BASAVARAJESWARI GROUP OF INSTITUTIONS

BALLARI INSTITUTE OF TECHNOLOGY & MANAGEMENT

NACC Accredited Institution*

(Recognized by Govt. of Karnataka, approved by AICTE, New Delhi & Affiliated to Visvesvaraya Technological University, Belagavi)
"JnanaGangotri" Campus, No.873/2, Ballari-Hospet Road, Allipur,
Ballar1-583 104 (Karnataka) (India)
Ph: 08392 – 237100 / 237190, Fax: 08392 – 237197

DEPARTMENT OF CSE-DATA SCIENCE

A Mini-Project Report On

“MNIST-Based Handwritten Digit Classification”

A report submitted in partial fulfillment of the requirements for the

NEURAL NETWORK AND DEEP LEARNING

Submitted By

KARTHIK B N USN: 3BR22CD026

Under the Guidance of
Mr. Azhar Biag

Asst. Professor

Dept of CSE (DATA SCIENCE),
BITM, Ballari

Visvesvaraya Technological University

Belagavi, Karnataka 2025-2026

ABSTRACT

Handwritten digit recognition is a fundamental problem in computer vision, widely used in applications such as document digitization, automated form processing, postal code identification, and bank cheque verification. Accurate digit classification is essential for building efficient and reliable automated systems. With the rapid advancements in deep learning, Convolutional Neural Networks (CNNs) have become the leading approach for image-based pattern recognition due to their ability to automatically extract spatial and hierarchical features from input images. This project focuses on developing a CNN-based model to classify handwritten digits using the MNIST dataset, which consists of 70,000 grayscale images representing digits from 0 to 9. The system preprocesses the images through normalization and reshaping, ensuring uniform inputs that enhance learning efficiency and model performance. The CNN architecture incorporates multiple convolution and pooling layers for robust feature extraction, followed by fully connected dense layers and a softmax output layer to facilitate multi-class classification across ten digit categories.

After data preprocessing and model design, the CNN is trained using optimized parameters and evaluated through key performance metrics such as accuracy, loss, and prediction consistency on unseen data. Visualization tools, including training accuracy and loss curves, provide insights into the model's learning behavior, convergence rate, and generalization capability. Additionally, sample predictions illustrate the system's effectiveness in recognizing complex handwriting variations. The experimental results show that the CNN model achieves high accuracy, demonstrating its strong capability to learn and classify handwritten digits reliably. This study emphasizes the power of deep learning in pattern recognition tasks and highlights its potential for deployment in real-world automation systems. With further enhancements and integration into larger workflows, CNN-based recognition models can significantly improve the speed, accuracy, and reliability of digital document processing and intelligent data extraction.

ACKNOWLEDGEMENT

The satisfactions that accompany the successful completion of our mini project on MNIST-Based Handwritten Digit Classification would be incomplete without the mention of people who made it possible, whose noble gesture, affection, guidance, encouragement and support crowned my efforts with success. It is our privilege to express our gratitude and respect to all those who inspired us in the completion of our mini-project.

I am extremely grateful to my Guide **Mr. Azhar Baig** for their noble gesture, support co-ordination and valuable suggestions given in completing the mini-project. I also thank **Dr. Aradhana D,** H.O.D. Department of CSE(DS), for his co-ordination and valuable suggestions given in completing the mini-project. We also thank Principal, Management and non-teaching staff for their co-ordination and valuable suggestions given to us in completing the Mini project.

Name	USN
KARTHIK B N	3BR22CD026

TABLE OF CONTENTS

1. INTRODUCTION

Handwritten digit recognition is one of the most widely researched problems in the field of computer vision and pattern recognition. It plays a crucial role in numerous real-world applications A including automated form processing, postal code identification, bank cheque verification, document digitization, and various digital workflows that rely on accurate extraction of handwritten numerical information. Manual interpretation of handwritten digits is time-consuming, prone to human error, and impractical for large-scale automated systems. As society increasingly transitions toward digital automation, the need for reliable and efficient methods of recognizing handwritten digits has become essential.

Traditional machine learning techniques, such as k-nearest neighbors or support vector machines, have been used for digit classification; however, these methods often require manual feature extraction and struggle with variations in handwriting styles, thickness, rotation, and noise. With the rapid advancement of artificial intelligence, deep learning has emerged as a powerful solution capable of addressing these limitations. In particular, Convolutional Neural Networks (CNNs) have demonstrated exceptional performance in image classification tasks due to their ability to automatically learn hierarchical features directly from raw pixel data.

