Project 2: MNIST Classification Using Feedforward and Convolutional Neural Networks

Project Overview:

This project aims to classify handwritten digits from the MNIST dataset using two neural network architectures: a feedforward neural network (FFNN) with fully connected layers and a convolutional neural network (CNN). The goal is to achieve a testing accuracy of 95% or higher with high probability. The project uses PyTorch for model implementation and is executed in a Google Colab environment.

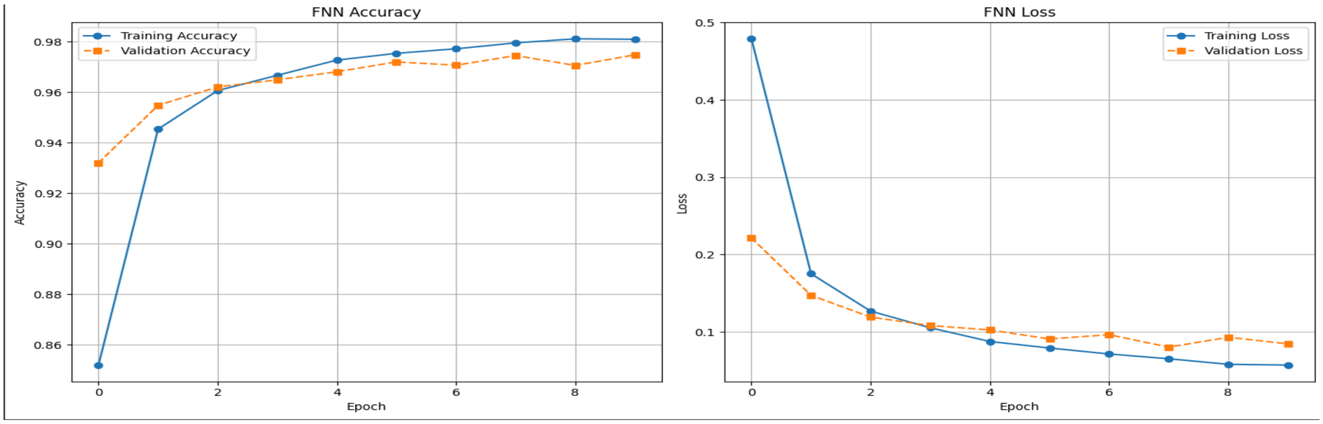
Datasets:

