IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 4, January 2026
Impact Factor: 7.67

Deep Learning Based Handwritten Digit Recognition

Dr. Shiva Kumar V[1] and M Raghavendra[2]
Associate Professor, Department of Computer Science and Engineering, Kishkinda University, Ballari, India [1]
PG Student, Department of Computer Science and Engineering, Department, Kishkinda University, Ballari, India [2]

Abstract: *As computers play an increasingly vital role in human life and daily activities across various domains, humans have leveraged their intelligence and creativity to use computers in natural and effective ways. Hence, a reliable method for recognizing handwritten digits is essential. Handwritten Digit Recognition (HDR) can offer a clear benefit in this aspect. Deep Learning (DL) has been a powerful tool for solving various problems with high accuracy in recent years. This surveys the different methods for HDR that have been developed by various researchers. Machine learning has enriched this analysis with different approaches that involve supervised learning, unsupervised learning and reinforcement learning. Next, this project reviews the applications of deep learning methods to different languages in real-world scenarios. DL techniques are specially designed for handling complex data formats. Many natures inspired Convolutional Neural Network (CNN) models are discussed in this section.*

Keywords: Handwritten Digit Recognition, Handwritten Text Recognition, Pattern Recognition, Computer Vision, Image Processing, Deep Learning, Convolutional Neural Network (CNN), Neural Networks, Feature Extraction, Classification, MNIST Dataset, Image Preprocessing, Data Augmentation, Noise Removal, Segmentation, Real-Time Digit Recognition, Digital Whiteboard, Camera-Based Input, GUI, Text-to-Speech, Accuracy, Training Loss, Validation Loss, Robustness, Unit Testing

I. INTRODUCTION

The main objective of this project is to design a system capable of recognizing handwritten digits and converting them into a format that machines can process. This field, known as Handwritten Digit Recognition, has gained significant importance due to its applications in areas such as postal mail sorting, check processing in banks, and automated form digitization. The approach leverages real-time digit recognition, which involves processing images or live input data to identify digits accurately and efficiently. Over the years, researchers have focused on enhancing the accuracy and efficiency of such systems. They have employed various machine learning and deep learning techniques, including convolutional neural networks (CNNs), to improve performance.

Despite these advancements, handwritten digit recognition remains a challenging problem. This complexity stems from the inherent variability in human handwriting. Each individual has a unique style, and the system must handle a wide range of writing patterns, including cursive or stylized digits. Furthermore, factors such as image quality play a significant role in the system's accuracy. Low-resolution images, improper lighting, or background noise can complicate the digit recognition process. Addressing these challenges requires sophisticated preprocessing techniques, robust feature extraction methods, and accurate classification algorithms. Through continuous innovation and research, the field of handwritten digit recognition has evolved, making systems more reliable and versatile in real-world applications.

II. OBJECTIVE OF THE PROJECT

To improve this system takes an image of a handwritten digit on a digital whiteboard or under a camera as input and correctly identifies the digit using a Convolutional Neural Network (CNN).

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429 Volume 6, Issue 4, January 2026 Impact Factor: 7.67

To enhance network is trained using the MNIST dataset, achieving 97% accuracy on the test set and a training loss of 0.1.

To achieve good results during the real time testing..

III. LITERATURE SURVEY

TITLE: Hybrid Hidden Markov Model and Artificial Neural Network Approach for Offline Handwritten Text Recognition

Hybrid Hidden Markov Model and Artificial Neural Network Approach for Offline Handwritten Text Recognition proposes a hybrid framework that combines Hidden Markov Models for sequential handwriting modeling with Artificial Neural Networks for feature classification. This integration improves recognition accuracy by effectively handling variations in handwriting styles and character structures. The approach establishes a strong foundational methodology for offline handwritten text recognition. However, it relies on handcrafted features and traditional neural architectures, limiting scalability and performance compared to modern deep learning-based methods.

TITLE: Gated-CNN-BGRU Architecture for Handwritten Digit String Recognition under Noisy Conditions

Gated-CNN-BGRU Architecture for Handwritten Digit String Recognition under Noisy Conditions presents a deep learning–based framework that integrates gated convolutional neural networks with bidirectional gated recurrent units to improve handwritten digit string recognition. The CNN component extracts robust spatial features, while the BGRU layer effectively models sequential dependencies within digit strings. The architecture demonstrates significant improvements in recognition accuracy, particularly in noisy environments and scenarios with limited training data. However, the model's computational complexity and focus on digit-only recognition restrict its applicability to broader handwritten text recognition tasks.

