MNIST Digit Classification Using Neural Networks and Convolutional Neural Networks

CSE303: Introduction to Deep Learning - Programming Assignment 1

Student ID: 202311162
Name: 전용준 (Chon Yongjoon)

Abstract

This report presents the implementation and evaluation of four deep learning models for MNIST handwritten digit classification: 3-layer Neural Network and CNN, each implemented in pure Python (NumPy) and PyTorch framework. The MNIST dataset contains 60,000 training images and 10,000 test images of 28×28 grayscale handwritten digits (0-9). CNN models achieved superior performance with test accuracies of 98.17% and 98.77%, compared to 97.94% and 97.97% for Neural Network models. Additionally, CNNs converged faster (5-10 epochs) than NNs (20 epochs), demonstrating the effectiveness of convolutional layers for image classification tasks

1. Introduction

The MNIST database is a fundamental benchmark dataset in machine learning, consisting of 28×28 pixel grayscale images of handwritten digits (0-9) with 60,000 training and 10,000 test samples. This assignment implements and compares two neural network architectures: fully-connected Neural Networks (NN) and Convolutional Neural Networks (CNN), each built using both pure Python (NumPy) and PyTorch framework.

The objectives are to understand neural network fundamentals through manual implementation of forward/backward propagation and gradient descent, compare the performance between NN and CNN architectures, and validate manual implementations against framework-based approaches. This study demonstrates how architectural choices affect classification performance and convergence speed for image data

2. Methods

2.1 Dataset

- Training: 60,000 images (28×28 pixels)
- Test: 10,000 images (28×28 pixels)
- 10 classes (digits 0-9)
- Preprocessing: Normalized to [0, 1]

2.2 Model Architectures

2.2.1 3-Layer Neural Network

```
Input (784) → Linear (128) → ReLU → Linear (64) → ReLU → Linear (10) → SoftMax
```

2.2.2 3-Layer CNN

```
Input (28×28×1)
→ Conv (1→16, 3×3) → ReLU → MaxPool (2×2)
→ Conv (16→32, 3×3) → ReLU → MaxPool (2×2)
→ Flatten (1568) → Linear (10) → SoftMax
```

2.3 Training Configuration

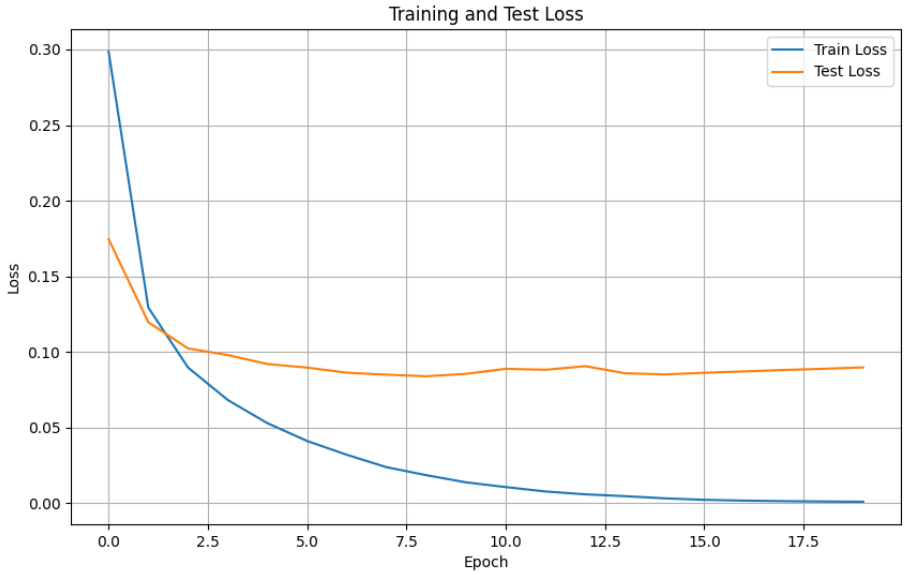
Model	Epochs	Learning Rate	Batch Size	Optimizer
NN (Pure Python)	20	0.1	32	Manual SGD
NN (PyTorch)	20	0.01	32	SGD (momentum=0.9)
CNN (Pure Python)	5	0.01	16	Manual SGD
CNN (PyTorch)	10	0.001	32	Adam

