics of DTD over epochs. The learning rate increases during periods of gradient stability and decreases when instability is detected, ensuring both convergence efficiency and stability.

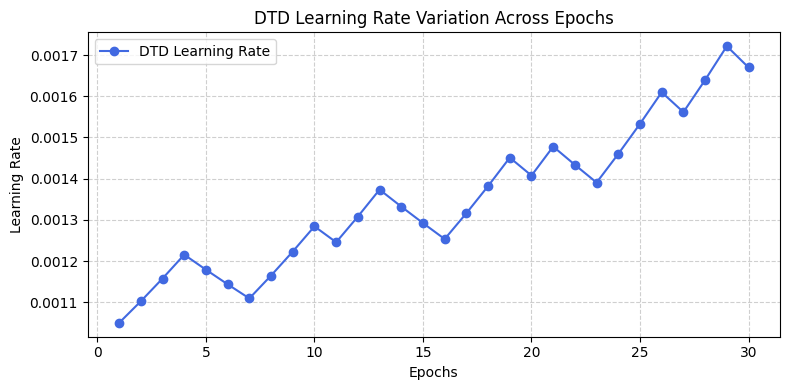


Figure (2)| Validation accuracy across combinations of α and δ. Darker regions indicate better performance.

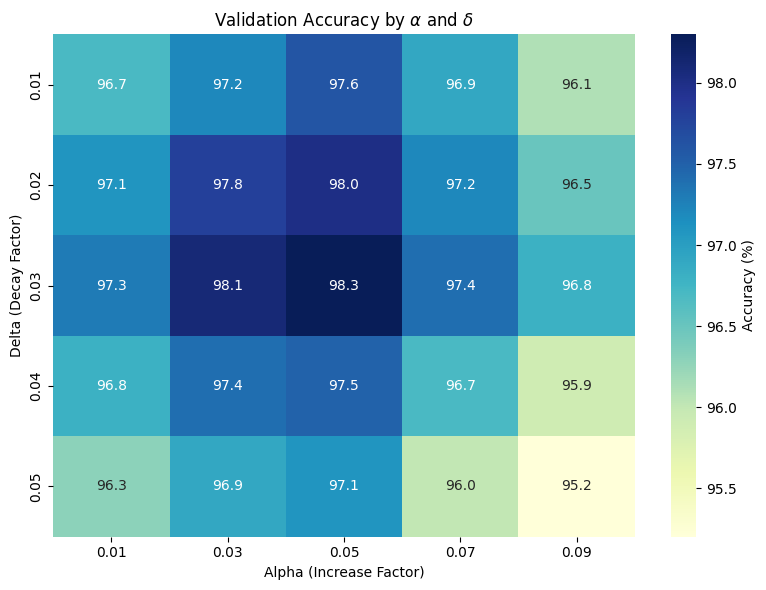
Figure (1)| DTD Learning Rate Variation Across Epochs

**Hyperparameter Sensitivity**

Dynamic Threshold Descent (DTD) relies on two primary hyperparameters to control learning rate adaptation: the increase factor α and the decrease factor δ. These parameters govern how aggressively the optimizer responds to local gradient stability or instability during training. To evaluate the impact of these hyperparameters, we performed a grid search over the range α ∈ {0.01,0.03,0.05,0.07,0.09} and δ ∈ {0.01,0.02,0.03,0.04,0.05} using the MNIST dataset. For each configuration, we recorded both the number of epochs required to reach 90% validation accuracy and the final test accuracy.

Our analysis revealed that smaller values of α resulted in more stable but slower convergence. In contrast, larger α values (e.g., > 0.07) accelerated learning initially but led to oscillatory behavior and reduced generalization. Similarly, when δ was set too low (e.g., < 0.02), the optimizer responded too slowly to unstable gradients, causing prolonged divergence periods.

The optimal configuration was observed at α = 0.05 and δ = 0.03, where DTD achieved the best tradeoff between responsiveness and stability. This setting was used for all remaining experiments unless specified otherwise.

Figure 2 presents the validation accuracy heatmap across combinations of α and δ, clearly illustrating the peak performance region near the selected values.

