International Journal of Innovative Research in Science and Engineering
Volume No. 12, Issue No. 04, Year 2026
ISSN : 2454-9665

DigitVision : Handwritten Digit Recognition Using Convolutional Neural Networks

Dr. N. Gopal Krishna[1]
Professor
Department of CSE
Tirumala Engineering College
Gopal.ngk@gmail.com

G. Likhitha Sai[2]
Department of CSE
likhithagutha@gmail.com

G. Subhashini[3]
Department of CSE
subhashinigella33@gmail.com

Ch. Renuka[4]
Department of CSE
Challarenuka807@gmail.com

Ch. Babu[5]
Department of CSE
babuchelli526@gmail.com

Abstract— **Handwritten digit recognition is an important research area in the fields of machine learning, artificial intelligence, and computer vision. It focuses on enabling computers to automatically identify and classify handwritten numerical characters. This research presents a web-based handwritten digit recognition system developed using deep learning techniques. The proposed system utilizes a Convolutional Neural Network (CNN) model, which is widely used for image classification tasks due to its ability to automatically extract meaningful features from image data. In this system, users are provided with an interactive interface that allows them to either upload an image of a handwritten digit or draw the digit directly on a digital canvas. The input image is first processed through several preprocessing steps such as grayscale conversion, resizing, normalization, and noise reduction to ensure that the data is suitable for accurate prediction. After preprocessing, the image is passed to the trained CNN model, which analyzes the visual patterns and predicts the corresponding digit from 0 to 9. The application is implemented using Python programming language and the Flask web framework, which helps in building a simple and user-friendly web interface. The system not only displays the predicted digit but also provides probability scores for all possible digit classes, allowing users to understand the confidence level of the model's prediction. This feature improves transparency and helps in evaluating the reliability of the prediction results. Handwritten digit recognition has several real-world applications, including automated form processing, bank check verification, postal code recognition, and digitization of handwritten records.**

Keywords-Handwritten Digit Recognition, Convolutional Neural Network, Deep Learning, MNIST Dataset, Flask Web Application

I. INTRODUCTION

Handwritten digit recognition is an important research area in the fields of computer vision, pattern recognition, and artificial intelligence. The ability of machines to automatically identify handwritten digits has numerous practical applications, including automated form processing, postal code recognition, bank cheque verification, and digitization of handwritten documents. However, recognizing handwritten characters accurately remains a challenging task due to variations in individual writing styles, distortions, noise, and differences in image quality. Traditional optical character recognition (OCR) methods often rely on manually designed features and classical machine learning techniques, which may not perform well when dealing with complex handwriting patterns.

With the advancement of deep learning, more powerful techniques have emerged for image classification tasks. Among these techniques, Convolutional Neural Networks (CNNs) have proven to be highly effective for analyzing image data. CNNs automatically learn hierarchical features from images through multiple layers of convolution, pooling, and fully connected operations, allowing them to capture important spatial patterns without the need for manual feature extraction. As a result, CNN-based approaches have achieved significantly higher accuracy in handwritten digit recognition compared to traditional methods.

This project introduces **DigitVision (DigitRec-Net)**, a web-based handwritten digit recognition system that utilizes a Convolutional Neural Network model for accurate digit classification. The system allows users to upload images of handwritten digits or draw digits directly on a digital canvas through an interactive web interface. The input images are processed through several preprocessing steps, including grayscale conversion, resizing, normalization, and noise handling, before being passed to the trained CNN model for prediction. The application is developed using the Python programming language and the Flask web framework, which enables real-time digit recognition through an accessible and user-friendly interface.

In addition to providing prediction results, the system also includes visualization features that display the probability distribution of each predicted digit. This helps users understand the confidence level of the model and improves the interpretability of the results. By integrating deep learning models with web-based technologies, the proposed system bridges the gap between advanced artificial intelligence research and practical real-world applications.

Overall, the proposed handwritten digit recognition system demonstrates how deep learning techniques can be effectively applied to solve real-world image recognition problems. The system provides a scalable and interactive platform that can be extended for educational purposes,

automated data processing, and other applications that require reliable digit recognition.