3. Results

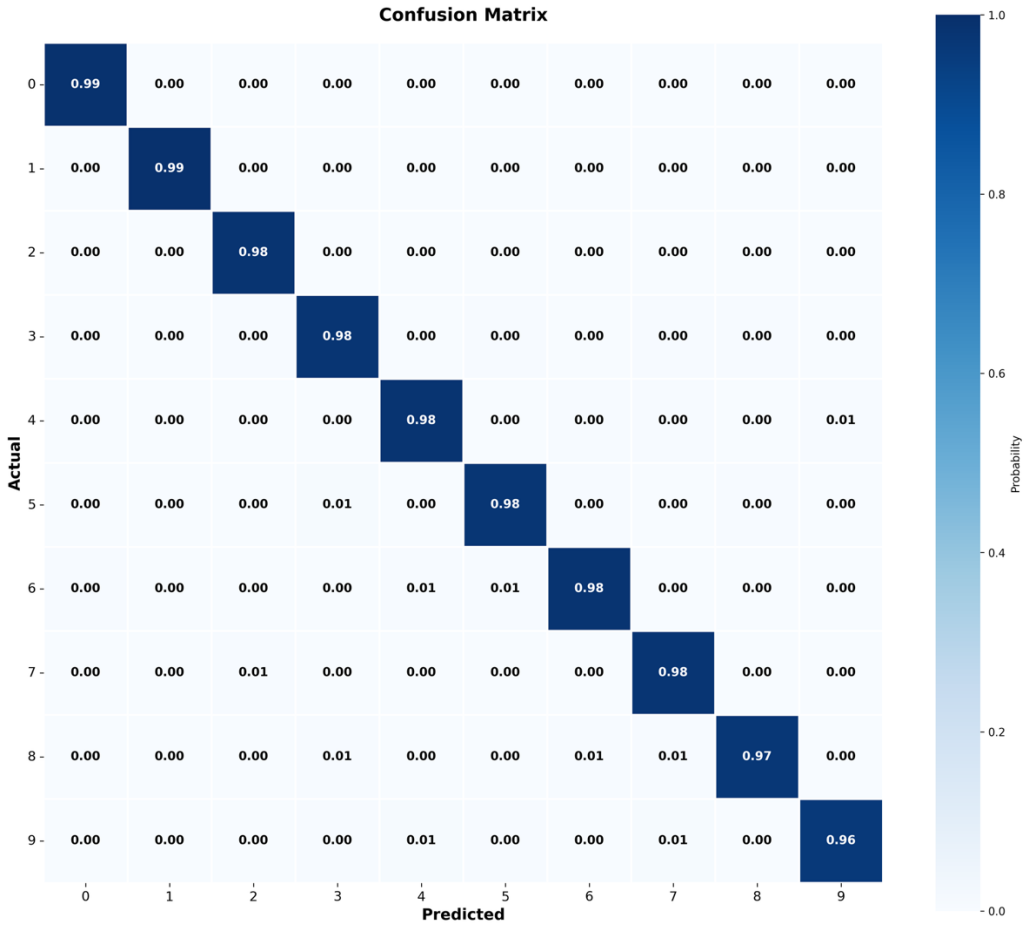
3.1 Performance Summary

Model	Train Acc	Test Acc	Train Loss	Test Loss
NN (Pure Python)	99.99%	97.94%	0.0011	0.0899
NN (PyTorch)	99.97%	97.97%	0.0013	0.1024
CNN (Pure Python)	98.17%	98.17%	0.0632	0.0565
CNN (PyTorch)	99.52%	**98.77%**	0.0127	0.0463