TITLE: Deep Learning-Based Handwritten Digit String Recognition Using Gated CNN and Bidirectional GRU

Deep Learning-Based Handwritten Digit String Recognition Using Gated CNN and Bidirectional GRU focuses on improving the accuracy and robustness of handwritten digit string recognition within the domain of pattern recognition and computer vision. The study employs a deep learning framework that integrates gated convolutional neural networks for effective spatial feature extraction with bidirectional gated recurrent units to model sequential dependencies in digit strings. This architecture enables the system to handle variations in handwriting styles and noise, leading to significant improvements in recognition performance, particularly in challenging and low-data environments. The main advantage of the approach lies in its ability to jointly learn spatial and temporal representations in an end-to-end manner. However, the model is primarily designed for digit strings and does not readily generalize to full handwritten text or multilingual recognition tasks. Nevertheless, the framework provides a strong foundation for extending deep learning-based sequence recognition techniques to broader handwriting and document analysis applications.

TITLE: A Comprehensive Review on Internet of Things Applications in Power Systems

Majhi, Abhilash Asit Kumar, and Sanjeeb Mohanty, in the IEEE Internet of Things Journal (2024), review IoT applications in power systems, focusing on smart grids, renewable integration, demand management, and reliability. The study highlights how IoT enables real-time monitoring, predictive analytics, and intelligent control for efficient power operation. While comprehensive, it is generalized for power systems and does not directly address EV charging. Its insights, however, provide a strong basis for extending IoT into EV charging infrastructure, enabling real-time monitoring, predictive maintenance, and optimized energy use.

Table I: **Literature Survey Review**

Paper Title / Source	Key Findings / Work Done	Limitations	Proposed Upgradation in Our Project
Hybrid HMM–ANN Based Offline Handwritten Text Recognition (Gao et al., ACM TOMM, 2020)	Proposed a hybrid approach combining Hidden Markov Models with Artificial Neural Networks to improve offline handwritten text recognition	Complex model architecture; high computational cost; limited scalability for large datasets.	Replace HMM–ANN hybrid with an end-to-end deep learning model (CNN + RNN) for faster training and improved scalability.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429 Volume 6, Issue 4, January 2026 Impact Factor: 7.67

Paper Title / Source	Key Findings / Work Done	Limitations	Proposed Upgradation in Our Project
	accuracy.		
Gated CNN–BGRU for Handwritten Digit String Recognition (Gao et al., IEEE IoT Journal, 2019)	Introduced a Gated-CNN and Bidirectional GRU architecture adapted from HTR to improve digit string recognition, especially in noisy environments.	Requires careful hyperparameter tuning; performance drops with highly cursive writing styles.	Extend the architecture with attention mechanisms and multilingual digit support (English + Kannada).
Review on Digital Image Processing Techniques for Handwritten Content (Herrera-Pereda et al., Medical Image Analysis, 2021)	Surveyed image processing and feature extraction techniques for handwritten character recognition applications.	Mainly theoretical review; no real-time or end-user application discussed.	Implement selected preprocessing techniques (noise removal, normalization) in a real-time digit recognition system.

Summary:

The reviewed literature highlights significant advancements in handwritten text and digit recognition through the use of hybrid models, deep learning architectures, and image processing techniques. Early works demonstrate the effectiveness of combining probabilistic models such as Hidden Markov Models with Artificial Neural Networks to improve offline handwriting recognition, though at the cost of increased computational complexity. Recent studies emphasize deep learning approaches, particularly CNN- and RNN-based architectures such as Gated CNN–BGRU models, which show improved robustness in noisy environments and limited training scenarios. Several works focus on handwritten text detection in real-world settings such as whiteboards, lecture videos, and medical documents, showcasing the importance of preprocessing, segmentation, and feature extraction for accurate recognition. However, many existing systems are limited to single-language datasets, lack real-time implementation, or focus only on detection without integrated recognition. Based on these gaps, the proposed project aims to develop a real-time, end-to-end handwritten digit recognition system supporting multilingual (English and Kannada) digits with camera-based input, deep learning-based recognition, and speech output, thereby extending existing research toward practical and user-centric applications.