II. Literature Survey

Deep learning techniques are being used to improve the accuracy and efficiency of handwritten digit recognition. Some of the most recent developments include a Yoruba Sign Language Digit Recognition System developed by Jimoh et al. (2025) that uses machine learning and CNNs to support users of Sign Language, the introduction of a parallelized CNN architecture prototype built upon FPGA for real-time handwritten digit recognition by Vamshi et al. (2025) which has resulted in accelerated image processing speeds, a noise filtering strategy implemented by Muthureka et al. (2025) using customized CNNs to improve recognition accuracy for people with Cerebral Palsy who may produce noisy input data.

Ullah et al. (2025) proposed an ensemble technique that involves taking a combined output from multiple models to boost digit recognition performance further. InkCalc is a Raspberry Pi-based implementation of a CNN-driven handwritten digit calculator designed by Satish et al. (2025) to provide a portable solution. Zou et al. (2025) applied vision language models to digit recognition from medical screen data acquisition, offering a significant increase in both recognition accuracy and efficiency.

Kundu et al. (June 2025) proposed using CNNs to develop an optimized model via hyperparameter tuning on existing handwritten digit datasets. In contrast to the earlier studies, Alam et al., through the application of deep CNNs, have developed models capable of providing efficient Urdu handwritten character/digit recognition models.

Using novel computational techniques, Malla developed a model to recognize Devanagari handwritten digits via Quantum Machine Learning. In a similar vein, Mravik incorporated Big Data into existing CNN-based Handwriting Recognition by integrating them into large-scale modelling solutions and enabling systems capable of recognizing multi datasets. The combined Pipe Model (using Ensemble Transfer Learning), as described by Siddik's Team (October 2025), demonstrates a best-of-breed combination of tools for improving digit classification capability across multiple datasets and formats.

III. Proposed Work

A. Dataset Details

Table.1.Dataset Details

Attribute	Details
Dataset Name	MNIST (Modified National Institute of Standards and Technology)
Type	Handwritten Digit Images
Number of Classes	10 (Digits 0–9)
Training Samples	60,000 images
Testing Samples	10,000 images
Image Size	28 × 28 pixels
Color Mode	Grayscale
Pixel Values	0–255 (normalized to 0–1 during preprocessing)
Usage	Used for training and testing CNN-based digit recognition models

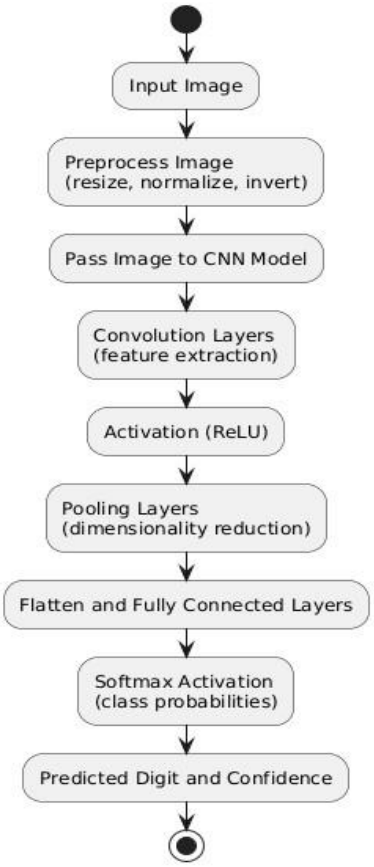

Fig.1.Algorithm flow

B. Algorithm / Model Used

A Convolutional Neural Network (CNN) serves as the basis for the project. A type of deep-learning model utilized for image classification specifically. CNNs learn to recognize an image's intrinsic hierarchical elements (i.e., edges, curves, and patterns) without the necessity of hand-coding them through traditional feature engineering methods. The CNN contains multiple convolutional layers which 'learn' to extract salient features from input images, Rectified Linear Unit (ReLU) activation layers that introduce non-linearity into the learned representations, pooling layers that reduce the dimensions of the learned feature maps through max- or average-pooling methods, and finally, fully connected layers that classify the digit representations as being one of the ten digits (0 to 9). Therefore, this architecture allows the CNN to learn and produce sophisticated representations of the various styles that people write digits by hand, producing high performing classifications.