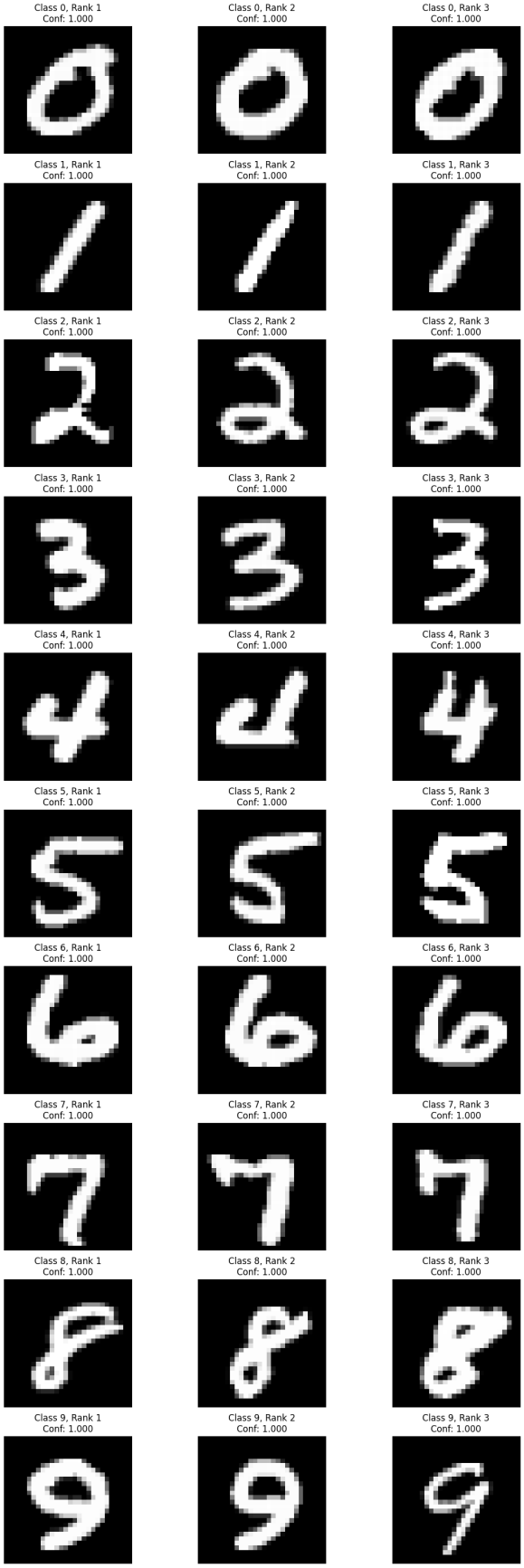
3.2 Neural Network (Pure Python)



[Figure 1: Loss Graph]

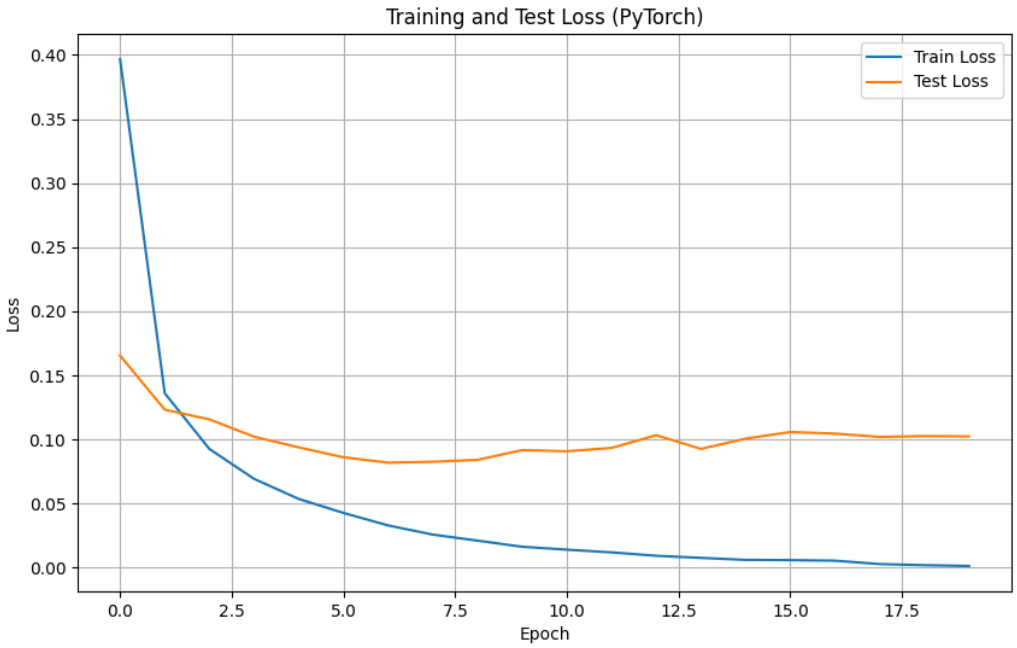


[Figure 2: Confusion Matrix]