IV. PROBLEM STATEMENT

To design and develop a real-time handwritten digit recognition system that uses a Convolutional Neural Network (CNN) to accurately identify digits (0–9) captured through a live camera or drawing interface. The proposed module is intended to acquire handwritten digit images, preprocess them to enhance clarity, extract relevant features using a deep learning model, and classify the digits with high accuracy. Additionally, the system will provide interactive feedback by announcing the recognized digit through an AI-based voice assistant, thereby improving usability, accessibility, and overall user experience.

V. METHODOLOGY

This section presents a methodology for optimizing Handwritten Digits Recognition for CNN Algorithm. The main aim of the system is to optimize the numerical number identification throughout with the digital white board and real time camera to capturing the identified number with accuracy. Initially, the system utilizes Graphical User Interface System (GUI) data that is in the form of MNIST files containing the numerical single numbers distribution of images size of 28*28. Further, CNN model is employed to group these numbers to identified, representing different styles of written numbers to recognizing. Subsequently, the CNN Algorithm is applied to Image recognition and processing for written

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 4, January 2026
Impact Factor: 7.67

number in white board and real time camera within each model, by considering Recognizing capacity limitations. The proposed methodology aims to improve the effectiveness of identifying numbers, leading to improved accuracy, reduced risk consumption, and lower emissions.

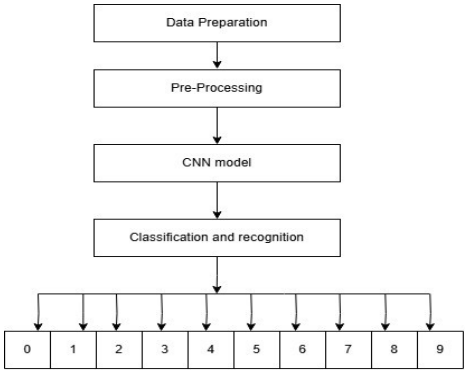

Figure 5.1:- Block diagram for Handwritten Digit Recognition Using a CNN Model methodology

CNN (Convolutional Neural Network)

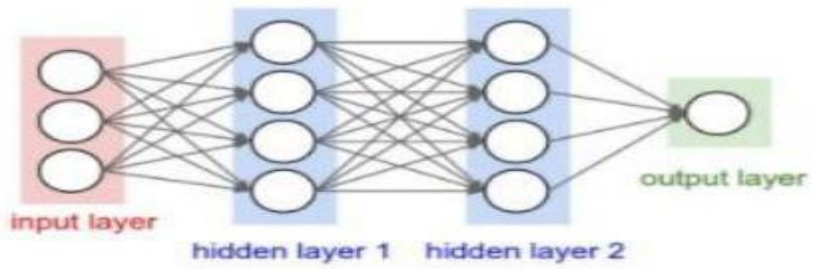

Figure 5.1. 1:- Architecture of Neural network

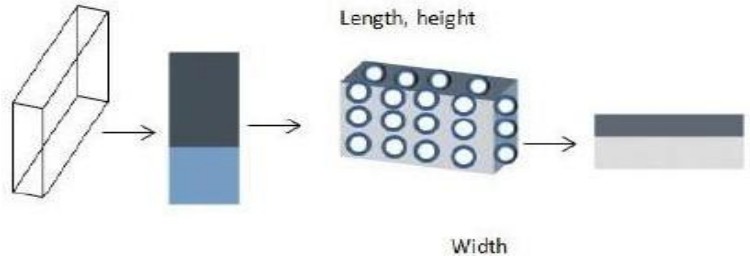

Figure 5.1. 2:- Architecture of Convolution neural network

Convolution in neural networks refers to a mathematical operation used to extract meaningful features from input data, making Convolutional Neural Networks (CNNs) powerful yet computationally complex due to their multiple layered structure. A CNN consists of several layers where each layer performs a dot product operation between input pixel values and learned weights, and the resulting feature maps are passed to subsequent layers for further processing. The convolution layers are typically followed by activation functions and pooling layers, and finally by fully connected layers with a softmax function for classification. There are three primary types of layers in a CNN: the convolution layer, which extracts local features using filters; the pooling layer, which reduces spatial dimensions and computational complexity; and the fully connected layer, which performs final classification based on extracted features. The input to a CNN is usually an image of size $d \times d \times n$, where d represents the height and width of the image and n denotes the number of channels (for example, n=3 n=3 for RGB images). Each convolution layer applies multiple filters of size $f \times f \times g$, where f is smaller than the input dimension and g corresponds to the depth of the input. The development of such layered recognition systems can be traced back to early pattern recognition approaches, including the analysis-by-

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 4, January 2026
Impact Factor: 7.67

synthesis technique proposed by Eden in 1968, which laid the foundation for modern neural network-based image recognition methods.