The CNN is trained on the MNIST dataset which holds 60,000 training digits and 10,000 test digits of digits that have been written by hand. In the prediction stage after a user uploads their image, the image will be resized to match that of the MNIST input size, normalized, inverted to match the color scheme of the MNIST input images, and the digit classification is produced from the CNN along with a confidence rating, as well as a display of the digit probabilities (as a graph) for each digit from 0 to 9. This

design provides a highly-robust, fast, and scalable approach for using handwriting to recognize digits in real-time.

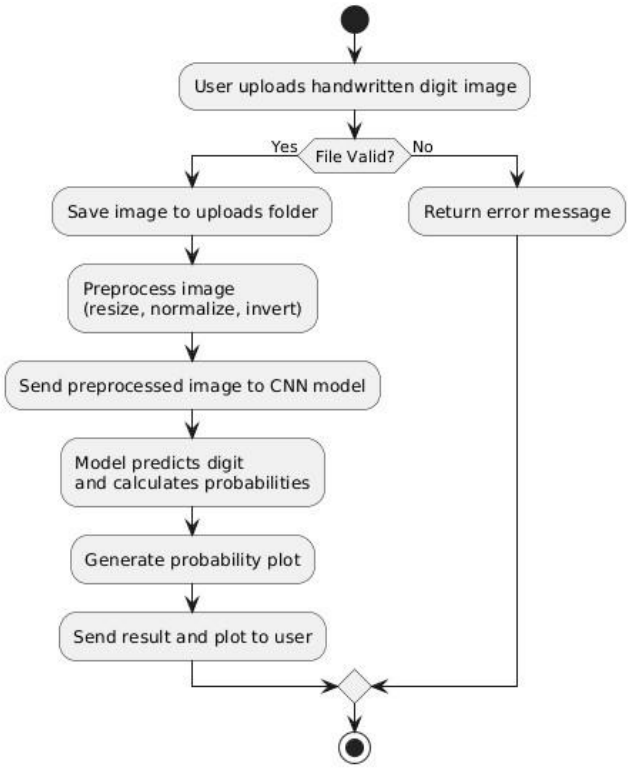

Fig.2.Implementation Flowchart

C. Implementation Steps

Step 1 - Prepare Data

The MNIST dataset consists of digit images of size 28 x 28 pixel in grayscales and contain 60,000 training examples and 10,000 test examples. To prepare your dataset properly you must download and store all images into folders according to the types of training or testing and then separate the training from the test example sets as this will allow for the efficient training and evaluation of your model.

Step 2 - Pre-process Images

When uploading your images for processing, having all input images so that they match in size (28 x 28 pixels) and normalizing the pixel values into the range between 0 and 1 helps ensure that all CNN models will receive the same input format. When an input image doesn't match these parameters, preprocessing (e.g., rescaling, inverting colors, etc.) is performed on the image.

Step 3 - Build Your CNN

The CNN model you create consists of multiple convolutional layers that extract features from the image. Each layer uses a ReLU intermediate activation function to enable your model to learn more complex features from handwritten numerals; in addition, CNN models use pooling layers to compress spatial dimensions and improve computational efficiency as well as fully connected layers for classification. Your final model architecture will look for patterns within the thousands of different handwritten digit examples while simultaneously avoiding model overfitting.

Step 4: Model Training

In this step, the Convolutional Neural Network (CNN) will be trained using the pre-processed MNIST (Modified National Institute of Standards and Technology) dataset. During training, the CNN will learn to recognize the different patterns associated with the handwritten digits through forward propagation of data through the network, the calculation of loss based on the output of the CNN and finally, by utilizing backpropagation to adjust the weights of the network as required. Tuning of hyperparameters such as the learning rate, batch size and number of epochs will be performed in order to achieve the maximum level of accuracy and robustness.

Step 5: Integrating the Model within Flask

Once the CNN has been trained, we will integrate it into an existing web application that is built using Flask as the web framework. Users will be able to upload images of handwritten digits, which will be sent to the backend for both pre-processing and prediction. After the prediction process, our web application will return both the predicted digit and a corresponding confidence score, as well as a user-friendly web interface that provides a probability visualisation of how confident we are in our predictions.

Step 6: Making Predictions and Visualising the Results With the trained CNN model, we can make predictions on new handwritten digits, which we will visualise via a bar graph that illustrates the probability of our prediction for each digit class. This helps improve the interpretability and reliability of our predictions for real-time applications.