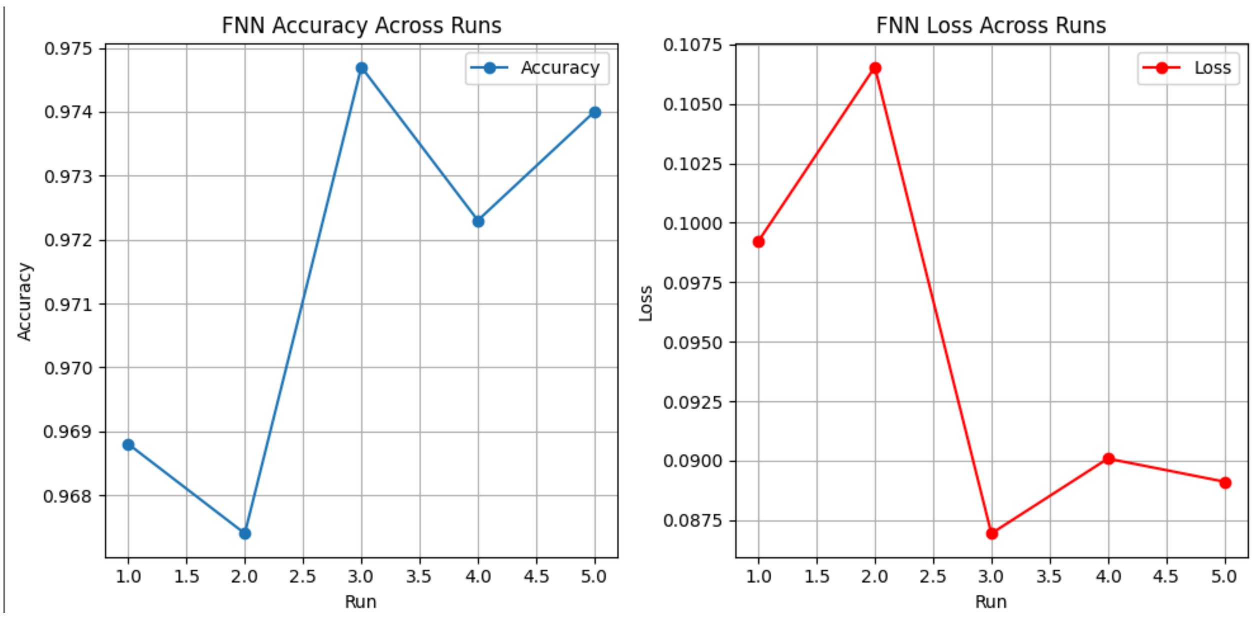
- The MNIST dataset consists of grayscale images of handwritten digits (0–9) for classification, with 60,000 training images (train-images.idx3-ubyte) and corresponding labels (train-labels.idx1-ubyte), along with 10,000 testing images (t10k-images.idx3-ubyte) and their labels (t10k-labels.idx1-ubyte).
- Each image is of size 28×28 pixels, representing a single handwritten digit, and the labels are integers in the range {0, 1, ..., 9} corresponding to the digit depicted in the image.
- The dataset is provided in .idx format, and the goal is to use this data to train and test two neural network architectures: a feedforward neural network with at least two hidden layers and a convolutional neural network with at least two convolutional layers and two fully connected layers.

Results obtained:

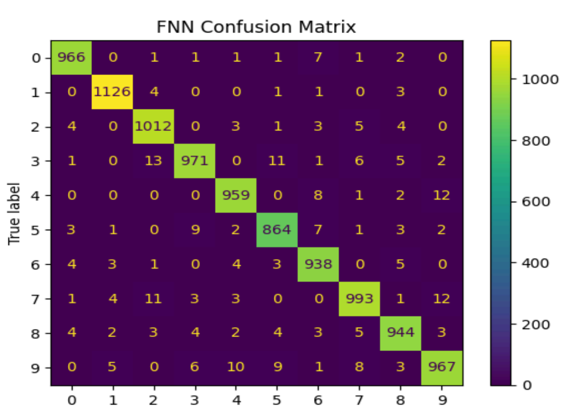
1. Feedforward Neural Network (FFNN)

- Average Testing Accuracy: 97.14%
- Average Testing Loss: 0.0944
- Performance Across Runs:
    - Run 1: Accuracy = 96.88%, Loss = 0.0992
    - Run 2: Accuracy = 96.74%, Loss = 0.1065
    - Run 3: Accuracy = 97.47%, Loss = 0.0869
    - Run 4: Accuracy = 97.23%, Loss = 0.0901
    - Run 5: Accuracy = 97.40%, Loss = 0.0891
- Visualizations:
    - Accuracy and Loss Plots (Training vs. Validation across epochs)
    - Confusion Matrix: Minimal misclassifications, with most digits correctly classified.



