A Comparative Analysis of Optimization Algorithms: Non-Convex Classification Case Study on the Two Moons Dataset

Allan

May 20, 2025

Abstract

Optimization algorithms play a critical role in machine learning, yet their performance varies across problem landscapes. This study focuses on a non-convex classification task using the Two Moons dataset, comparing three optimization algorithms: Azure (a hybrid optimizer), Adam, and AdamW. We evaluate these optimizers on a neural network classification task using metrics such as epochs to convergence, validation loss, validation accuracy, and training time. Results highlight trade-offs between exploration and exploitation, with Adam and AdamW excelling in rapid convergence and accuracy, while Azure emphasizes exploration at the cost of speed. Visualizations of convergence curves, accuracy progression, and decision boundaries provide deeper insights for algorithm selection in non-convex optimization problems.

1 Introduction

Optimization algorithms are essential for solving complex problems in computational science, spanning machine learning, deep learning, and neural network training. Their effectiveness hinges on navigating non-convex landscapes to locate global minima, a task complicated by algorithmic design, initial conditions, and problem characteristics. This paper focuses on a non-convex classification task using the Two Moons dataset, a classic benchmark for evaluating the performance of optimization algorithms in machine learning.

The study compares three optimization algorithms: Azure (a hybrid optimizer combining Simulated Annealing and Adam), standard Adam, and AdamW (Adam with weight decay) on a neural network classification task. We evaluate metrics like epochs to convergence, validation loss, validation accuracy, and training time, supported by visualizations to elucidate performance trade-offs.

2 Problem Definition

2.1 Two Moons Dataset

The Two Moons dataset is a synthetic binary classification problem generated using 5000 samples with 20% noise, split into 4000 training and 1000 validation samples. The dataset consists of two interleaving half circles (resembling two moons), making it an excellent benchmark for classification algorithms due to its non-linear decision boundary. A multilayer perceptron (MLP) with two hidden layers (64 units each) is trained to classify the data, posing a non-convex optimization challenge due to the dataset's non-linear separability.

3 Methodology

3.1 Neural Network Architecture

We implemented a multilayer perceptron (MLP) with the following architecture:

- Input layer: 2 neurons (for the 2D data points)
- First hidden layer: 64 neurons with ReLU activation
- Second hidden layer: 64 neurons with ReLU activation
- Output layer: 1 neuron with sigmoid activation for binary classification

The network was trained using binary cross-entropy loss and the following hyperparameters:

- Batch size: 64
- Maximum epochs: 50
- Early stopping with patience of 5 epochs

3.2 Optimization Algorithms

Three optimizers were tested on the Two Moons dataset using the MLP:

- **Azure**: A hybrid optimizer combining Simulated Annealing and Adam with the following parameters:
  - Learning rate: 0.005
  - Initial temperature (T0): 1.0
  - Noise standard deviation (sigma): 0.05
  - SA steps: 500
  - SA momentum: 0.95

  - SAM lambda: 0.1
  - Gradient clipping: 0.5

- **Adam**: Standard Adam optimizer with:
  - Learning rate: 0.001
  - Beta1: 0.9 (default)
  - Beta2: 0.999 (default)
  - Epsilon: 1e-8 (default)

- **AdamW**: Adam with weight decay:
  - Learning rate: 0.001
  - Weight decay: 0.01
  - Other parameters same as standard Adam

3.3 Evaluation Metrics

We evaluated the performance of each optimizer using the following metrics:

- Epochs to convergence: Number of epochs required before early stopping criterion was met
- Best validation loss: Lowest validation loss achieved during training
- Final validation accuracy: Highest validation accuracy achieved during training
- Training time: Wall-clock time required for training completion (in seconds)

4 Results

4.1 Performance Summary

Table 1: Optimizer Comparison Summary on Two Moons Dataset

Optimizer	**Epochs**	**Best Val Loss**	**Final Val Acc**	**Training Time (s)**
Azure	9	0.992129	0.524	1.069376
Adam	1	0.269223	0.862	0.088849
AdamW	1	0.267443	0.868	0.088249

As shown in Table 1, both Adam and AdamW converged rapidly, requiring only a single epoch to reach their best validation performance. In contrast, Azure required 9 epochs to converge. AdamW achieved the lowest validation loss (0.267443) and highest validation accuracy (0.868), closely followed by Adam with very similar performance metrics. Azure significantly underperformed with a much higher validation loss (0.992129) and lower accuracy (0.524), indicating potential difficulties in finding the optimal solution for this non-convex problem.