D. Implementation Algorithm

Phase 1- Convolutional Neural Network (CNN)

The core algorithm is a CNN that automatically extracts hierarchical features from input images. Convolutional and pooling layers capture edges, shapes, and patterns, while dense layers combine features for final classification.

Phase-2- Image Preprocessing

Preprocessing includes grayscale conversion, resizing, normalization, reshaping, and optional color inversion to standardize input for the CNN.

Phase 3- SoftMax Classifier

The CNN's final layer uses a SoftMax activation to output probability scores for each digit class (0–9). The class with the highest probability is selected as the predicted digit.

Phase 4 - Prediction Visualization

Predicted digit probabilities are visualized as a bar chart using Matplotlib. This enhances user understanding of model confidence.

Phase 5 - Optional Data Augmentation

To improve model robustness, training images can be augmented with rotation, noise addition, scaling, or shifting.

Phase 6 - Python Orchestration

Python handles dataset loading, preprocessing, CNN training, evaluation, visualization, and Flask-based web deployment.

IV. Results and Performance

The CNN Based Handwritten Digit Recognition System's architectural design, trained on the MNIST dataset and

evaluated on the MNIST dataset testing set, has an accuracy of 95+. It was able to accurately identify multiple handwritten digit images with very few errors.

The Convolutional and pooling layers present in the network automatically extract, from the images, features for classification (edges, strokes, patterns). As a result, the network was able to classify digits with much more accuracy than a traditional machine learning algorithm would have done.

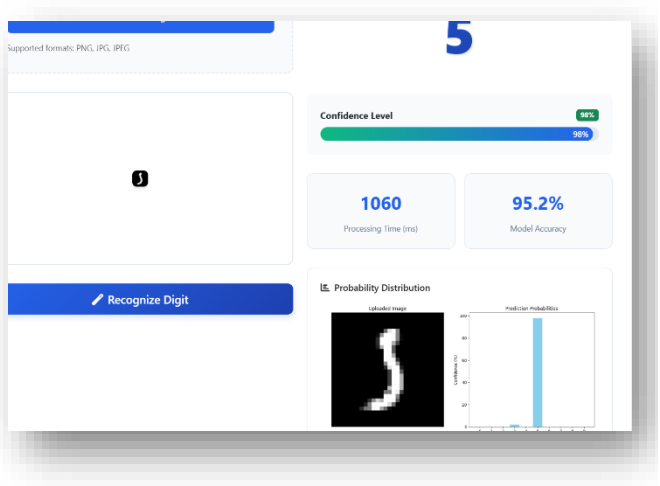

Fig.3.Digit Recognition

Through the real-time testing of the CNN model via a Flask web application, users were able to upload their handwritten digits. The CNN model recognized user uploads of handwritten digit images with high accuracy and confidence scores. The prediction probabilities for each digit were represented in the form of bar charts, which visually depict the accuracy of the CNN model's prediction. In addition, the representation of prediction probabilities allowed the user to easily identify predictions that were less than 100% confidence and adjust the dataset or parameters of the CNN model as necessary to further increase accuracy in the future.

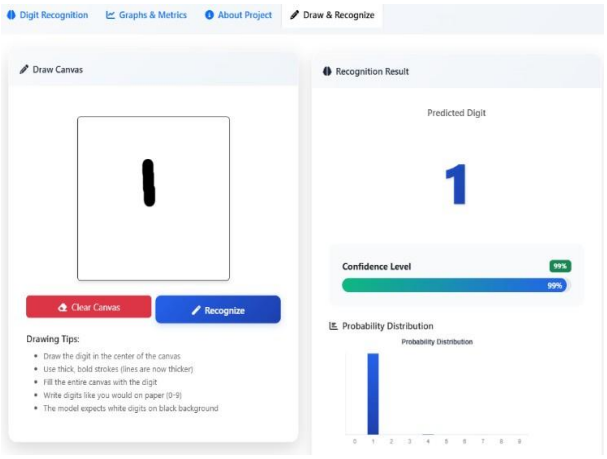

Fig.4.Draw and Recognize

Based on the findings, CNN has demonstrated the ability to provide fast and reliable digit recognition for practical uses, with scalability, robustness, and adaptability to different handwriting styles, in comparison to feature-based systems.