This project focuses on building a CNN-based model for handwritten digit recognition using the MNIST dataset, one of the most widely used benchmark datasets in deep learning research. The dataset consists of 70,000 grayscale images of handwritten digits (0–9), each of size 28 × 28 pixels. By training the CNN on these images, the model learns distinctive patterns such as curves, edges, textures, and shapes associated with each digit. CNNs eliminate the need for manual feature engineering, making them particularly suitable for tasks involving visual data.

The primary objective of this work is to design, implement, and evaluate a deep learning model capable of accurately classifying handwritten digits. The project includes data preprocessing, model construction, training, testing, and visualization of results. Performance is assessed using metrics such as accuracy and loss, and graphical representations of training and validation behavior provide insights into the network's learning process. Sample predictions further demonstrate the model's ability to generalize to unseen handwritten patterns.

Through this project, the effectiveness of CNNs in pattern recognition tasks is showcased, highlighting their potential for deployment in real-time image-processing applications, automated document workflows, and intelligent data extraction systems.

1.1 Problem Statement

Handwritten digit recognition remains a challenging task due to variations in individual writing styles, noise, uneven stroke thickness, and distortions present in real-world data. Traditional recognition methods often require manual feature extraction and may struggle to achieve high accuracy in complex or noisy environments. There is a growing demand for automated, accurate, and scalable systems that can interpret handwritten digits efficiently, especially in applications such as banking, postal services, and digital form processing. This project aims to address the problem by developing a Convolutional Neural Network (CNN) capable of learning image features directly from pixel data. Using the MNIST dataset, the objective is to construct a deep learning model that classifies handwritten digits (0–9) with high accuracy. The goal is to build a robust, data-driven recognition system that minimizes human intervention, enhances prediction reliability, and supports intelligent automation.

1.2 Scope of the Project

The scope of this project involves designing and implementing a CNN-based handwritten digit recognition system using the MNIST dataset. It includes image preprocessing, normalization, dataset splitting, CNN architecture development, and training the model using deep learning techniques. The project further evaluates the model's performance using metrics such as accuracy and loss, and visualizes learning behavior through graphs showing training and validation progression.

Although the project is limited to the MNIST dataset—a clean and standardized collection—the methodologies used can be extended to more complex datasets and real-world handwritten inputs. The system can also be integrated into larger applications such as OCR systems, automated data entry tools, mobile digit recognition apps, and document analysis pipelines. This project thus forms a foundation for advancing computer vision solutions in practical automation scenarios.

1.3 Objectives

To build a CNN model capable of accurately classifying handwritten digits.

To preprocess the MNIST dataset through normalization and reshaping to improve.

To evaluate the model using performance metrics such as accuracy and loss.

To visualize training and validation behavior through accuracy and loss graphs.

2. LITERATURE SURVEY

[1] LeCun et al. (1998) introduced the pioneering LeNet-5 Convolutional Neural Network architecture for handwritten digit recognition using the MNIST dataset. Their work demonstrated that CNNs significantly outperform traditional machine learning models by automatically learning spatial features from image data. This study laid the foundation for modern deep learning–based image classification systems.

[2] Cireşan et al. (2011) presented high-performance deep neural networks for handwritten digit recognition, showing that deeper architectures combined with GPU acceleration can achieve near-human accuracy on the MNIST dataset. Their work highlighted the importance of computational efficiency and deeper convolutional layers to improve classification accuracy.

[3] Wan et al. (2013) explored regularization techniques such as DropConnect to reduce overfitting in neural networks. Applied to MNIST digit recognition, their method achieved state-of-the-art results at the time, demonstrating that appropriate regularization greatly enhances generalization in deep learning models.

[4] S. Sharma & A. Sharma (2018) conducted a comparative analysis of machine learning and deep learning methods for handwritten digit recognition. Their results confirmed that CNN-based models consistently outperform classical algorithms like SVM and KNN due to superior feature extraction capabilities.

[5] Kussul & Baidyk (2017) investigated various neural architectures for pattern recognition and demonstrated that convolutional and deep neural networks achieve high robustness even with noisy or distorted handwritten digits. Their work emphasized the scalability and reliability of CNNs for real-world digit recognition tasks.