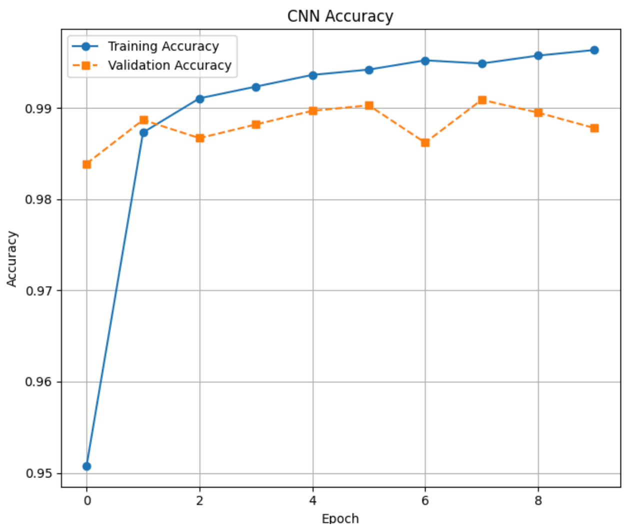


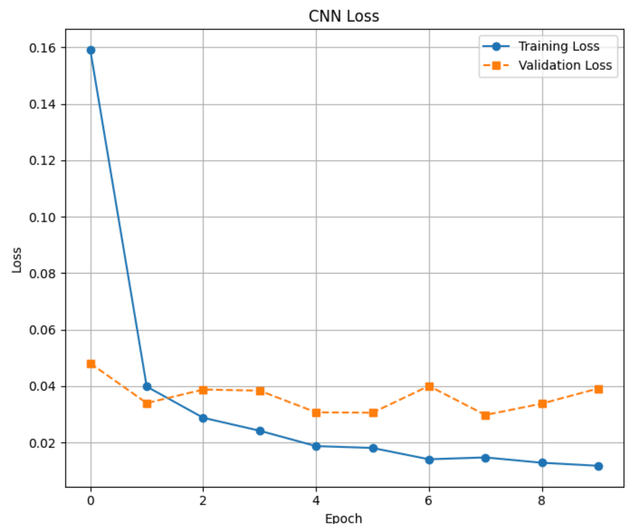
Confusion Matrix for FNN:

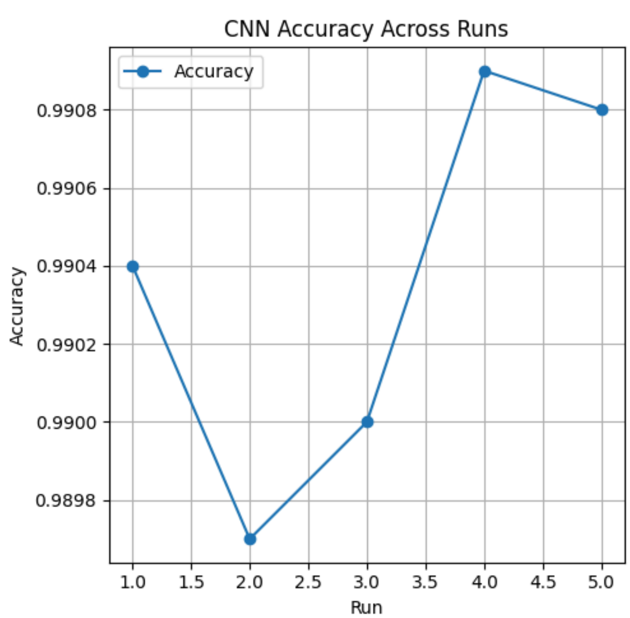


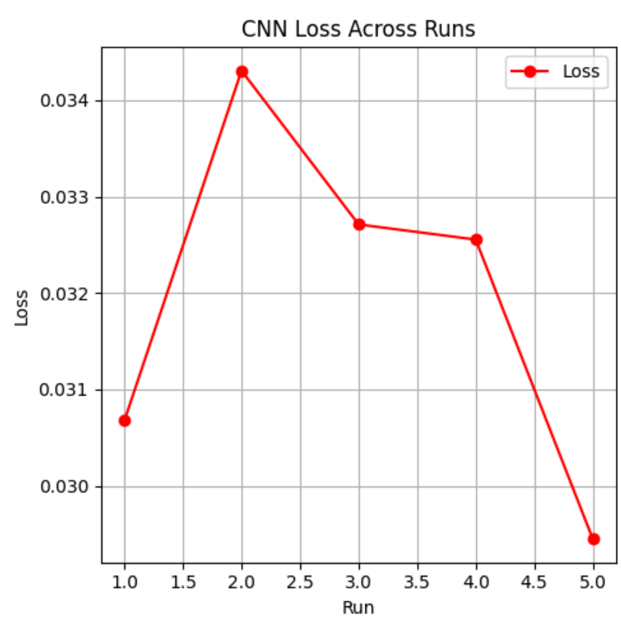
2. Convolutional Neural Network (CNN)

