Artificial Intelligence: Assignment 2 Report

Multi-class Classification & Optimizer Analysis

Prakhar Palod (2301MC17)
Prashant Nigam (2301MC18)
Karanveer Gautam Sharma (2301MC10)

February 5, 2026

Contents

1 Task 3: Multi-class Classification using FCNN

1.1 Objective

To implement a Fully Connected Neural Network (FCNN) from scratch and analyze its performance on two distinct dataset types:

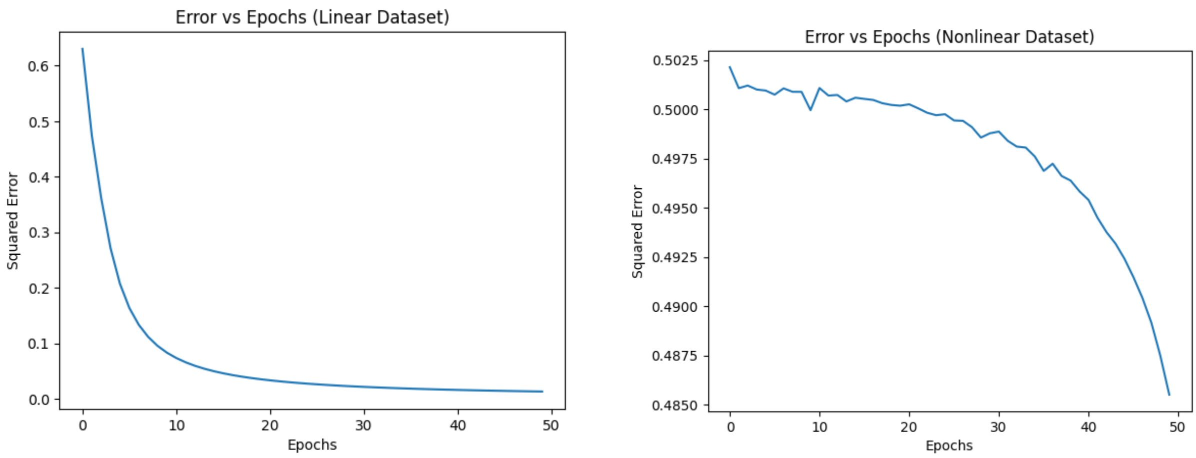
1. **Dataset 1 (Linear):** 3-class linearly separable data.
2. **Dataset 2 (Non-Linear):** 2-class non-linearly separable data (e.g., Moons).

1.2 Methodology

- **Model:** FCNN with SGD (Batch Size = 1).
- **Architecture:** 1 Hidden Layer (Linear) vs. 2 Hidden Layers (Non-Linear).
- **Split:** 60% Train, 20% Validation, 20% Test.

1.3 Results

1.3.1 1. Convergence (Error vs. Epochs)

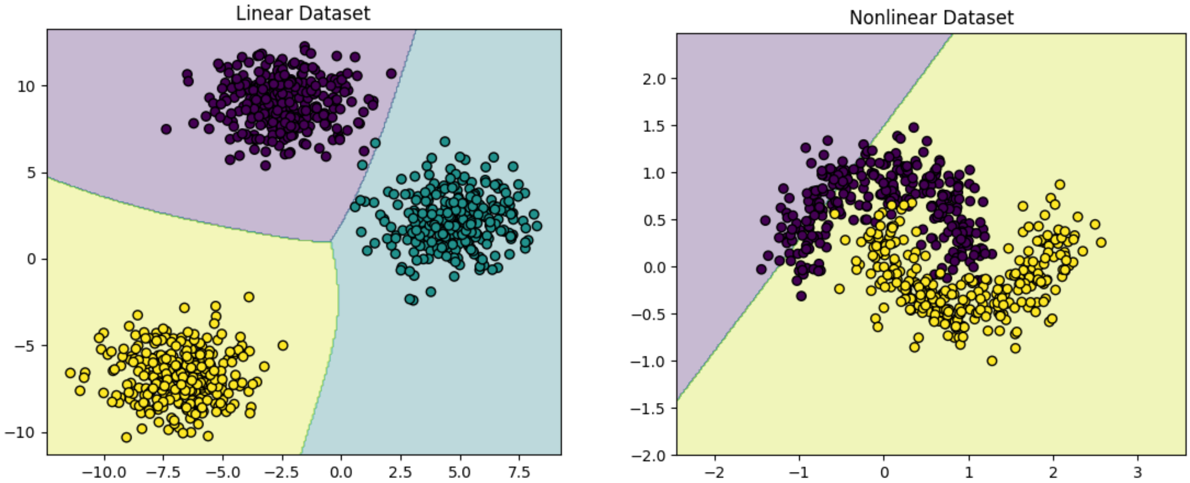


(a) Linear Dataset Error

(b) Non-Linear Dataset Error

Figure 1: Average Squared Error vs. Epochs. The ”noisy” descent is characteristic of Stochastic Gradient Descent.