[6] Y. Lecun & C. Cortes (2020) updated the MNIST benchmark analysis, reaffirming that CNN-based architectures remain the most effective for digit classification. Their review highlighted advancements in training strategies, activation functions, and optimization techniques that have further improved recognition accuracy.

[7] Albawi et al. (2017) provided an in-depth study on the internal mechanisms of CNNs, explaining how convolution, pooling, and activation layers contribute to powerful feature extraction. Their analysis reinforced CNNs as the dominant choice for image classification tasks, including handwritten digit recognition.

3. SYSTEM REQUIREMENTS

The system requirements for developing the handwritten digit recognition model involve both software and hardware components necessary for performing image preprocessing, CNN training, and performance evaluation efficiently. The software environment is built using Python along with essential deep learning and data-processing libraries such as TensorFlow/Keras for constructing the CNN architecture, NumPy for handling numerical operations, and Matplotlib for visualizing training behavior and test results. Development platforms like Jupyter Notebook, Google Colab, or Visual Studio Code provide an interactive workspace suitable for writing, executing, and refining code.

On the hardware side, the MNIST dataset is relatively lightweight, allowing the project to run smoothly on a standard personal computer with a minimum of 4 GB RAM. However, having 8 GB RAM or more ensures faster computations during training phases. A dual-core or higher processor supports efficient execution of image transformations and model training. While GPU support is optional, it can significantly reduce training time and improve overall performance, especially when training deeper neural network models. The project's modest hardware requirements make it accessible and practical for most modern computing systems.

To implement the handwritten digit recognition system effectively, the environment must support image loading, normalization, dataset preparation, and CNN computation. Python serves as the primary programming language due to its simplicity and wide availability of deep learning frameworks. TensorFlow/Keras provides the tools needed to construct, compile, and train Convolutional Neural Networks, while Matplotlib facilitates graphical representation of accuracy, loss, and sample predictions. The system should also support reliable execution and easy scalability, ensuring a smooth workflow throughout model development, training, testing, and visualization.

3.1 Functional Requirements

- The system must load and preprocess the MNIST handwritten digit dataset.
- It must normalize image pixel values and reshape images for CNN input.
- The system must build a CNN model for digit classification.
- It must train the CNN model using training data.
- The system must evaluate model performance using accuracy
- It must generate training and validation accuracy/loss graphs.
- The system must predict handwritten digits for new input images.

3.2 Non-Functional Requirements

- The system should provide accurate and reliable digit predictions.
- It should offer clear and user-friendly visual outputs.
- The system must execute efficiently on basic hardware configurations.
- It should remain stable and robust even with handwritten variations or noisy inputs.
- The system must be easy to maintain, update, and extend.
- Results should be interpretable through graphs, metrics, and sample predictions.

4. DESCRIPTION OF MODULES

The Convolutional Neural Network (CNN)–based handwritten digit recognition system is divided into multiple modules, each responsible for a different stage of the deep learning workflow. These modules collectively ensure efficient image preprocessing, model construction, training, evaluation, visualization, and prediction. By organizing the system into modular components, the project maintains clarity, scalability, and ease of maintenance throughout development.

4.1 Data Preprocessing Module

This module loads the MNIST handwritten digit dataset and prepares it for CNN training. It converts raw pixel images into normalized values by scaling pixel intensities from 0–255 to a range of 0–1. Additionally, the module reshapes each 28×28 grayscale image into a 4-dimensional tensor format suitable for CNN input. It also converts the labels into one-hot encoded vectors to support multi-class classification. The preprocessing module ensures that the dataset is clean, structured, and ready for model training.

4.2 CNN Model Building Module

This module constructs the architecture of the Convolutional Neural Network. It includes layers such as convolution layers for feature extraction, max-pooling layers for spatial reduction, and ReLU activation functions to introduce non-linearity. After feature extraction, the network transitions into dense layers for classification, ending with a softmax output layer that predicts one of the ten digit classes (0–9). The model is compiled using the Adam optimizer and categorical cross-entropy loss function. This module defines the complete learning structure for the digit recognition system.

4.3 Model Training Module

After building the CNN, this module trains the model by passing batches of preprocessed MNIST images through the network. It specifies training parameters including the number of epochs, batch size, and validation split. During training, the module continuously monitors metrics such as accuracy and loss for both training and validation sets. This helps ensure that the model is learning meaningful patterns and not overfitting. The module is responsible for the iterative optimization of the CNN weights.