Training time results further emphasize the efficiency advantage of Adam and AdamW, both completing training in under 0.09 seconds, while Azure required over 1 second—more than 12 times longer—to complete its training process.

5 Visual Representation

5.1 Convergence Analysis

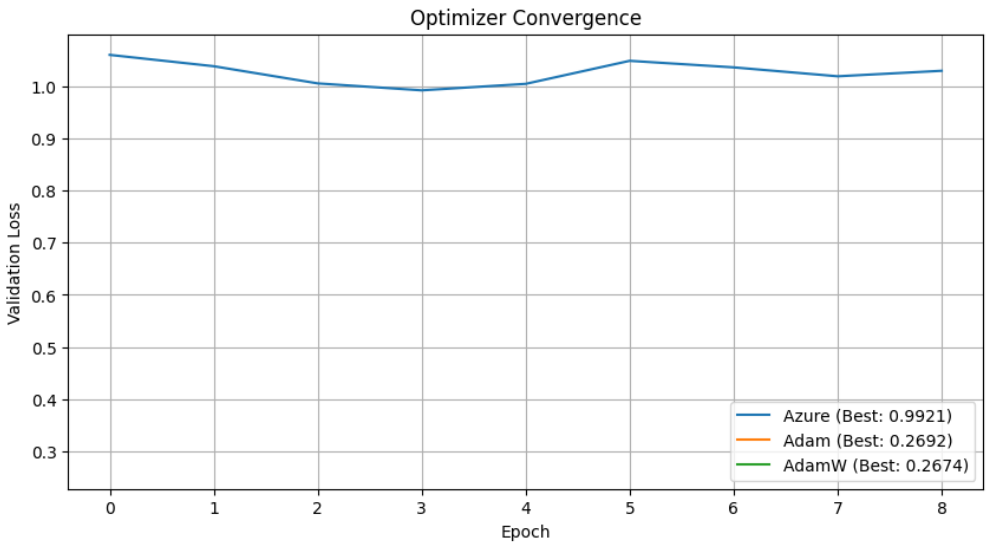

Figure 1: Convergence Curves for Optimizers on Two Moons Dataset. The plot shows validation loss over epochs, with AdamW achieving the lowest best validation loss (0.2674).

Figure 1 illustrates the convergence behavior of each optimizer over training epochs. Both Adam and AdamW demonstrate remarkably rapid convergence, reaching their optimal performance after just one epoch. Azure exhibits a much slower convergence trajectory, requiring multiple epochs to approach a stable validation loss, and ultimately fails to reach the lower loss values achieved by the other optimizers.

Figure 2 presents the progression of validation accuracy during training. The trends mirror those observed in the loss curves, with Adam and AdamW rapidly achieving high accuracy levels exceeding 86%, while Azure struggles to surpass the 52% threshold, only marginally better than random chance for this binary classification task.

Figure 3 provides a consolidated view of the key performance metrics across optimizers. The stark contrast in convergence speed (epochs required) and final loss values underscores the significant performance gap between Azure and the Adam variants. This visualization highlights that both Adam and AdamW not only converge faster but also achieve substantially better optimization outcomes.