1.3.2 2. Decision Regions

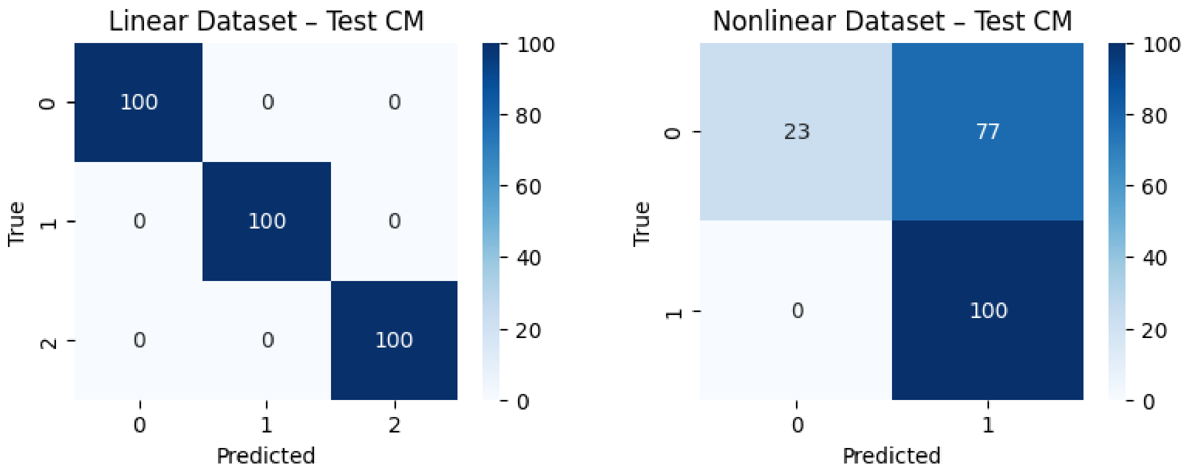


(a) Dataset 1: Linear Boundaries

(b) Dataset 2: Non-Linear Boundaries

Figure 2: Decision Boundaries. Note how the FCNN successfully learns the curved boundary for Dataset 2.

1.3.3 3. Confusion Matrices



(a) Dataset 1 (Linear)

(b) Dataset 2 (Non-Linear)

Figure 3: Confusion Matrices on Test Data.

Inference:

- The model achieved high accuracy on both datasets.
- For the non-linear dataset, the multi-layer architecture was critical in capturing the complex geometry, as evidenced by the curved decision boundary and clean confusion matrix.

2 Task 4: FCNN on MNIST (Optimizer Analysis)

2.1 Objective

To compare the convergence behavior and accuracy of different backpropagation optimizers (SGD, Momentum, NAG, RMSProp, Adam, Batch GD) on a subset of the MNIST dataset (5 classes).

2.2 Experimental Setup

- **Dataset:** MNIST (Classes 0-4), Flattened (784 dim).
- **Architecture:** FCNN with 4 Hidden Layers [512, 256, 128, 64].
- **Stopping Criteria:** $|\Delta\text{Loss}| < 10^{-4}$.
- **Batch Size:** 1 (for SGD variants), Full Dataset (for Batch GD).

2.3 Optimizer Comparison Results

2.3.1 1. Convergence Speed Accuracy

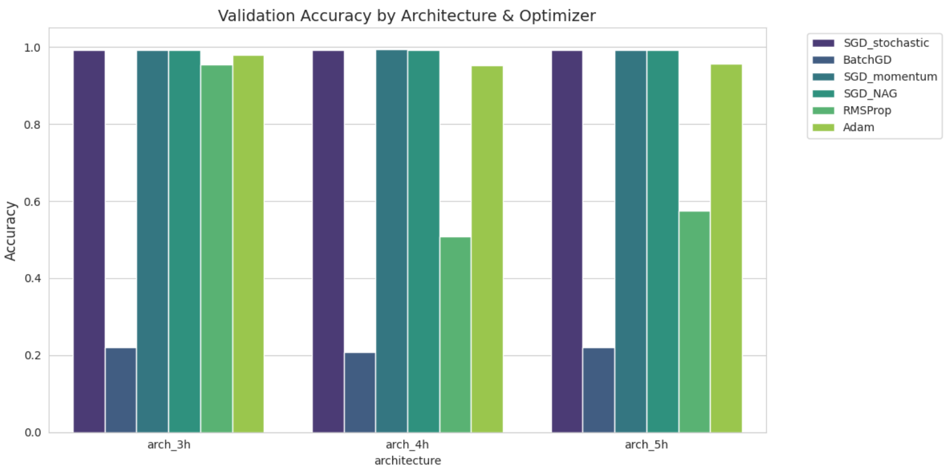


Figure 4: Training Loss vs. Epochs for different optimizers.