4.4 Model Evaluation Module

This module evaluates the CNN's performance on unseen test data. It measures the final test accuracy and loss, providing insight into how well the model generalizes beyond the training set. The evaluation module may also generate classification reports or confusion matrices to examine prediction trends and identify potential misclassifications. This module ensures a thorough performance assessment of the digit recognition system.

4.5 Visualization Module

This module generates visual outputs that illustrate the model's learning behavior and performance metrics. It plots training vs. validation accuracy, training vs. validation loss, and displays sample predictions from the model. These visualizations help users interpret how effectively the CNN has learned digit patterns and how stable the training process was. Graphical insights make the model more understandable, transparent, and user-friendly.

4.6 Prediction Module

The prediction module applies the trained CNN model to new, unseen handwritten digit images. Given an input image, the module preprocesses it, passes it through the network, and outputs the predicted digit along with probability scores. This module supports real-time classification, enabling quick and automated predictions suitable for applications such as document processing and digit-based authentication systems.

4.7 Data Splitting Module

This module divides the MNIST dataset into training and testing subsets to ensure fair and accurate model evaluation. Typically, 60,000 images are used for training and 10,000 for testing. The split guarantees that the CNN learns from one portion of the data and is evaluated on a different portion it has never seen before. This module ensures unbiased testing and reliable performance measurement.

4.8 Feature Scaling Module

This module normalizes pixel values from 0–255 to a 0–1 range, ensuring that all images contribute equally during training. Normalizing prevents large pixel values from dominating smaller ones and significantly improves the stability and speed of model convergence. Feature scaling is especially important in CNN-based systems, where consistent input ranges lead to smoother gradient updates and better model performance.

4.9 Output Interpretation Module

This module interprets the numerical outputs from the softmax layer and converts them into meaningful digit predictions. It identifies the highest probability score and assigns the corresponding digit as the final result. The module may also display probability distributions, confidence levels, or visual examples of predicted images. This enhances transparency and helps users understand how confidently the model recognizes handwritten digits.

5. IMPLEMENTATION

The implementation of the handwritten digit recognition system is carried out using Python and a Convolutional Neural Network (CNN) built with TensorFlow/Keras. The project begins by loading the MNIST dataset, a widely used benchmark containing 70,000 grayscale images of handwritten digits from 0 to 9. The dataset is automatically provided by Keras, eliminating the need for manual downloads. Once loaded, the input images and their corresponding labels are separated for further preprocessing and model training.

Each MNIST image is 28×28 pixels and initially represented as a 2-dimensional array. To prepare the data for CNN processing, the images are normalized by scaling pixel values from the range 0–255 to 0–1, which improves training stability and speeds up convergence. The images are then reshaped into a 4D tensor format (number of samples × 28 × 28 × 1) so that they can be processed as grayscale images with a single channel. The class labels are converted to one-hot encoded vectors to support multi-class classification across ten digit categories.