- Average Testing Accuracy: 99.04%
- Average Testing Loss: 0.0319
- Performance Across Runs:
  - Run 1: Accuracy = 99.04%, Loss = 0.0307
  - Run 2: Accuracy = 98.97%, Loss = 0.0343
  - Run 3: Accuracy = 99.00%, Loss = 0.0327
  - Run 4: Accuracy = 99.09%, Loss = 0.0326
  - Run 5: Accuracy = 99.08%, Loss = 0.0294
- Visualizations:
  - Accuracy and Loss Plots (Training vs. Validation across epochs)
  - Confusion Matrix: Very few misclassifications, demonstrating excellent performance.

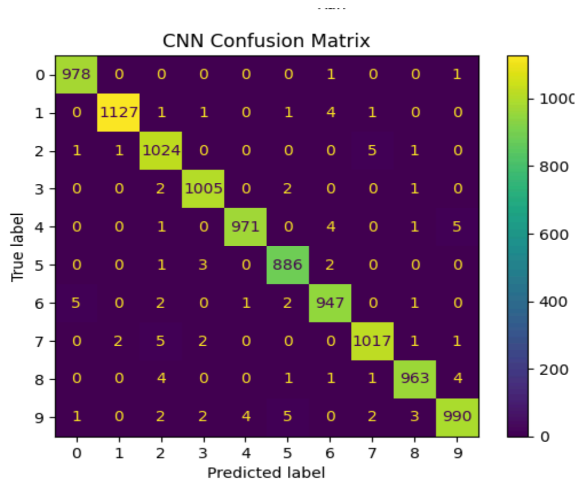








Confusion Matrix for CNN :



Both FFNN and CNN were used to achieve the aims of the project: an accuracy of over 95%. FFNN averaged 97.14% while CNN reached an even more admirable figure of 99.04%. CNN is more effective in image classification because of its lower loss (0.0319 versus 0.0944) and its better handling of spatial features. While FFNN performs well enough, CNN is the preferred model due to higher accuracy and relatively stable performance.

Lessons Learnt while working on the architecture design and parameter selection:

The key insights learnt while working on this project are:

1.Choosing the Right Architecture for the Task:The Feedforward Neural Network (FFNN) was designed with three fully connected layers and achieved the required accuracy. However, the Convolutional Neural Network (CNN), with its convolutional and pooling layers, performed better in terms of accuracy, showcasing its ability to capture spatial features critical for image classification tasks.

2.Role of Regularization:Adding dropout layers in both FFNN and CNN architectures effectively minimized overfitting, as evidenced by the close match between training and validation accuracies in the training metrics.

3.Optimization Algorithm Impact:Using Adam optimizer with weight decay improved training convergence, reduced manual tuning of learning rates, and ensured that both models quickly stabilized their performance within 10 epochs.

4.Importance of Preprocessing:Normalizing the MNIST dataset to scale pixel values to the range [0, 1] significantly improved the stability of the training process for both models, as shown by the steady decline in losses.

5.Filter Count and Layer Design in CNN:Increasing the number of filters in CNN from 32 to 64 in subsequent layers improved its feature extraction capabilities. The MaxPooling layers efficiently reduced feature map sizes while retaining critical spatial information, contributing to the CNN's superior accuracy.

6.Epoch Selection:Training both FFNN and CNN for 10 epochs was sufficient for convergence, as evidenced by the stabilization of accuracy and loss metrics, avoiding unnecessary computation and time.

7.Performance Stability Across Runs:Repeated runs of both models highlighted their stability and consistency. For instance, the CNN consistently achieved an average accuracy of over 99%, validating the reliability of the chosen architecture and parameters.