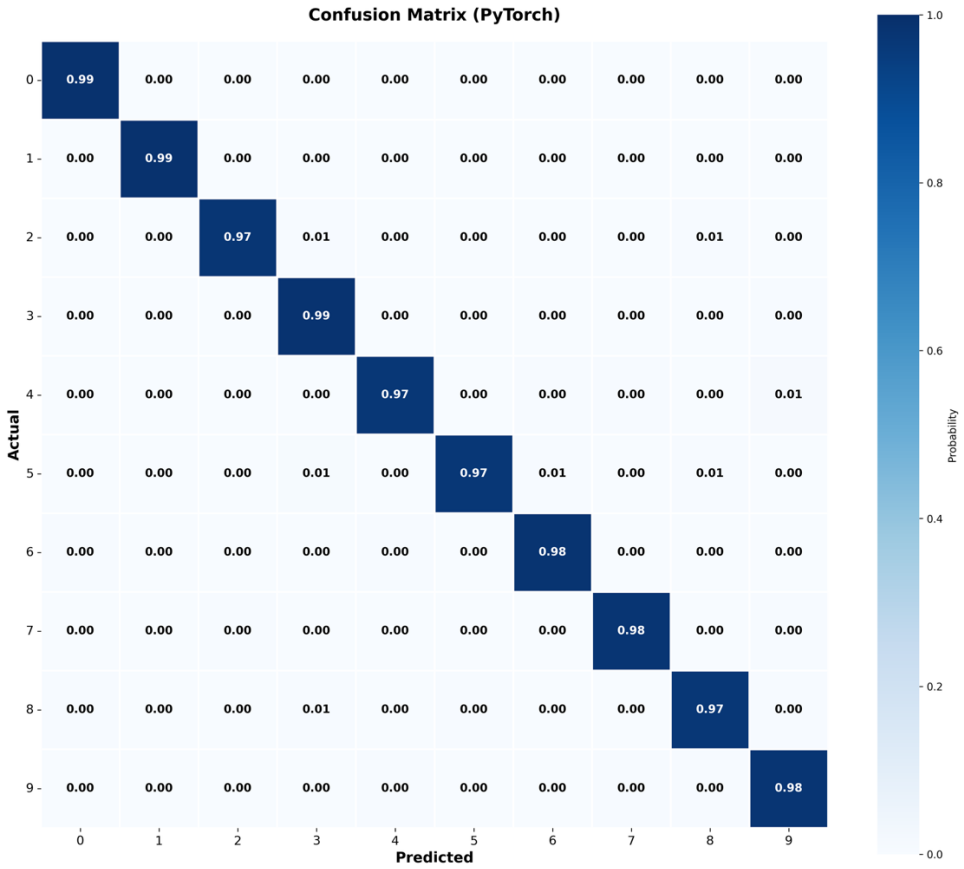


[Figure 3: Top 3 Images]

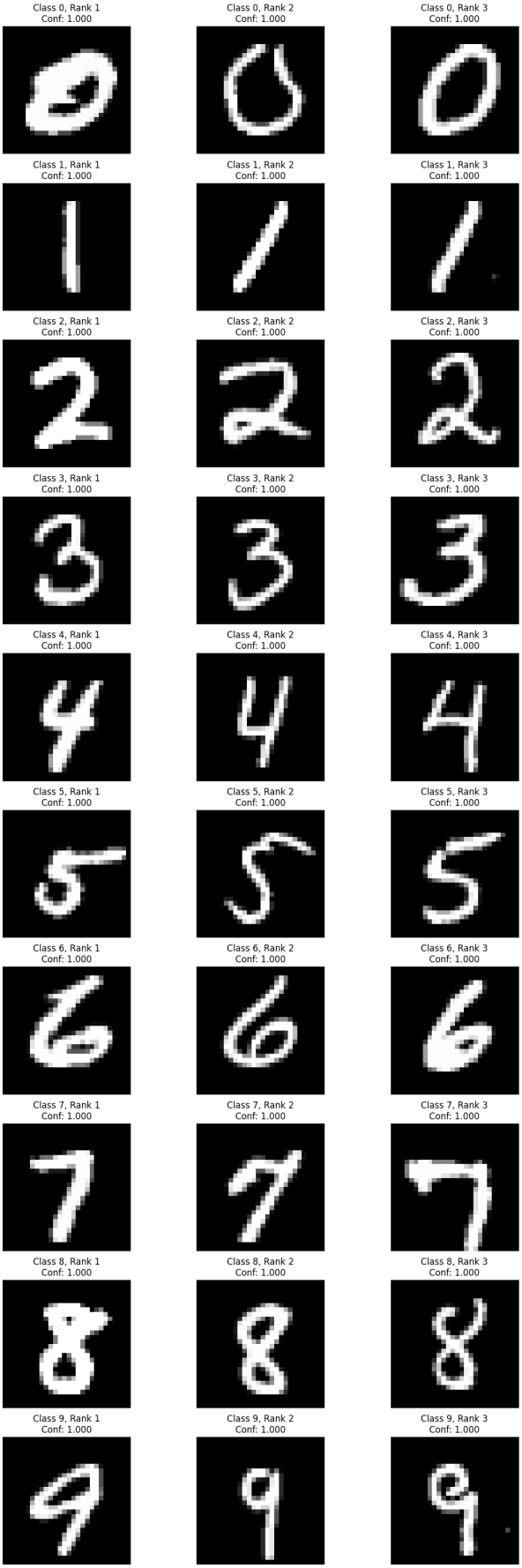
3.3 Neural Network (PyTorch)