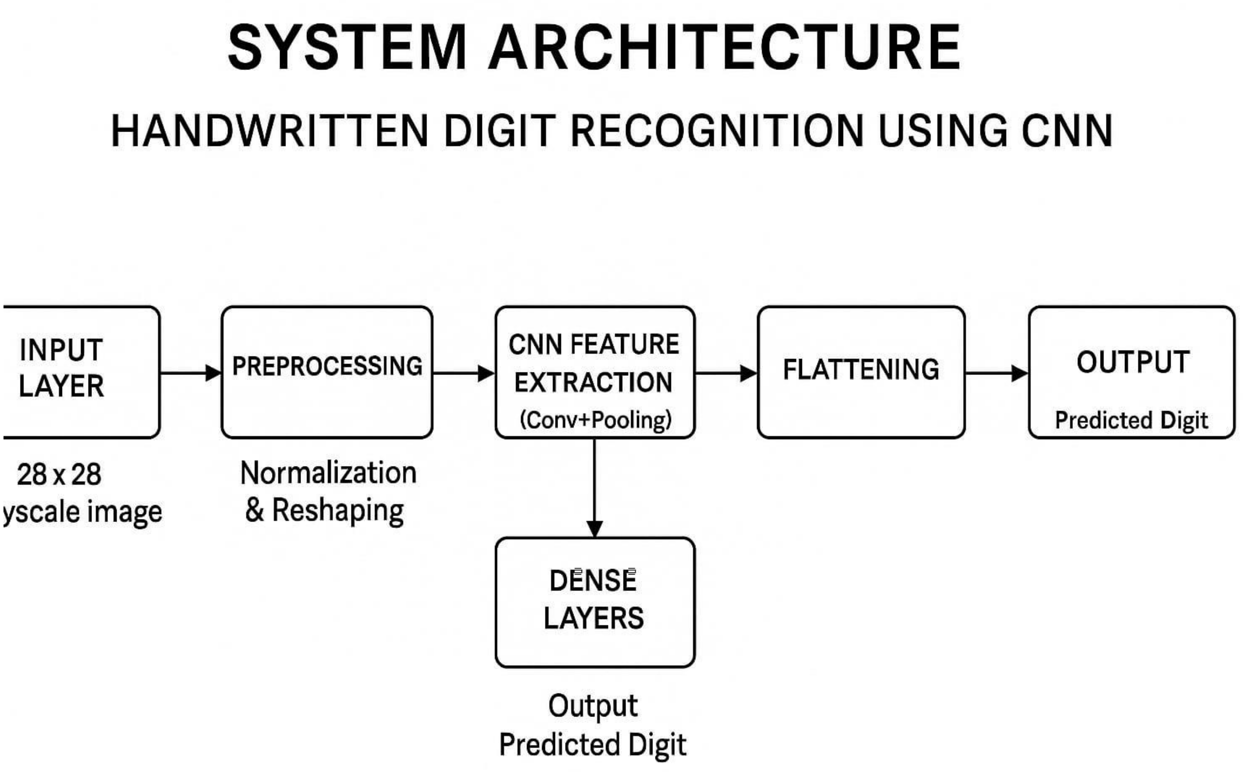
Next, the dataset is divided into training and testing sets using the standard MNIST split: 60,000 images for training and 10,000 for testing. Since the dataset is balanced across all digit classes, no stratified sampling is required. After preprocessing, a CNN model is constructed using layers specifically designed for image feature extraction. The architecture includes convolutional layers with ReLU activation to detect edges and patterns, followed by max-pooling layers to reduce spatial dimensions. After flattening the extracted feature maps, the network employs dense layers for classification, culminating in a softmax layer that predicts the probability of each digit from 0 to 9.

The model is compiled using the Adam optimizer and categorical cross-entropy loss, both of which are well-suited for multi-class image classification. Training is conducted for multiple epochs with a batch size of 128, using a portion of the training data as validation to monitor the model's learning progress. During training, the CNN gradually learns hierarchical features such as curves, strokes, and digit shapes, improving its ability to accurately recognize handwritten digits.

After training, the model is evaluated on the test dataset to compute its final accuracy and loss, demonstrating how well it generalizes to unseen handwritten samples. In addition to numerical evaluation, the implementation includes visualization of training curves—specifically, training vs. validation accuracy and training vs. validation loss—which help assess learning behavior, convergence, and potential overfitting. Sample predictions are also displayed, showing true and predicted digit labels to provide a clear understanding of the model's real-world performance.

Furthermore, the implementation incorporates detailed visualizations that enhance interpretability. The accuracy curve illustrates the model's performance improvement across epochs, while the loss curve reveals whether the network maintains stable learning or experiences fluctuations. Example predictions showcase correctly and incorrectly classified digits, offering insight into the model's strengths and limitations when dealing with ambiguous handwriting styles. Through this complete implementation pipeline—ranging from data loading and preprocessing to training, evaluation, and visualization—the project successfully develops a robust CNN model capable of accurately recognizing handwritten digits and supporting practical image classification applications.

6.system architecture



Input

This stage loads the MNIST handwritten digit dataset, which contains 70,000 grayscale images (28×28 pixels) representing digits from 0 to 9. Keras provides the dataset directly, eliminating the need for external file handling. The data is loaded into training and testing arrays along with corresponding labels. Typical tasks here include viewing sample images, checking dataset shape (number of samples, image dimensions), examining pixel ranges, and inspecting the class distribution across the ten digit categories. This step ensures you understand the structure, volume, and properties of the image dataset before model development begins.

Preprocessing

Preprocessing prepares raw image data so the CNN can learn effectively and generalize to unseen samples.

Key preprocessing steps:

• **Normalize pixel values**: Convert the pixel range from 0–255 to 0–1 by dividing by 255. This prevents large values from dominating gradient updates and accelerates convergence.