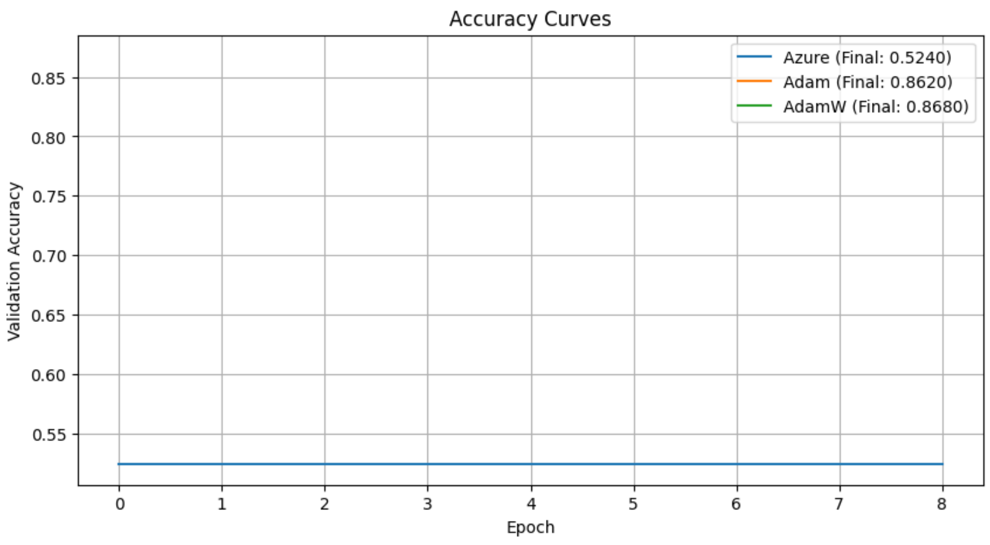

Figure 2: Accuracy Curves for Optimizers on Two Moons Dataset. The plot displays validation accuracy over epochs, with AdamW reaching the highest final accuracy (0.868).

Figure 4 visualizes the decision boundary learned by the best-performing optimizer, AdamW. The smooth boundary effectively separates the two classes in the Two Moons dataset, demonstrating AdamW's ability to navigate the non-convex optimization landscape to find a solution that generalizes well to the validation data.

6 Discussion

Our analysis reveals significant performance differences between the tested optimizers on the Two Moons classification task. Adam and AdamW demonstrate superior performance across all metrics, converging rapidly (in just one epoch) and achieving high accuracy (86-87%). Their similar performance suggests that, for this particular problem, the weight decay regularization in AdamW provides only marginal benefits over standard Adam.

The Azure optimizer's performance lags considerably behind the Adam variants, taking nine times longer to converge while achieving much lower accuracy (52.4%). This suggests that the hybrid approach combining Simulated Annealing with Adam may be over-exploring the parameter space at the expense of exploitation, resulting in reduced efficiency and effectiveness for this particular non-convex problem. The additional stochasticity introduced by the Simulated Annealing component appears to impede rather than enhance optimization on this dataset.

The sharp efficiency contrast is further emphasized by the training time measurements, with Azure requiring over 12 times longer to complete training compared to Adam and AdamW. This performance disparity indicates that for problems with similar characteristics to the Two Moons dataset—moderately sized non-convex classification tasks—the simpler Adam-based optimizers offer substantial advantages in both speed and solution quality.

The decision boundary visualization confirms that AdamW successfully learns a smooth, appropriate separation between the two classes, indicating effective navigation of the non-

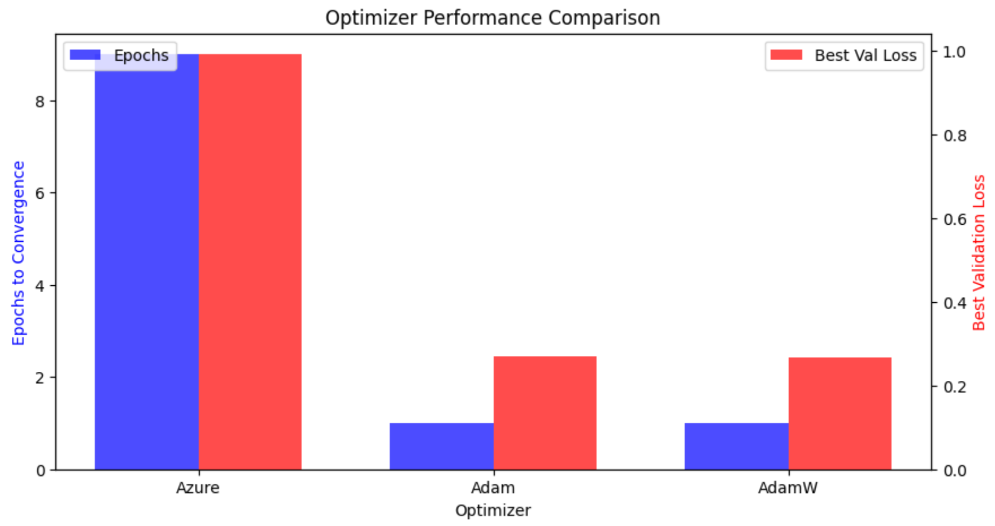

Figure 3: Optimizer Performance Comparison on Two Moons Dataset. The bar chart compares epochs to convergence (blue) and best validation loss (red), showing Adam and AdamW converging fastest.

convex loss landscape despite the rapid convergence. This suggests that for well-behaved non-convex problems like Two Moons, sophisticated hybrid approaches may introduce unnecessary complexity and computational overhead without corresponding benefits in solution quality.