[Figure 4: Loss Graph]

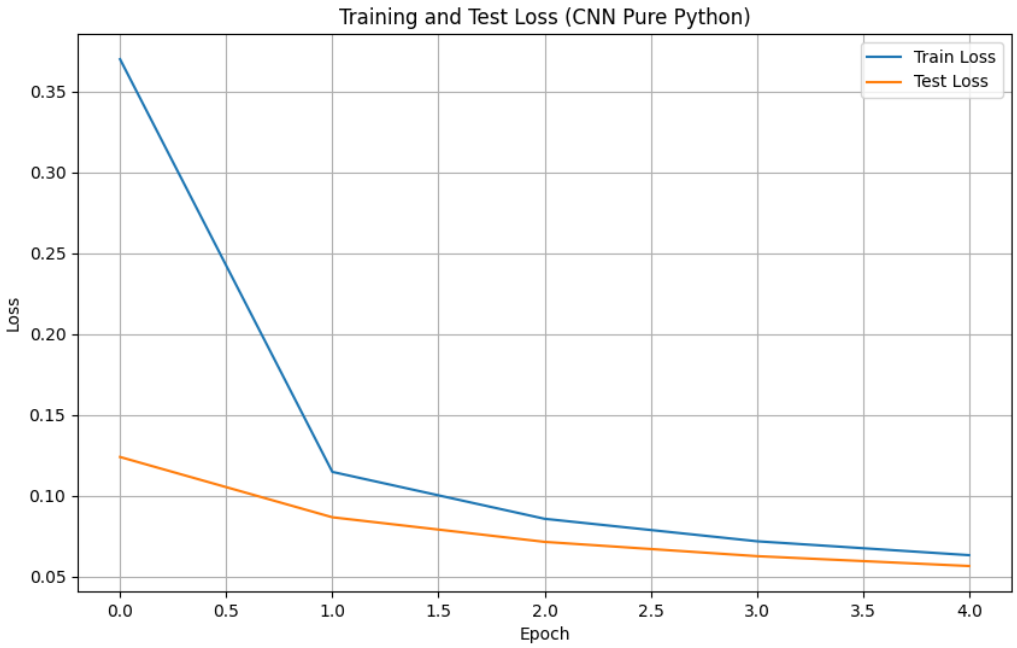


[Figure 5: Confusion Matrix]