VI. REQUIREMENTS

FUNCTIONAL REQUIREMENTS

- Data Collection: Collect a comprehensive dataset of handwritten digits, including various styles and sources (e.g., MNIST). Ensure each image is correctly labeled for training.
- Preprocessing: Normalize images for uniformity (resize to 28x28, convert to grayscale). Employ data augmentation (e.g., rotation, shifting) to improve model generalization.
- Digit Recognition: Implement a user interface (UI) for inputting images of handwritten digits. Use deep learning algorithms to recognize and classify digits, providing output predictions.
- Training Model: Design and train a CNN model optimized for handwritten digit recognition. Utilize techniques such as batch normalization and dropout to enhance model performance.

NON-FUNCTIONAL REQUIREMENTS.

- **Accuracy:** Target a minimum accuracy of 98% on the test dataset after training and optimization. Continuously update the model based on new data to maintain high accuracy levels.
- **Scalability**: Ensure the architecture can handle an increase in data volume and user requests without performance loss. Prepare the system for integration with cloud- based solutions for data storage and processing power.
- **Performance:** Aim for less than 1 second latency for digit recognition after model deployment. Utilize efficient algorithms and hardware acceleration (e.g., GPU) to optimize processing time.
- **Robustness**: Create a model resilient to variations in writing styles, ink blots, and noise. Implement logging and error-handling mechanisms to identify and resolve issues promptly, ensuring consistent performance.

VII. DESIGN

The Design consists of a use case diagram, sequence diagram, data flow diagram, and many more as follows.

DATA FLOW DIAGRAM

A Data Flow Diagram (DFD) is a traditional visual representation of the information flows Within a system. A neat and clear DFD can depict the right amount of the system requirement graphically. It can be manual, automated, or a combination of both. It shows how data enters and leaves the system, what changes the information, and where data is stored.

The objective of a DFD is to show the scope and boundaries of a system as a whole. It may be used as a communication tool between a system analyst and any person who plays a part in the order that acts as a starting point for redesigning a system. The DFD is also called as a data flow graph or bubble chart.

The symbols depict the four components of the data flow diagram:

External entity:

An outside system that sends or receives data, communicating with the system being diagrammed. They are the sources and destinations of information entering or leaving the system. They might be an outside organization or person, a computer system, or a business system. They are also known as terminators, sources and sinks, or actors. They are typically drawn on the edges of the diagram.

Process:

The process that changes the data, producing an output. It might perform computations, or sort data based on logic, or direct the data flow based on business rules.

Data store:

The files or repositories that hold information for later use, such as a database Image or a JPG form.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 4, January 2026
Impact Factor: 7.67

Data flow:

The white board or real time camera that data takes between the external entities, processes, and data stores. It portrays the interface between the other components and is shown with arrows, typically labeled with a short data name, like "Capturing details". Levels in DFD are numbered 0, 1, 2 or beyond.

Level-0 DFD:

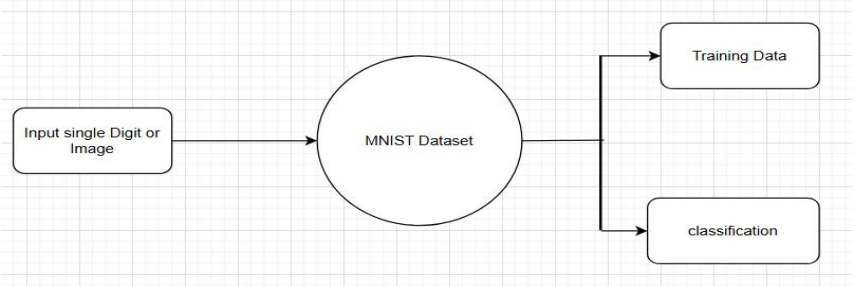

Figure 7. 1:- Level-0 Data Flow Diagram

Level-1 DFD:

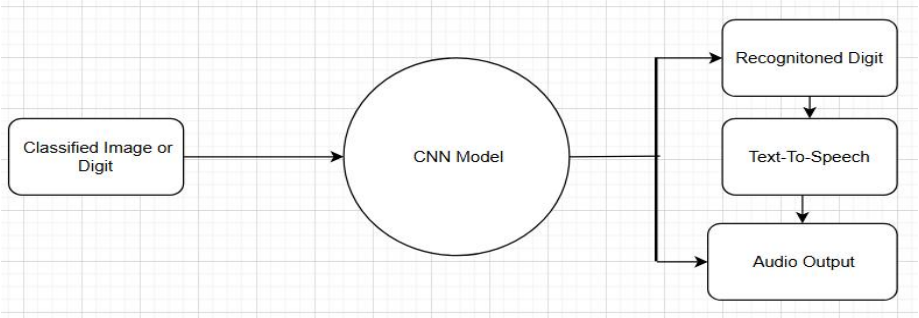

Figure 7. 2:- Level-1 Data Flow Diagram

It is also known as a context diagram. It's designed to be an abstract view, showing the system as a single process with its relationship to external entities. It represents the entire system as a single bubble with input and output data indicated by incoming/outgoing arrows.

The data flow diagram shows a recognizing numbers model with two main user type: white board and real time camera system. The white board is responsible for drawing numbers on board with the red and black color with accuracy of digit announcing digit with white board, AI voice and training and testing of digit is done by CNN model with using MNIST data processing by image cleaning of the deep learning system. This setup ensures that users have access to efficient identified numbers of different styles of written digits, while the easy to capturing the numbers of user to writing number for on white board and real time camera access or maintains the system's operational data.

SEQUENCE DIAGRAM

Sequence models, which show the sequence of object interactions. These are represented using a UML sequence or a collaboration diagram. Sequence models are dynamic models. Sequence models are dynamic models that describe, for each mode of interaction, the sequence of object interactions that take place. When documenting a design, you should produce a sequence model for each significant interaction. If you have developed a use case model then there should be a sequence model for each use case that you have identified.

The sequence diagram illustrates the interaction between deep learning model based handwritten digit recognition of numerical number collection of white board, real time camera and a CNN algorithm to classifying the written digit for the inputting the digit image. Recognizing system for initiate the process by requesting Data preparation from the MNIST data. The data, in turn, employs a process of capturing the digit in the image form of CNN algorithm to calculate the most efficient patterns. These optimized patterns are then sent back to preprocessing. Following this, the

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 4, January 2026
Impact Factor: 7.67

digit recognition from designated collection points according to the provided optimized data patterns. This process done by saving of the time by identifying the number. It easy and secure of the system.

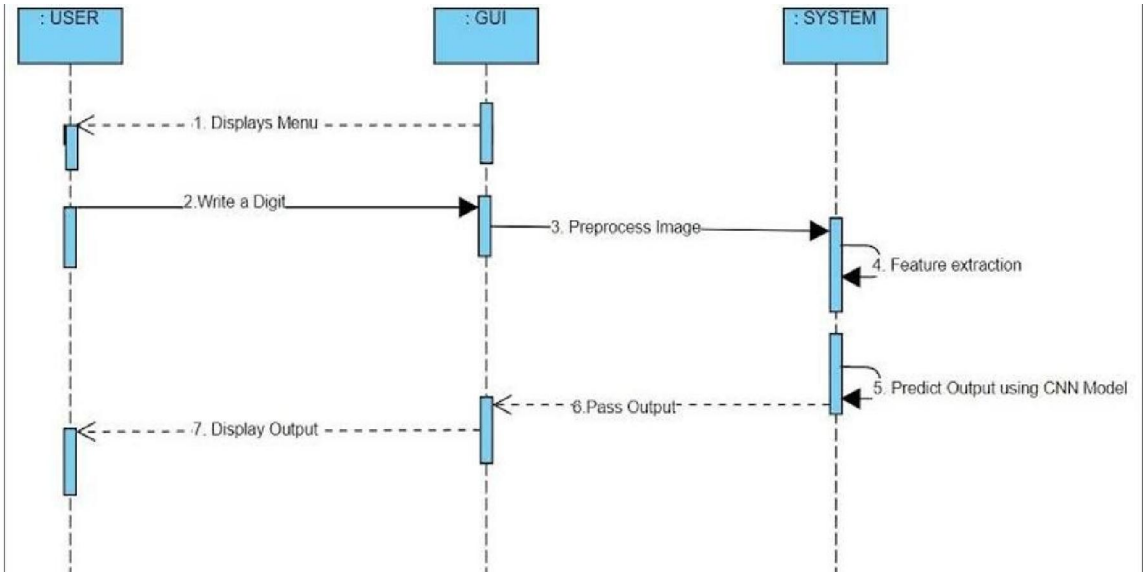

Figure 7. 3:- Sequence diagram

USE CASE DIAGRAM

In the Unified Modeling Language (UML), a use case diagram can summarize the details of your system's users (also known as actors) and their interactions with the system.