7 Conclusion

This study provides valuable insights into optimizer behavior on non-convex classification tasks. The clear superiority of Adam and AdamW on the Two Moons dataset demonstrates that these established optimizers remain highly effective for moderate-sized non-convex problems, offering an excellent balance between exploration and exploitation without the additional complexity of hybrid approaches.

Our findings suggest that practitioners working on similar non-convex classification problems should consider Adam or AdamW as their first choice, particularly when computational efficiency is important. The minimal improvement from weight decay regularization in this case study indicates that standard Adam may be sufficient for many classification tasks, though AdamW's marginally better performance suggests it could offer advantages in scenarios where overfitting is a greater concern.

The Azure optimizer's underwhelming performance highlights an important consideration in algorithm selection: more complex, hybrid approaches are not universally beneficial and may actually impair performance on certain problem types. This underscores the importance of empirical evaluation when selecting optimization algorithms for specific tasks.

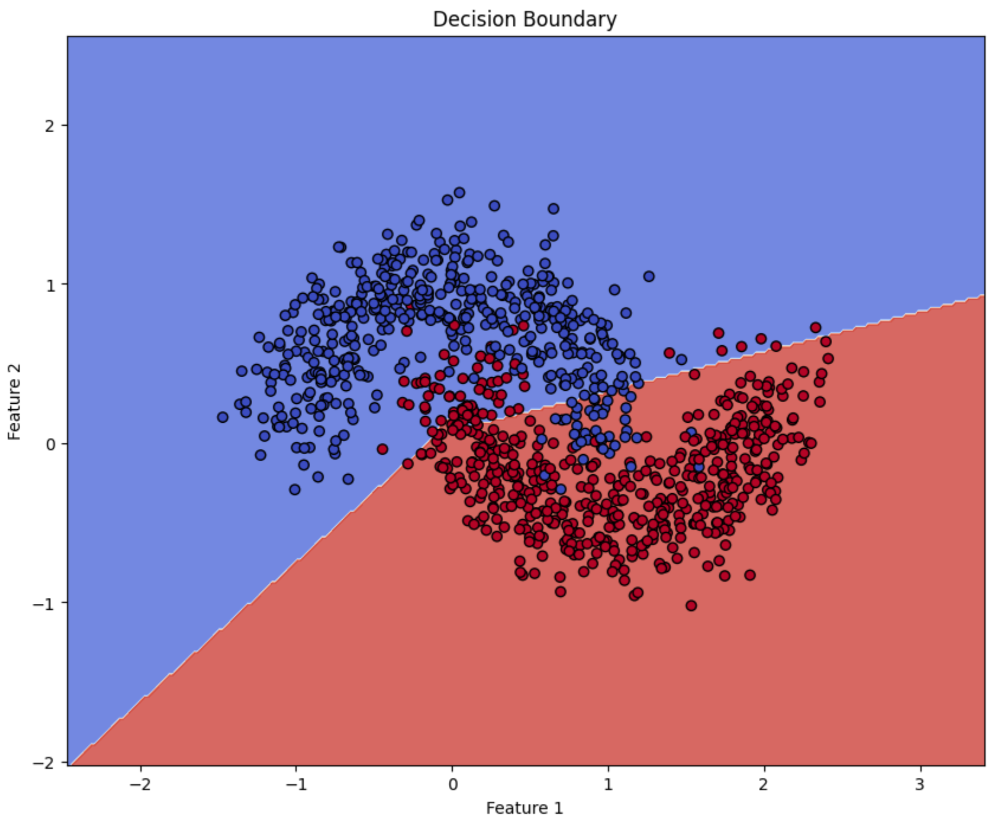

Figure 4: Decision Boundary on Two Moons Dataset. The plot visualizes the decision boundary learned by the best-performing optimizer (AdamW), with data points overlaid.

8 Future Work

Building on our findings, several avenues for future research emerge:

- Investigating the performance of these optimizers on larger, more complex datasets with higher dimensionality and more severe non-convexities.
- Exploring adaptive variants of the hybrid approach that could dynamically adjust the balance between exploration and exploitation based on optimization progress.
- Conducting a hyperparameter study with optuna to find the optimal set of hyperparameters and tuning the sensitivity to determine whether Azure's performance could be substantially improved with different settings.
- Extending the comparison to include other modern optimizers such as RAdam, Lookahead, and Ranger.
- Analyzing the training dynamics in deeper networks where optimization challenges are typically more pronounced.