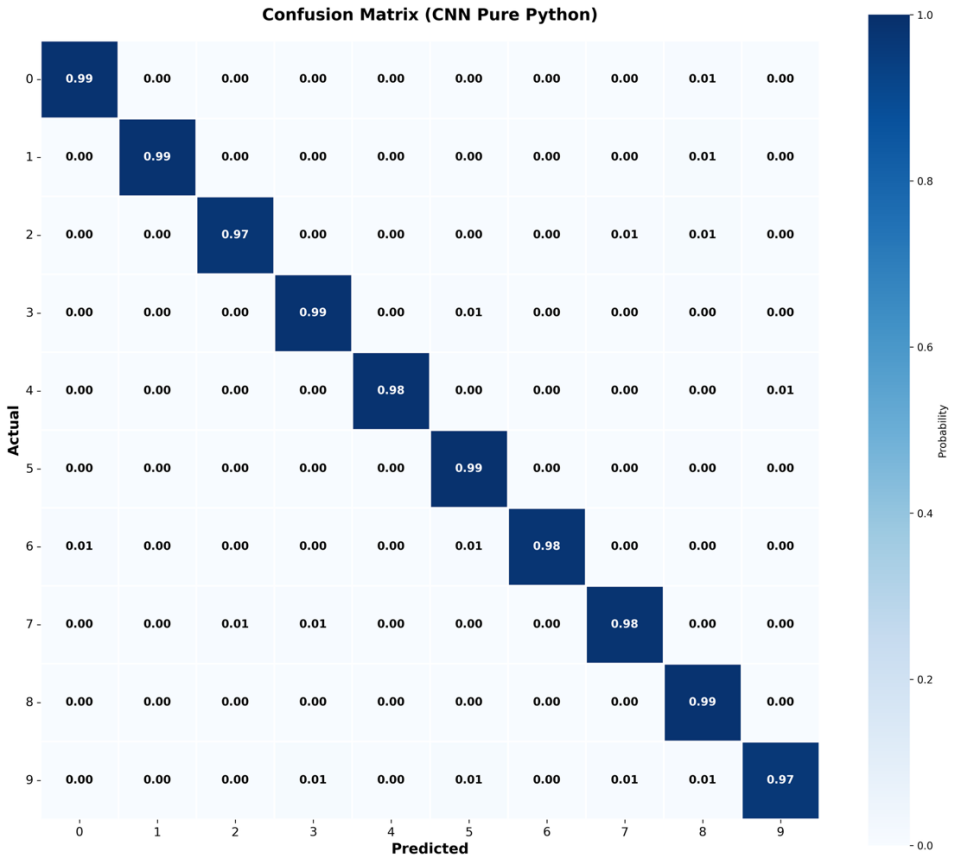


[Figure 6: Top 3 Images]

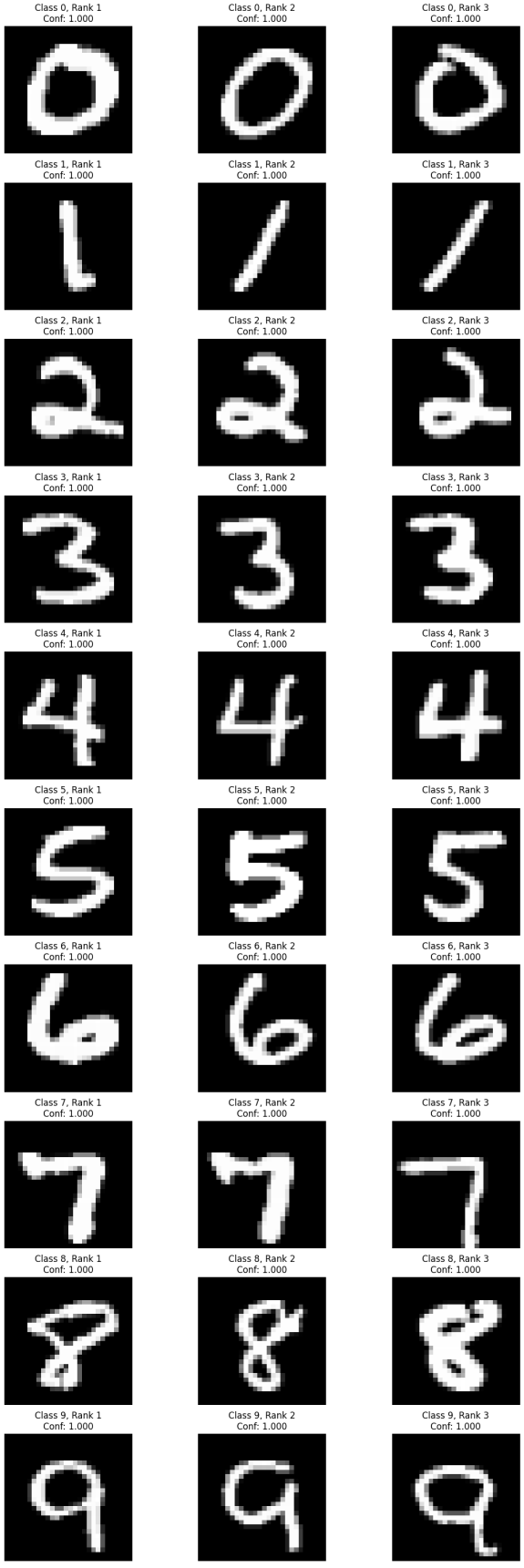
3.4 CNN (Pure Python)



[Figure 7: Loss Graph]