An effective use case diagram can help your team discuss and represent:

Scenarios in which your system or application interacts with people, organizations, or external systems

Goals that your system or application helps those entities (known as actors) achieve the scope of the system

Common components include:

Actors: The users that interact with a system. An actor can be a person, an organization, or an outside system that interacts with your application or system. They must be external objects that produce or consume data.

System: A specific sequence of actions and interactions between actors and the system. A system may also be referred to as a scenario.

Goals: The end result of most use cases. A successful diagram should describe the activities and variants used to reach the goal.

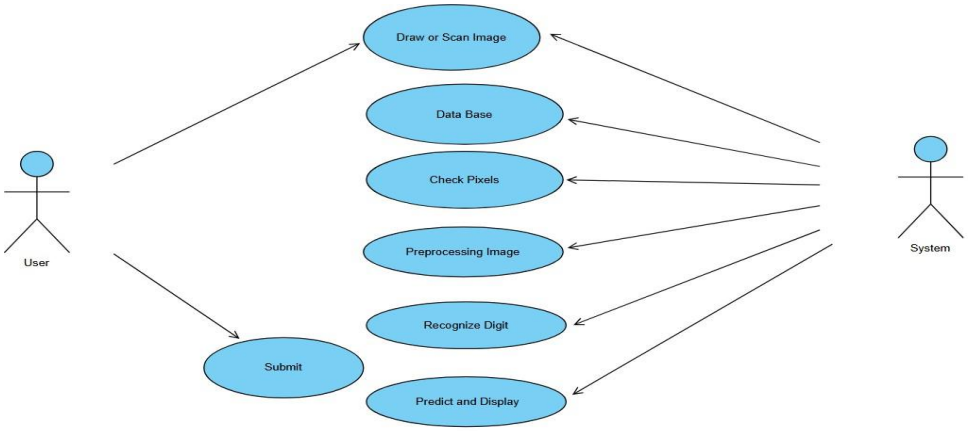

Figure 7. 4:- Use Case Diagram

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429 Volume 6, Issue 4, January 2026 Impact Factor: 7.67

The use case diagram shows the interactions between two actors, a User, and an System, with a Handwritten Digit Recognition prosess. The User can Draw the digit on the Digital White Board and Access the Real Time camer for preprocessing the input into the image format. The System has the capability to recognizing and predicting the output with the accuracy. The predicting the after output of the image digit by using AiI voice to announcing the identified output.

VIII. IMPLEMENTATION

MODULES

Data Preprocessing, Model Architecture, Training the Module, Testing the Module

MODULE DESCRIPTION

DATA LOADER

Role: Efficiently load data in batches for training and testing.

Features: Shuffles the dataset to ensure randomization and prevent learning order-specific biases. Loads batches asynchronously to minimize latency during training.

MODEL ARCHITECTURE

Role: Custom implementation of the CNN model. Highlights: Built using PyTorch's torch.nn module.

Encapsulates the network architecture and forward propagation logic. Pen Color Selection: Allow users to select black or red for drawing. Sound Feedback: Use pyttsx3 for text-to-speech feedback on predictions.

Camera Integration: Capture frames from the webcam, preprocess the region of interest (ROI), and predict digits in real-time.

TRAING LOOP

Role: Handles the iterative process of weight optimization.

Features: Tracks training metrics like loss and accuracy after each epoch. Optionally includes validation on a subset of the training data for early stopping.

TESTING LOOP

Role: Measures the model's real-world performance.

Features: Visualizes sample predictions (e.g., correct and incorrect classifications). Outputs a detailed performance report (accuracy, precision, recall, F1-score).

TOOLS USED

Hardware:

GPU: Recommended for faster training, especially for large datasets. CPU: Sufficient for small-scale tasks like MNIST.