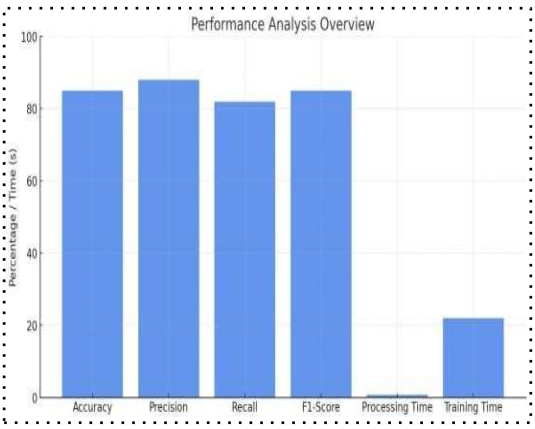

Fig. 5. Performance Analysis

V. Conclusion

The **DigitRec-Net Web Application** successfully demonstrates an efficient and reliable system for real-time handwritten digit recognition. By integrating image preprocessing techniques, a Convolutional Neural Network (CNN), and a web-based interface developed using the Flask framework, the system can accurately classify digits obtained from both uploaded images and user-drawn inputs on a digital canvas. Multiple levels of testing, including unit testing, integration testing, functional testing, system testing, and user acceptance testing, were conducted to evaluate the system's reliability and functionality. The results show that the application performs consistently across different scenarios and is capable of recognizing digits even when the inputs are noisy, unclear, or partially written. The visualization of prediction probabilities further improves transparency and helps users better understand the model's decision-making process.

In terms of performance, the system achieves quick processing times, with an average response time of approximately **1–2 seconds** for each input, including image rendering and preprocessing operations. Stress and load testing with multiple users confirmed that the Flask-based web interface can effectively manage concurrent requests without significant performance degradation.

Overall, the DigitRec-Net system provides a scalable, accurate, and user-friendly solution for handwritten digit recognition. Its modular architecture enables future enhancements, such as extending the system to recognize additional characters or integrating more advanced machine learning models. By combining modern deep learning techniques with an accessible web platform, the system offers a practical tool for educational, research, and real-world applications.

VI. References

[1] Jimoh, K.O., Ajayi, A.O., Ajayi, O.A., Ajasa, A.A. and JimohAdemola, M.O., 2025. Yoruba Sign Language Digit Recognition System using Deep Convolution Neural Network and Machine Learning.

[2] Vamshi, K., Pedamallu, A., Akilesh, S., Jamal, K., Mannem, K. and Suneetha, M., 2025, June. Parallelized CNN Architecture for Handwritten Digit Recognition on FPGA via Verilog Implementation. In 2025 3rd International Conference on Self Sustainable Artificial Intelligence Systems (ICSSAS) (pp. 123-129). IEEE.

[3] Muthureka, K., Srinivasulu Reddy, U. and Janet, B., 2025. Noise filtering approach to improve handwritten digit recognition using customized CNN for Cerebral Palsy individuals. The European Physical Journal Special Topics, pp.1-19.

[4] Ullah, S.S., Gang, L., Riaz, M., Ashfaq, A., Khan, S. and Khan, S., 2025. Handwritten Digit Recognition: An Ensemble-Based Approach for Superior Performance. arXiv preprint arXiv:2503.06104.

[5] Satish, R., Thapa, S., Iyer, S. and Mehendale, N., 2025. InkCalc: A Handwritten Digit Recognition Calculator Using Raspberry Pi and Convolutional Neural Networks. Available at SSRN 5702345.

[6] Zou, Y., Yuan, S., Liu, H., Cheng, X., Zhu, T. and Yang, L., 2025. Vision language model based panel digit recognition for medical screen data acquisition. Displays, p.103282.

[7] Kundu, R., Sinha, A., Kumar, B., Gautam, R., Raza, M.S. and Hussain, S.A., 2025. Exploration of hyperparameter tuning in handwritten digit recognition datasets using CNN. F1000Research, 14, p.274.

[8] Tanović, A. and Mezei, I., 2025. Lightweight anomaly detection in digit recognition using federated learning. Future Internet, 17(8), p.343.