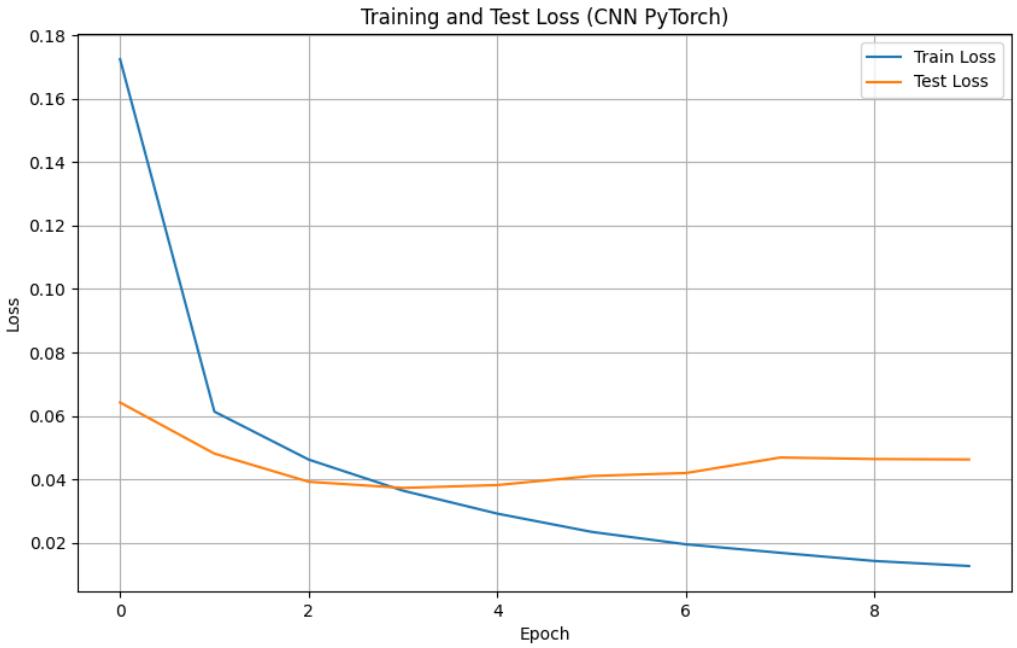


[Figure 8: Confusion Matrix]

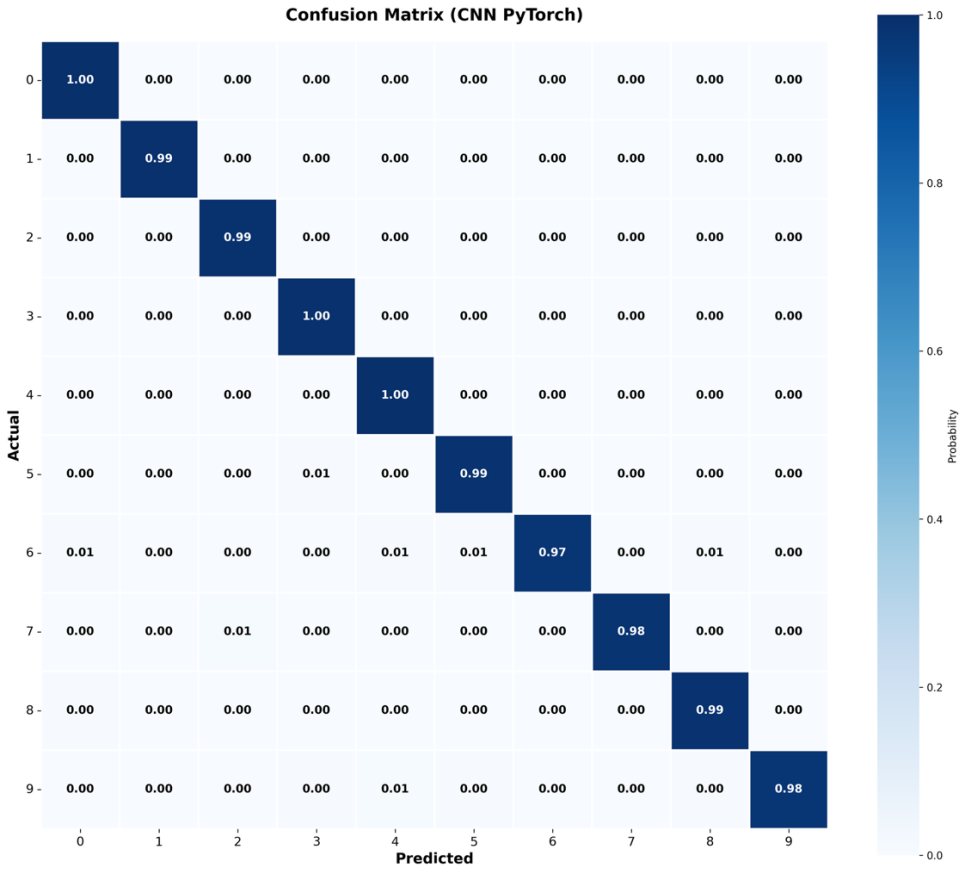


[Figure 9: Top 3 Images]