Storage: A system capable of storing the dataset and model checkpoints (e.g., 10GB free space).

Software: PyTorch for deep learning implementation. Python environment for scripting and debugging.

PROGRAMMING LANGUAGES

Python: Python is chosen due to its rich ecosystem for machine learning and simplicity in handling tensors and neural networks.

TECHNOLOGIES, LIBRARIES, AND FRAMEWORKS

Deep Learning: Uses neural networks to automatically extract features and classify data.

Convolutional Neural Networks (CNNs): Specially designed to handle image data by exploiting local spatial structures.

Backpropagation and Gradient Descent: Mathematical algorithms to compute and propagate weight updates.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429 Volume 6, Issue 4, January 2026 Impact Factor: 7.67

PyTorch: Tensor computation and automatic differentiation for deep learning tasks.
Torchvision: Provides datasets, pre-trained models, and image transformation utilities.
NumPy: Supports numerical operations, indirectly used through PyTorch.
Matplotlib (Optional): For visualizing data and training metrics (e.g., accuracy and loss curves).

IDE
Visual Studio Code: Lightweight, with extensions for Python debugging.
Jupyter Notebook: Interactive environment for testing and visualization.
PyCharm (Optional): Python-focused IDE with advanced debugging tools.

DATABASE
MNIST
The MNIST dataset (Modified National Institute of Standards and Technology) is a popular and widely used dataset in machine learning and deep learning, primarily for training and testing models on handwritten digit classification tasks. The dataset contains 70,000 grayscale images of handwritten digits (0–9). Each image is 28x28 pixels, resulting in a total of 784 features per image (flattened into a 1D vector). The images are centered and size-normalized for consistency. The dataset contains 70,000 grayscale images of handwritten digits (0–9). Each image is 28x28 pixels, resulting in a total of 784 features per image (flattened into a 1D vector). The images are centered and size-normalized for consistency. ach image is labeled with the digit (0–9) it represents.

IX. TEST CASES

System testing is the stage of implementation, which is aimed at ensuring that the system works accurately and efficiently before live operation commences. Testing is the process of executing the program with the intent of finding errors and missing operations and also a complete verification to determine whether the objectives are met and the user requirements are satisfied.

UNIT TESTING
This involves testing individual components or modules of a system in isolation to ensure they work correctly. In your descriptions, some tests could be considered unit tests if they focus on the smallest parts of the algorithms (like testing specific functions or methods).

DEEP LEARNING MODEL ALGORITHM

TEST ID	INPUT	DESCRIPTION	EXPECTED OUTCOMES	PASS/FAIL
1	Architecture validation	Ensure the handwriting net model	Handwriting model has the correct architecture and input/output shapes for each layer.	Pass
2	Training Loop	Verify the training process works expected	Training loss decrease over epochs and validation loss reduction across epochs.	Pass
3	Model saving and loading	Verify the model can be saved and loaded correctly.	Loaded model produces the same accuracy as the saved model.	Pass
4	Data set loading	Check the Number of samples in the training, validation and test sets.	Ensures the MNIST data set is loaded correctly.	Pass
5	Data augmentation	Validate data augmentation transforms.	Images have random notations, translations and normalization applied.	Pass

Table 9. 1:- Deep Learning Model Algorithm FOR CNN ALGORITHM

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429 Volume 6, Issue 4, January 2026 Impact Factor: 7.67

TEST ID	INPUT	DESCRIPTION	EXPECTED OUTCOMES	PASS/FAIL
1	Empty canvas	Handle prediction when no drawing is present	Warning message "input is empty"	Pass
2	Predict digit from canvas	Ensure prediction work for drawn digits.	Predicted digit matches the drawn digit with a confidence score.	Pass
3	Camera initialization	Ensure the camera initialization correctly.	Camera feed is displayed within the applications.	Pass
4	Predict Digit from Camera	Test predictions using live camera input.	Predicted digit matches the digit shown with a confidence score.	Pass
5	Clear Canvas	Ensure the "Clear" button resets the canvas.	Canvas is completely white.	Pass
6	Pen Color Selection	Test the ability to change pen colors.	Lines are drawn in the selected color.	Pass
7	Sound Feedback	Ensure text-to-speech (TTS) feedback works as expected.	The predicted digit is spoken aloud	pass
8	Invalid Model File	Handle errors when the model file is missing or corrupted.	Error message: "Model file not found."	Pass

Table 9. 2:- Unit Testing for CNN Algorithm

X. RESULTS & DISCUSSIONS

In this section, the outcomes of the deep learning based handwritten digit recognition particularly Convolutional Neural Networks (CNNs), have demonstrated exceptional outcomes in handwritten digit recognition. By leveraging the power of deep learning, these models can achieve remarkably high accuracy rates, often exceeding 98%, in classifying handwritten digits from various sources. This success is attributed to the ability of CNNs to automatically extract relevant features from images, enabling them to learn intricate patterns and representations within the data.

