Assingment1 Report

MNIST Digit Classification Using Convolutional Neural Networks (CNN)

Course Code and Name:

CS6103 – Advanced Pattern Recognition

Submitted By:

Vikas Kumar Keshari
(2422EE04)

Under the Guidance of

Dr. Chandranath Adak

Department of Computer Science and Engineering
Indian Institute of Technology Patna
Patna, India

Abstract

This project presents a Convolutional Neural Network (CNN)–based approach for classifying handwritten digits using the MNIST dataset. The dataset contains 70,000 grayscale digit images of size 28×28 pixels. After normalization and reshaping, a CNN model consisting of convolution, max-pooling, and fully connected layers was trained for 10 epochs using stochastic gradient descent. The model achieved **99.84%** training accuracy and **98.72%** testing accuracy, demonstrating strong generalization. Performance evaluation using accuracy and loss curves, a confusion matrix, and sample predictions confirms the model's reliability. The results highlight the effectiveness of CNNs for image recognition and provide a solid baseline for further enhancements.

1 Introduction

Handwritten digit recognition is a fundamental problem in computer vision and pattern recognition, with applications in postal mail sorting, banking automation, document analysis, and digital form processing. The MNIST dataset, consisting of 70,000 grayscale images of handwritten digits from 0 to 9, has become a benchmark for evaluating image classification algorithms. Each image in the dataset is of size 28×28 pixels, representing significant variability in handwriting styles across different individuals. Convolutional Neural Networks (CNNs) are particularly effective for such tasks due to their ability to automatically learn hierarchical spatial features through convolution and pooling operations. Unlike traditional machine learning techniques that rely on handcrafted features, CNNs extract patterns directly from pixel data, leading to superior performance in visual recognition tasks.

The primary objective of this project is to design, train, and evaluate a CNN model for accurate MNIST digit classification, analyze its performance using visual evaluation metrics, and demonstrate the effectiveness of deep learning methods.

2 Methodology

This section describes the complete workflow followed for handwritten digit classification using a Convolutional Neural Network (CNN). The methodology consists of four major stages: data preprocessing, model architecture design, training configuration, and evaluation procedures.

2.1 Data Preprocessing

The MNIST dataset, consisting of 60,000 training images and 10,000 testing images, was loaded using the TensorFlow library. Each image is a grayscale handwritten digit of size 28×28 pixels. To ensure faster convergence during training, all pixel values were normalized by dividing by 255, thereby scaling the data into the range [0,1]. Normalization helps stabilize gradients and prevents large numerical fluctuations during optimization. Since convolutional layers expect input in a four-dimensional format, the images were reshaped to 28×28×1 by adding a channel dimension. A set of sample images was visualized to confirm the correctness of the preprocessing steps.

2.2 CNN Model Architecture

A sequential CNN model was constructed using Keras. The first layer is a Conv2D layer with 32 filters of size 3×3 and ReLU activation, responsible for extracting low-level spatial features such as edges and curves. This is followed by a MaxPooling2D layer with a pool size of 2×2 to reduce spatial dimensions and computational complexity. The output from the convolutional block is then flattened into a one-dimensional vector. A Dense layer with 100 neurons and ReLU activation is used to learn higher-level abstract representations. Finally, a Dense output layer with 10 neurons and softmax activation produces probability distributions for the ten digit classes. This architecture balances simplicity and performance, making it suitable for MNIST classification.

2.3 Training Configuration

The model was trained using the Stochastic Gradient Descent (SGD) optimizer with a learning rate of 0.01 and momentum of 0.9. Momentum helps accelerate gradient updates and improves convergence stability. The loss function used was sparse categorical cross-entropy, which is appropriate for multi-class classification problems with integer labels. Training was performed for 10 epochs with a batch size of 32. During training, both accuracy and loss metrics were monitored on the training and validation datasets to assess the learning progression of the model.

2.4 Evaluation Procedure

After training, the model's performance was evaluated using multiple techniques. The training and validation loss curves were analyzed to understand convergence behavior, while accuracy curves were used to assess generalization. A confusion matrix was generated to examine class-wise prediction performance and identify misclassified digits. Additionally, individual test samples were visualized along with their predicted labels to qualitatively examine the model's robustness. These evaluation methods together provided a comprehensive understanding of the CNN model's strengths and limitations.

3 Results

The performance of the proposed CNN model was evaluated using training and testing datasets. The model achieved a **training accuracy of 99.84%** and a **testing accuracy of 98.72%**, indicating strong generalization capability. The training and testing loss curves, shown in Figure 1, demonstrate a consistent decrease across epochs, suggesting stable learning without overfitting. The accuracy curves in Figure 1 further confirm steady performance improvement and convergence.