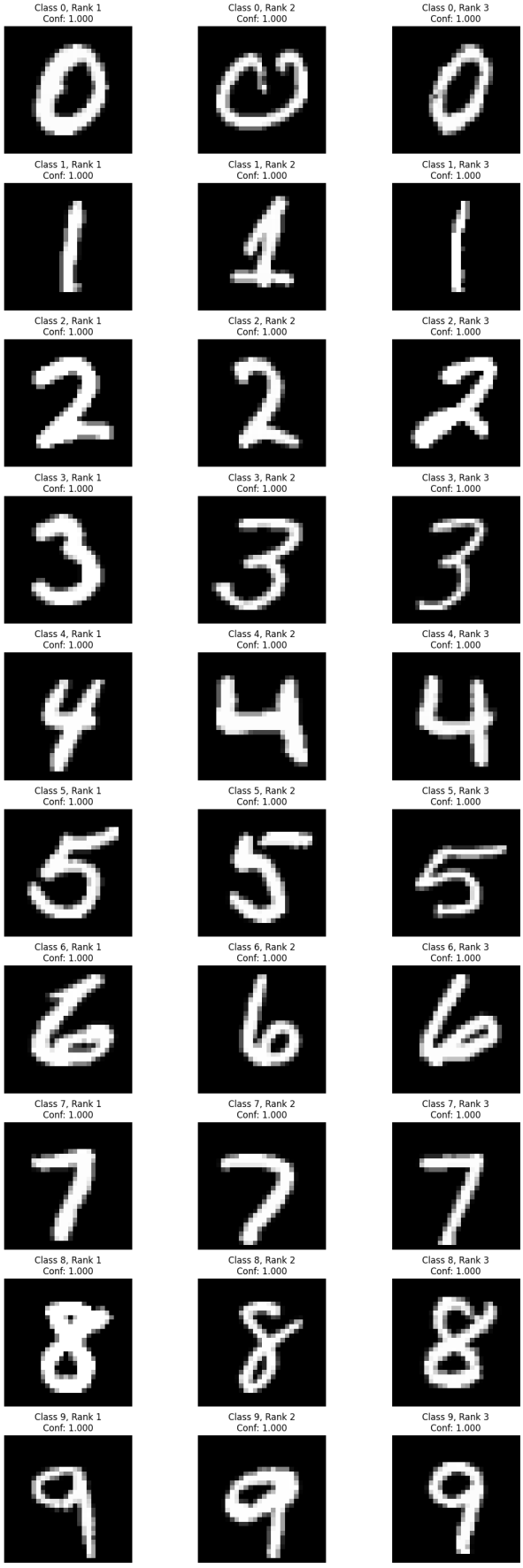
3.5 CNN (PyTorch)



[Figure 10: Loss Graph]



[Figure 11: Confusion Matrix]



[Figure 12: Top 3 Images]

4. Discussion

4.1 Key Findings

CNN vs. NN:

- CNN models achieve higher test accuracy (98.17-98.77%) than NN models (97.94-97.97%)
- CNNs converge faster with fewer epochs (5-10 vs. 20)
- Convolutional layers effectively capture spatial features

Pure Python vs. PyTorch:

- Manual implementations achieve comparable accuracy to frameworks
- PyTorch shows better optimization with advanced optimizers (Adam, momentum)
- Framework implementations benefit from GPU acceleration

Overfitting:

- NN models show overfitting (train 99.97-99.99% vs. test 97.94-97.97%)
- CNN Pure Python shows best generalization (train = test = 98.17%)
- CNN PyTorch achieves highest test accuracy despite slight overfitting

Common Misclassifications:

- Digit 4 ↔ 9 (similar strokes)
- Digit 7 ↔ 1 (short horizontal stroke)
- Digit 3 ↔ 5, 8 (overlapping curves)

5. Conclusion

This study implemented and evaluated four deep learning models for MNIST digit classification. CNNs outperformed fully-connected Neural Networks with test accuracies of 98.17-98.77% compared to 97.94-97.97%, while also converging faster (5-10 epochs vs 20 epochs). Manual NumPy implementations achieved comparable results to PyTorch frameworks, validating understanding of neural network fundamentals.

CNN PyTorch achieved the best performance (98.77% in 10 epochs), demonstrating the benefits of convolutional architectures for spatial feature extraction and optimized frameworks for image classification. These results confirm that CNNs provide distinct advantages over fully-connected networks for computer vision tasks through local connectivity and parameter sharing

References

1. LeCun, Y., et al. (2010). MNIST handwritten digit database. http://yann.lecun.com/exdb/mnist
2. Goodfellow, I., et al. (2016). Deep Learning. MIT Press.
3. Paszke, A., et al. (2019). PyTorch: An Imperative Style, High-Performance Deep Learning Library. NeurIPS.

GitHub Repository: https://github.com/yongjoon2001/DGIST-Deep-Learning-PA1