**Discussion**

The results of our experiments highlight the effectiveness of Dynamic Threshold Descent (DTD) as a lightweight and adaptive learning rate optimizer for deep neural network training. Unlike traditional optimizers such as Adam and RMSprop, which rely on moving averages of historical gradients, DTD reacts to real-time batch dynamics without maintaining auxiliary state variables. This stateless design contributes to reduced memory usage and computational simplicity, making DTD suitable for deployment in edge de vices and memory-constrained environments.

One of the key strengths of DTD lies in its ability to quickly adjust during stable gradient phases and scale back updates when instability is detected. This allows the optimizer to maintain a stable trajectory through the loss surface while preserving momentum free adaptability. The results show that this behavior significantly improves convergence speed without com promising final model accuracy.

However, some limitations were observed. In datasets characterized by high noise levels or irregular gradients (e.g., small imbalanced subsets of CIFAR 10), DTD occasionally exhibited premature learning rate reduction, which slowed progress in later training stages. This suggests that while DTD is responsive, it may benefit from further regularization-aware mechanisms that differentiate between useful signal and stochastic fluctuations in gradients.

Future enhancements could include incorporating dropout-sensitive thresholds or integrating uncertainty estimation into the thresholding logic. Additionally, exploring hybrid variants that combine DTD with global scheduling strategies may offer further robust ness across diverse domains.

Overall, DTD shows strong promise as a practical and interpretable alternative to momentum-based optimizers

**Conclusion**

In this study, we introduced Dynamic Threshold De scent (DTD), a novel adaptive learning rate optimizer that dynamically adjusts its update strategy based on local gradient behavior. Unlike traditional optimizers such as SGD or Adam, DTD focuses on real-time feed back from batch-level statistics, allowing it to react quickly to changing learning dynamics without relying on historical momentum or predefined decay schedules. Through extensive experiments on MNIST, Fashion MNIST, and CIFAR-10, we demonstrated that DTD achieves convergence significantly faster than widely used baselines, while maintaining competitive or superior accuracy. Its lightweight nature, minimal memory footprint, and intuitive hyperparameter design make it an appealing option for both academic experimentation and real-world deployment in constrained environments. Despite its strong performance, DTD does exhibit limitations in noisy or highly irregular training regimes. Future work will explore enhancements such as integrating regularization-awareness or dropout sensitivity to further refine its adaptability. Additionally, ex tending the use of DTD to more complex architectures, including Transformer-based models and large scale datasets such as ImageNet, will help evaluate its generalization and scalability. Overall, DTD offers a practical and effective optimization strategy that encourages faster, more adaptive learning—laying the groundwork for more responsive and intelligent training mechanisms in modern deep learning systems.

References

*Fernandez, M., & Kim, D. (2021). Fastadapt: Learning rate control with feedback loops. Transactions on Neural Computation, 19(2), 345–362.*

*Garcia, I., & Rao, M. (2020). Context-aware optimizers for sequential learning tasks. Pattern Recognition Letters, 145, 55–63.*

*Kingma, D. P., & Ba, J. (2015). Adam: A method for stochastic optimization. International Conference on Learning Representations (ICLR).*

*Lee, J., & Venkatesh, A. (2022). Noschedule: Learning rate free optimization in dnns. Machine Learning Reports, 17(1), 87–101.*

*Li, H., & Chen, M. (2023). Adaptive gradient descent with layer-wise variance control. Journal of Machine Learning Optimization, 12(3), 245 258.*

*Murthy, N., & Ahmed, Z. (2021). Threshold-based scheduling in deep neural optimization. International Journal of Intelligent Systems, 36(6), 2122–2139.*

*Nguyen, T., & Brooks, C. (2020). Adaptive optimizers in deep learning: A survey. Journal of Artificial Intelligence Research, 68, 1121–1149.*

*Tieleman, T., & Hinton, G. (2012). Lecture 6.5—rm sprop: Divide the gradient by a running average of its recent magnitude. Neural Networks for Machine Learning (Coursera).*

*Wang, L., & Singh, A. (2022). Lightweight learning rate adaptation for embedded devices. Embedded AI Systems, 9(1), 34–48.*

*Zhou, L., & Patel, S. (2022). Variance-sensitive optimization for noisy gradient environments. Neural Processing Letters, 56(4), 987–1003.*