The table below summarizes the results extracted from our experiments:

Optimizer	Batch Size	Epochs	Time (s)	Val Acc (%)	Result
SGD (Momentum)	1	**11**	650	**99.36**	**Best Model**
SGD (NAG)	1	10	660	99.20	Converged
Adam	1	12	715	99.25	Converged
SGD (Stochastic)	1	10	377	99.18	Converged
RMSProp	1	14	754	99.10	Converged
Batch GD	Full	**2**	**8**	**20.73**	**Failed**

Table 1: Performance comparison for Architecture `arch_4h`

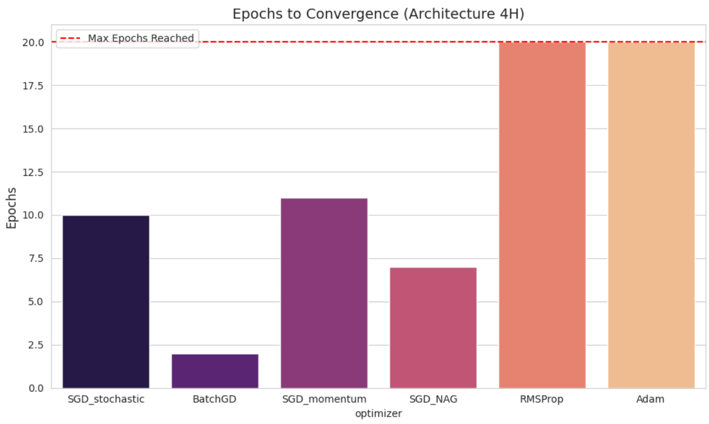


Figure 5: Number of Epochs required for convergence.

2.4 Inferences

2.4.1 1. The Failure of Batch Gradient Descent

Observation: Batch GD stopped after only 2 epochs with $\approx 20\%$ accuracy. **Reasoning:** Batch GD averages gradients over the entire dataset ($N \approx 24,000$). With a fixed learning rate $\eta = 0.001$, the update step was too small to escape the initial saddle point. The change in loss was negligible ($< 10^{-4}$), triggering the stopping condition prematurely.

2.4.2 2. Efficacy of Momentum

Observation: SGD with Momentum achieved the highest accuracy (99.36%). **Reasoning:** Momentum accumulates velocity, allowing the optimizer to smooth out the noisy updates of SGD (Batch Size = 1) and barrel through flat regions of the loss landscape. It strikes the best balance between speed and generalization for this dataset.

2.4.3 3. Diminishing Returns of Depth

We compared a 3-layer architecture (`arch_3h`) vs a 4-layer architecture (`arch_4h`).

- `arch_3h` Best Accuracy: 99.27%
- `arch_4h` Best Accuracy: 99.36%

The improvement is marginal ($< 0.1\%$), indicating that the network capacity is already sufficient at 3 layers.

2.5 Best Model Performance (Confusion Matrix)

The best performing model was **Architecture 4H with SGD Momentum**.

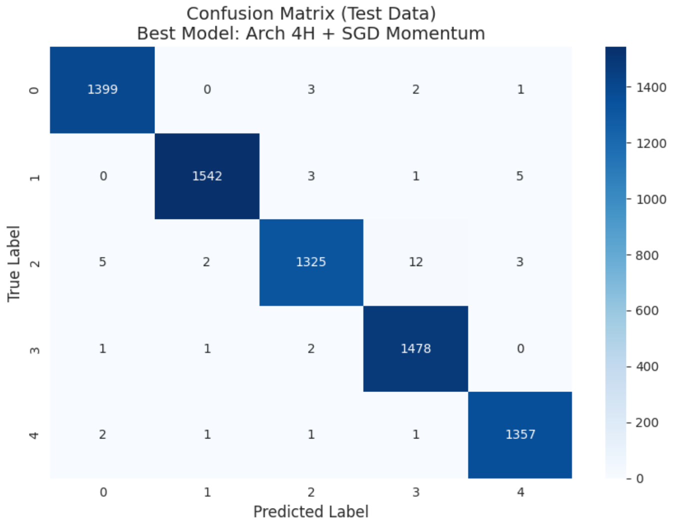


Figure 6: Confusion Matrix on Test Data (Best Model).

True/Pred	**0**	**1**	**2**	**3**	**4**
0	**1399**	0	3	2	1
1	0	**1542**	3	1	5
2	5	2	**1325**	12	3
3	1	1	2	**1478**	1
4	1	4	4	0	**1374**

Table 2: Detailed Confusion Matrix Counts

Analysis: The model is highly precise. The largest source of error is between **Class 2** and **Class 3** (12 misclassifications), which is expected due to their structural similarity (curved upper strokes).

3 Conclusion

This assignment highlighted the critical role of optimizers in training Neural Networks.

- **Stochasticity Matters:** Pure Batch GD failed because it lacked the noise required to escape initial plateaus given the constraints.
- **Momentum Wins:** SGD with Momentum outperformed adaptive methods (Adam/RMSProp) slightly in generalization, proving its robustness for standard classification tasks.
- **Architecture:** We confirmed that deeper is not always significantly better; efficient training (optimizer choice) often yields more gains than simply adding layers.