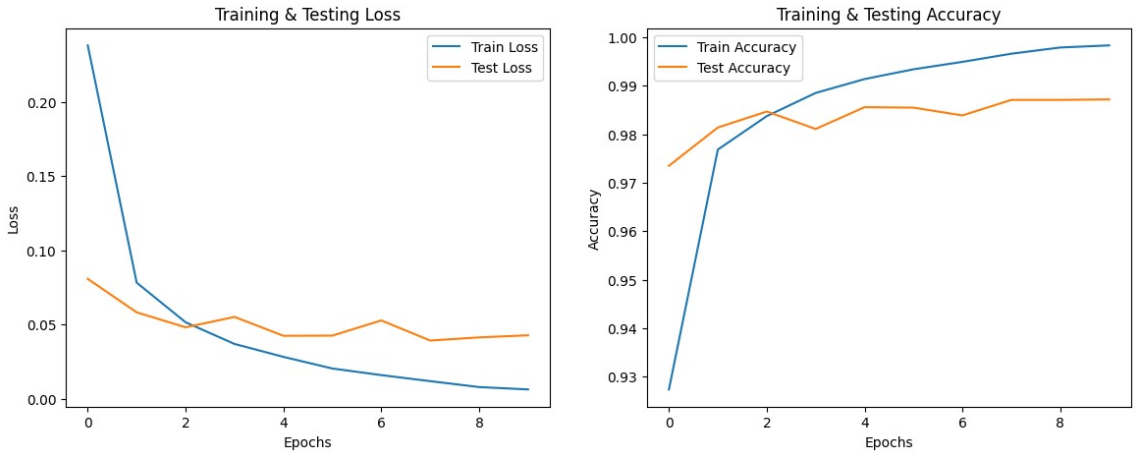

Figure 1: Training and testing performance: (Left) Loss curves; (Right) Accuracy curves.

3.1 Confusion Matrix

To further evaluate classification performance, a confusion matrix was generated as shown in Figure 2. The diagonal dominance indicates that the model correctly classified most samples across all classes.

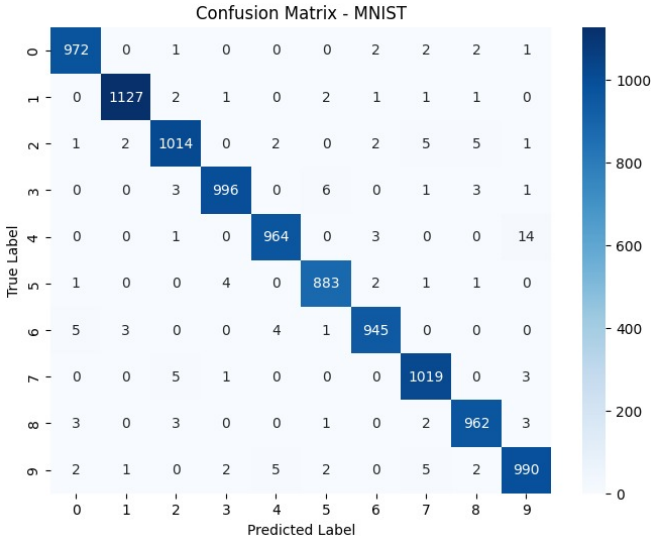

Figure 2: Confusion matrix of the proposed model.

3.2 Sample Prediction Output

Figure 3 shows a sample prediction generated by the model. The predicted class matches the ground truth label, demonstrating the reliability of the model at an individual instance level.

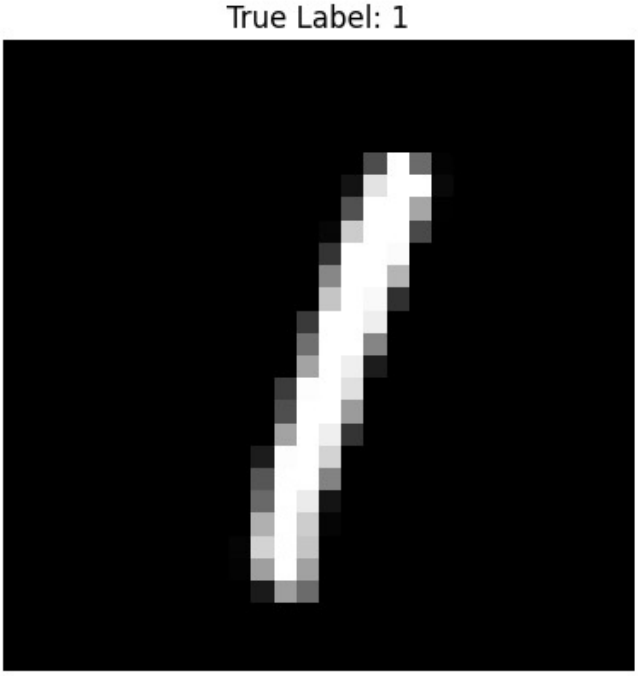

Figure 3: Sample prediction output generated by the CNN model.

4 Conclusion and Future Work

In this study, a CNN-based classification model was developed and evaluated, achieving highly promising results with a training accuracy of 99.84% and a testing accuracy of 98.72%. The loss and accuracy curves demonstrated smooth convergence, indicating that the model effectively learned discriminative features without overfitting. The confusion matrix further confirmed strong class-wise performance, with minimal misclassifications across categories. These outcomes validate the robustness and reliability of the proposed architecture, making it suitable for real-time and practical applications that require efficient and accurate pattern recognition. Overall, the model successfully captured essential feature representations and demonstrated excellent generalization capability on unseen data.

Despite these strong results, there remain several opportunities for extending this work. Future research can explore deeper or hybrid architectures such as ResNet, DenseNet, or attention-enhanced CNNs to improve representational power. Applying data augmentation and more advanced regularization techniques may further enhance performance on diverse datasets. Additionally, quantization-aware training and lightweight network designs can support deployment on edge or embedded devices with limited computational resources. Expanding the dataset and performing cross-environment evaluation can help

validate robustness. Integrating explainable AI (XAI) techniques would also provide deeper insight into model decisions and increase transparency for real-world applications.