Mnist Images Analysis Report

1. Main Objective

The primary goal of this analysis is to classify handwritten digits from the MNIST dataset using three types of deep learning models: Convolutional Neural Networks (CNN), Recurrent Neural Networks (RNN), and Multi-Layer Perceptrons (MLP). Each model leverages unique architecture principles to process the input data and learn patterns for accurate digit recognition.

This analysis demonstrates the strengths and weaknesses of each model for image classification. The insights are beneficial for businesses relying on automated digit recognition systems, such as bank check verification, postal sorting, or document digitization. By understanding which model performs best, stakeholders can optimize systems for higher accuracy, reduced errors, and operational efficiency.

Additionally, this project highlights how architectural choices in deep learning can influence performance based on the nature of the data, offering a framework for future decision-making in similar applications.

2. Dataset Description

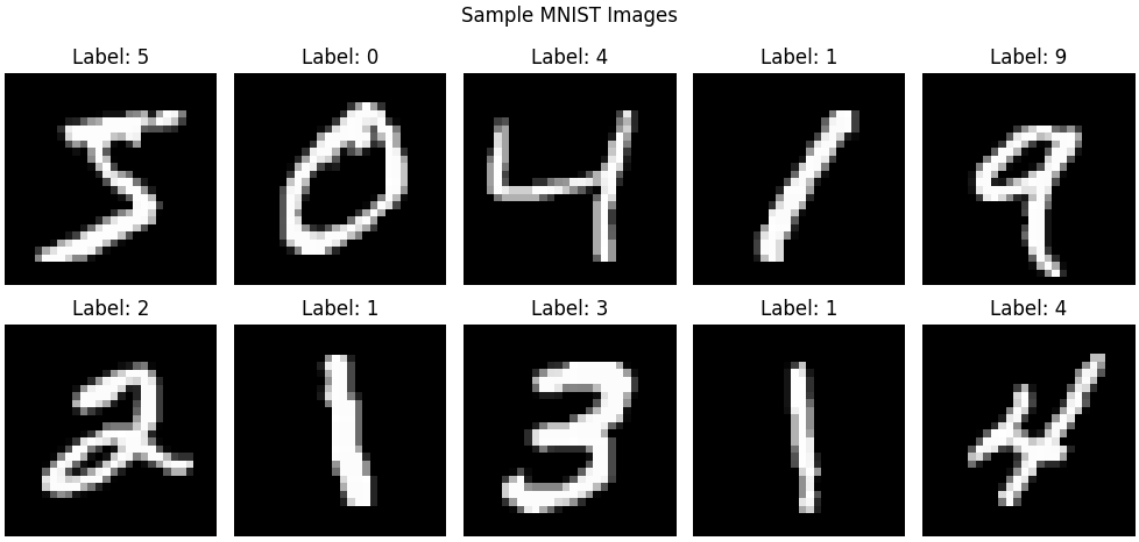
The MNIST dataset is a benchmark dataset in deep learning, consisting of handwritten digits (0–9). It is widely used to test and compare machine learning and deep learning models. Key characteristics of the dataset include:

- **Total Samples**: 70,000 images split into 60,000 training samples and 10,000 test samples.
- **Image Size**: Each image is a grayscale 28 x 28 matrix.
- **Balanced Classes**: All digits (0–9) are evenly represented, ensuring unbiased model evaluation.

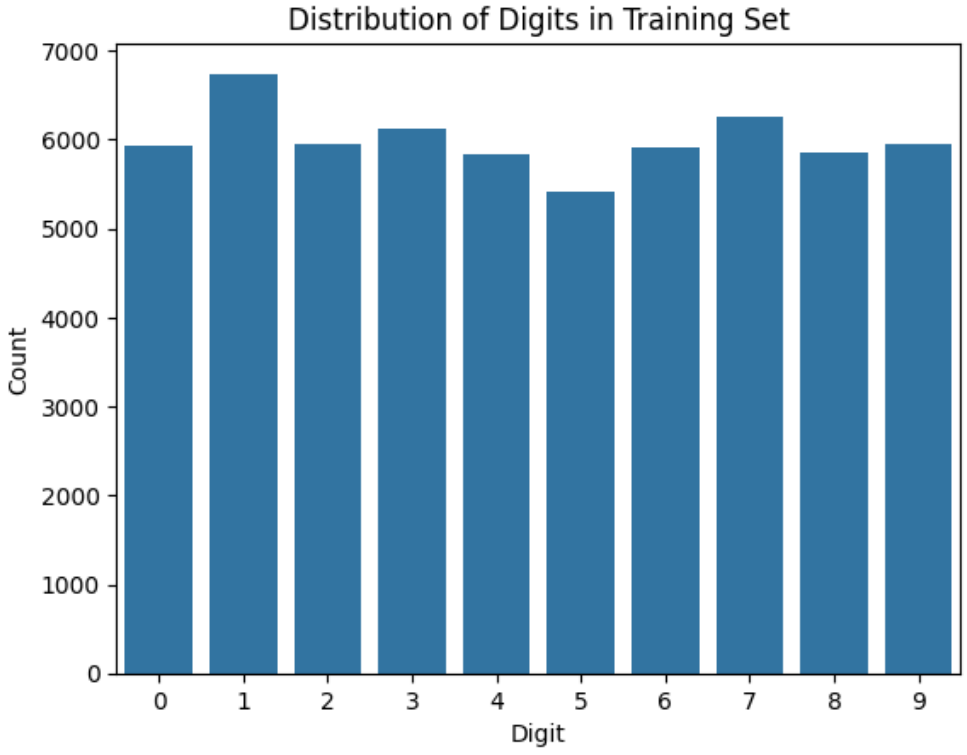
Graphs and visualizations:

- **Sample Images**: Several sample images were plotted to give a sense of the diversity in handwriting styles, such as variations in size, stroke thickness, and

slant.



• **Class Distribution**: A bar chart showed the uniform distribution of digits across the dataset, confirming the absence of class imbalance.



These insights are crucial for understanding the dataset and ensuring that our models are evaluated fairly across all digit classes.

3. Data Exploration and Cleaning

Before model training, the dataset was thoroughly explored and preprocessed:

• **Exploration**:

• Visualized a random set of images to examine handwriting styles and variations.

• Plotted the distribution of digits in the dataset to confirm balance.

• **Preprocessing**:

1. Normalization: Pixel values (0–255) were scaled to the range [0, 1] for numerical stability and faster convergence during training.

2. Reshaping:

• CNN: Images were reshaped to 28 \times 28 \times 1 to include the channel dimension.

• RNN: Images were reshaped to 28 \times 28 for sequential processing of rows.

• MLP: Images were flattened to a 1D vector of 784 features (28 \times 28).

3. One-Hot Encoding: Target labels were converted into one-hot vectors to suit the categorical output of deep learning models.

These steps ensured that the dataset was prepared appropriately for each architecture while minimizing the risk of data-related issues during training.

4. Models and Training

Three deep learning models were developed and trained to classify the digits:

1. **Convolutional Neural Network (CNN)**:

• **Architecture**:

• Two convolutional layers with 32 and 64 filters, respectively, to extract spatial hierarchies.

• Max-pooling layers to down-sample the feature maps and reduce computational complexity.

• Fully connected dense layers for classification.

• **Training Results**:

- CNN achieved **99.2% accuracy** on the test set.
- Its hierarchical feature extraction made it highly effective for image data.

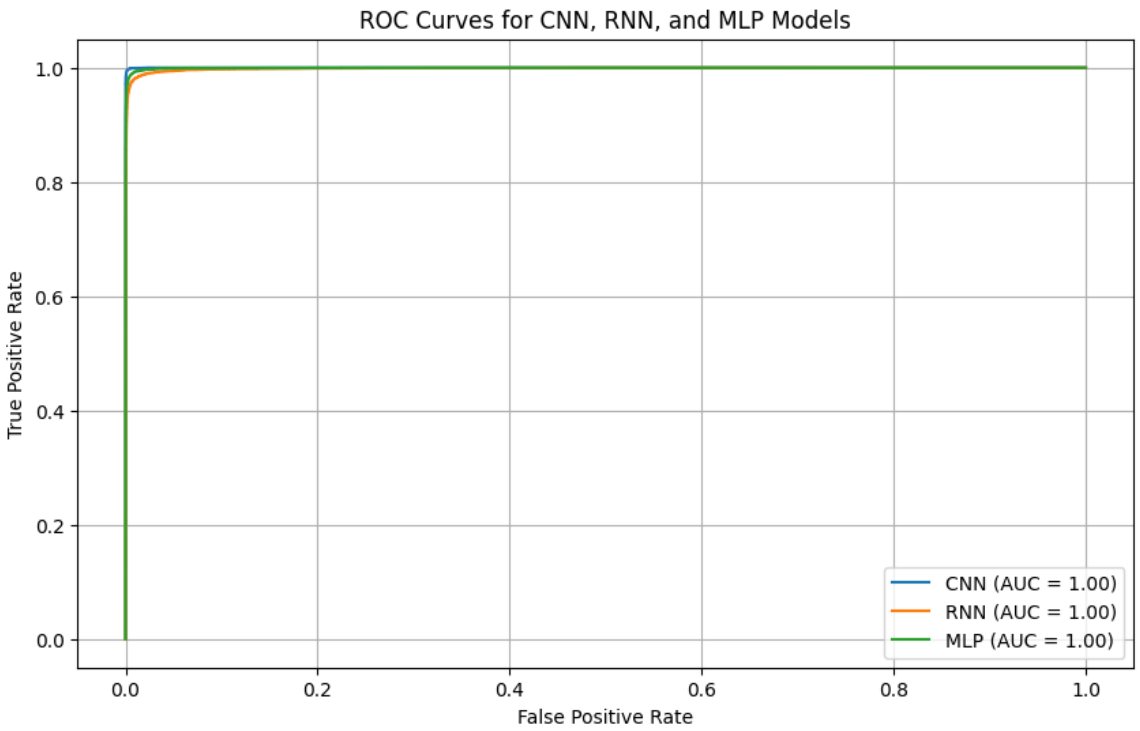
2. **Recurrent Neural Network (RNN)**:

- **Architecture**:
  - One SimpleRNN layer to process sequences row by row.
  - Dense layers for final classification.
- **Training Results**:
  - RNN achieved **96.4% accuracy** on the test set.
  - Its sequential nature was less effective for spatial data, leading to poorer performance.

3. **Multi-Layer Perceptron (MLP)**:

- **Architecture**:
  - Two fully connected dense layers with 256 and 128 neurons, respectively.
  - Flattened input images as 1D vectors.
- **Training Results**:
  - MLP achieved **97.7% accuracy** on the test set.
  - While better than RNN, it lacked the spatial feature extraction capability of CNN.

Each model was trained for 5 epochs using the Adam optimizer and categorical cross-entropy loss.



The ROC curves for the CNN, RNN, and MLP models all exhibit near-perfect AUC values of 1.0, indicating excellent classification performance. The models demonstrate a strong ability to distinguish between the classes, with very low false positive rates and high true positive rates across all thresholds. This reflects that all three models performed exceptionally well on the MNIST dataset, with minimal misclassifications. However, CNN's suitability for image data and its higher interpretability still make it the most robust choice.

5. Recommended Model

CNN is recommended as the best model for this task due to its:

- **High Accuracy**: Achieved the highest test accuracy of **99.2%**, outperforming RNN and MLP.
- **Suitability for Image Data**: Its convolutional layers excel at extracting spatial features, such as edges and textures, making it ideal for image classification tasks.
- **Scalability**: CNNs can easily be extended to handle more complex datasets or tasks with minimal architectural changes.

RNN is not recommended for this type of problem, as it is designed for sequential data like text or time-series, not images. While MLP performed reasonably well, it lacks the hierarchical learning capability that gives CNNs their edge in image-related tasks.

6. Key Findings and Insights

The analysis revealed several important insights:

1. **Model Performance**:

   • CNNs excel at image classification due to their ability to learn spatial hierarchies.

   • RNNs struggled with image data, reinforcing their role as sequential data processors.

   • MLPs performed well but were limited by their inability to leverage spatial structures.

2. **Dataset Characteristics**:

   • The MNIST dataset's balanced distribution of digits ensured unbiased evaluation.

   • Despite its simplicity, MNIST highlighted the architectural strengths and weaknesses of the models.

3. **Generalization**:

   • CNNs demonstrated strong generalization, suggesting their potential for more complex real-world datasets.

These findings confirm that CNNs are the most appropriate choice for tasks involving image data.

7. Suggestions for Next Steps

To improve and extend the analysis, the following steps are recommended:

1. **Explore Advanced Architectures**:

   • Implement deeper CNN architectures like ResNet or VGG for even better performance.

   • Explore hybrid architectures, combining CNNs and RNNs, for tasks involving spatiotemporal data.

2. **Apply Data Augmentation**:

   • Techniques like rotation, flipping, and zooming can increase dataset variability, improving model robustness and generalization.

3. **Experiment with Larger Datasets**:

   • Test the models on more complex datasets (e.g., CIFAR-10 or ImageNet) to evaluate scalability and performance on diverse image types.

4. **Hyperparameter Tuning**:

   • Fine-tune learning rates, batch sizes, and number of layers to optimize performance further.

By following these steps, the models can be refined and scaled to handle more challenging problems, ensuring their applicability to real-world tasks.