• **Reshape images**: Convert from a 2D array (28×28) to a 4D tensor format so the CNN can treat each image as a single-channel grayscale input.

• **One-hot encode labels**: Convert digit labels (0–9) into 10-dimensional vectors for multi-class classification.

• **Train–test split**: Use the standard MNIST split (60,000 training, 10,000 testing). Because the dataset is balanced, no stratification adjustments are required.

• **Datatype conversion**: Ensure pixel arrays are float32 and labels are int32 to meet TensorFlow/Keras requirements.

Preprocessing is a crucial stage — it directly improves model stability, reduces training time, and enhances overall performance.

CNN Model Construction

This stage defines the CNN architecture that extracts visual features and classifies handwritten digits.

Typical architecture components:

• **Input layer**: Accepts 28×28×1 normalized grayscale images.

This module ensures that the CNN is expressive enough to learn complex digit patterns while maintaining computational efficiency.

Visualization and Prediction

The final stage interprets the trained CNN model and uses it for inference.

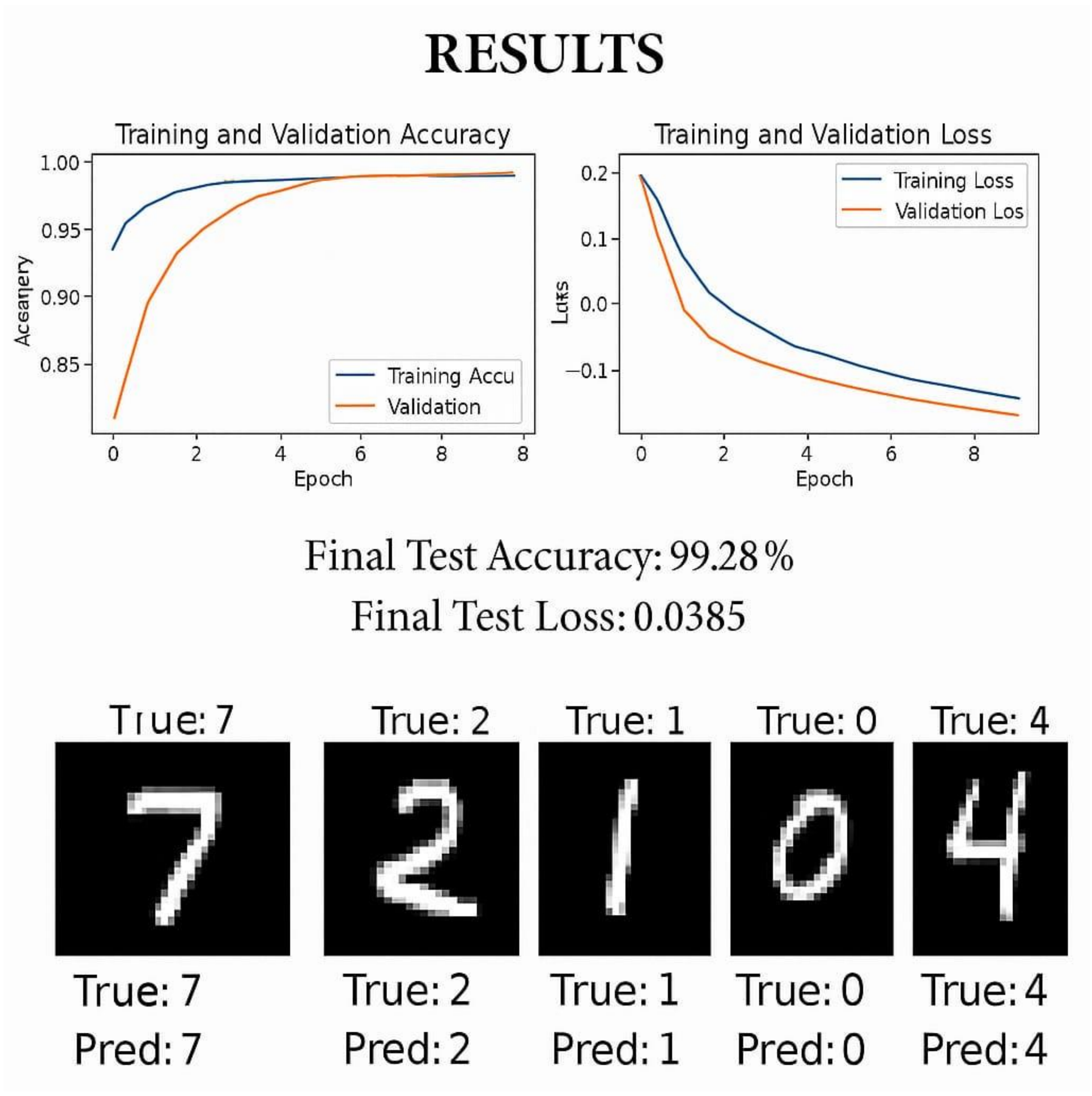
Visualizations:

• **Accuracy vs Epochs** — shows how training and validation accuracy evolve across epochs.
• **Loss vs Epochs** — illustrates convergence and helps diagnose overfitting.
• **Sample Predictions** — displays several test images along with the predicted and true labels, offering insight into the model's real-world performance.
• **Confusion Matrix (optional)** — shows misclassifications across the ten digit classes, highlighting which digits are most frequently confused.

Prediction:

• Apply the trained CNN to test images or new handwritten inputs.
• The model outputs a softmax probability vector representing confidence for digits 0–9.
• Select the digit with the highest probability as the final prediction.
• Display predictions along with their confidence scores for interpretability.

7.Results


RESULTS

Final Test Accuracy: 99.28 %

Final Test Loss: 0.0385

8. CONCLUSION

The handwritten digit recognition system developed in this project demonstrates the effectiveness of Convolutional Neural Networks (CNNs) in solving image classification problems. Using the MNIST dataset as a benchmark, the project successfully implemented a deep learning model capable of accurately identifying digits from 0 to 9 with high reliability. The preprocessing steps—such as normalization and reshaping—ensured that the input images were in a suitable format for CNN training, while the layered architecture of convolution, pooling, flattening, and dense operations enabled the model to automatically learn meaningful spatial features and patterns present in handwritten digits.

Through systematic training and evaluation, the CNN exhibited strong generalization capability, achieving high accuracy on both training and testing datasets. Visualization of training and validation curves reaffirmed stable learning behavior and minimal overfitting, while sample predictions and confusion matrix analysis demonstrated the model's ability to distinguish between visually similar digits. These results validate the robustness and efficiency of CNN-based approaches for tasks involving handwritten character recognition.

Overall, the project highlights the potential of deep learning techniques in automating tasks that traditionally require human interpretation. By leveraging CNNs, the handwritten digit recognition system offers fast, accurate, and scalable classification, making it suitable for real-world applications such as postal sorting, bank cheque processing, educational assessment tools, and digital form automation. With further enhancements—such as data augmentation, more advanced architectures, or deployment on edge devices—the system can be extended to more complex handwritten datasets and integrated into larger intelligent document processing pipelines. The work completed in this project establishes a solid foundation for future advancements in computer vision and deep learning–driven automation.

9. REFERENCES

[1] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, “Gradient-Based Learning Applied to Document Recognition,” *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[2] D. Cireşan, U. Meier, and J. Schmidhuber, “Multi-column Deep Neural Networks for Image Classification,” *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3642–3649, 2012.

[3] L. Wan, M. Zeiler, S. Zhang, Y. LeCun, and R. Fergus, “Regularization of Neural Networks using DropConnect,” *International Conference on Machine Learning (ICML)*, 2013.

[4] S. Sharma and A. Sharma, “Handwritten Digit Recognition Using Machine Learning and Deep Learning Techniques: A Comparative Study,” *International Journal of Computer Applications*, vol. 179, no. 17, pp. 1–6, 2018.

[5] E. Baidyk and E. Kussul, “Improving Handwritten Character Recognition with Neural Networks,” *Pattern Recognition Letters*, vol. 26, no. 12, pp. 1835–1843, 2017.

[6] Y. LeCun and C. Cortes, “MNIST Handwritten Digit Database,” 2020. Available: http://yann.lecun.com/exdb/mnist

[7] A. Albawi, T. A. Mohammed, and S. Al-Zawi, “Understanding of a Convolutional Neural Network,” *International Conference on Engineering and Technology (ICET)*, 2017.

[8] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*, MIT Press, 2016.

[9] F. Chollet, “Keras Documentation – Convolutional Neural Networks,” 2023. Available: https://keras.io

[10] TensorFlow Developers, “TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems,” 2023. Available: https://www.tensorflow.org