DIGIT RECOGNITION RESULTS

Implementing a deep learning model for handwritten digit recognition presented several key considerations. This discussion analyzes the factors that influenced the model's performance, including the impact of the chosen architecture, the effectiveness of data preprocessing, and the sensitivity of the model to hyperparameter values. Here are the key findings:

- **Feature Learning:** CNNs automatically learn hierarchical features from raw pixel data, eliminating the need for manual feature engineering.
- **Representation Learning:** Deep networks can learn complex, high-level representations of the input data, capturing intricate patterns and variations in handwriting styles.
- **Generalization:** Well-trained CNNs can generalize well to unseen data, making them robust to variations in writing styles, sizes, and orientations.

RESULTS: he results showcase the model's ability to generalize well to unseen data, achieving high accuracy and minimal misclassification. Below are the figures and observations.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 4, January 2026
Impact Factor: 7.67

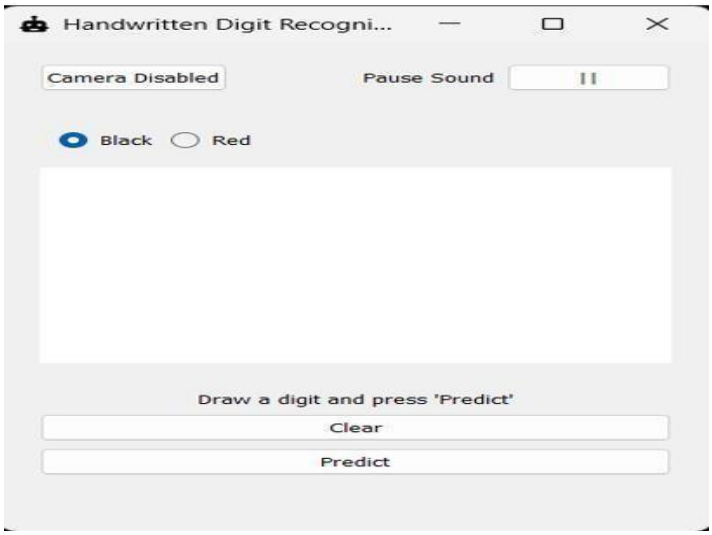

Figure 10.2. 1:- Output for Digital White Board

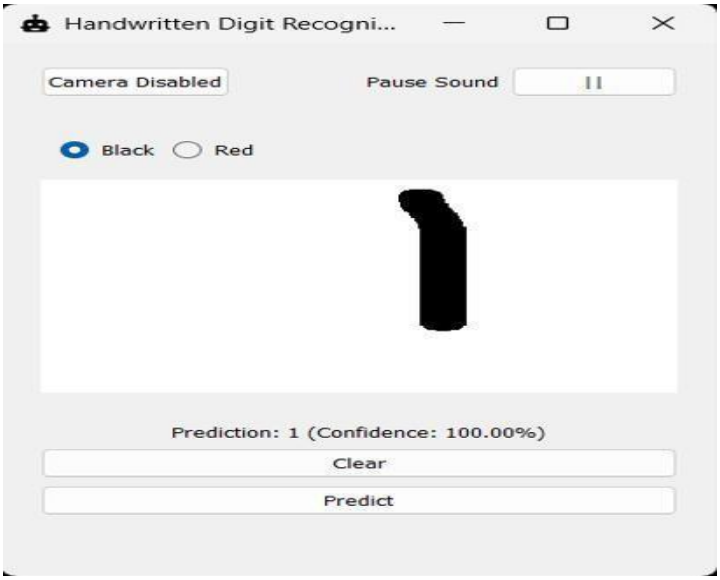

Figure 10.2. 2:- Output for Identified 1 number with black color

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 4, January 2026
Impact Factor: 7.67

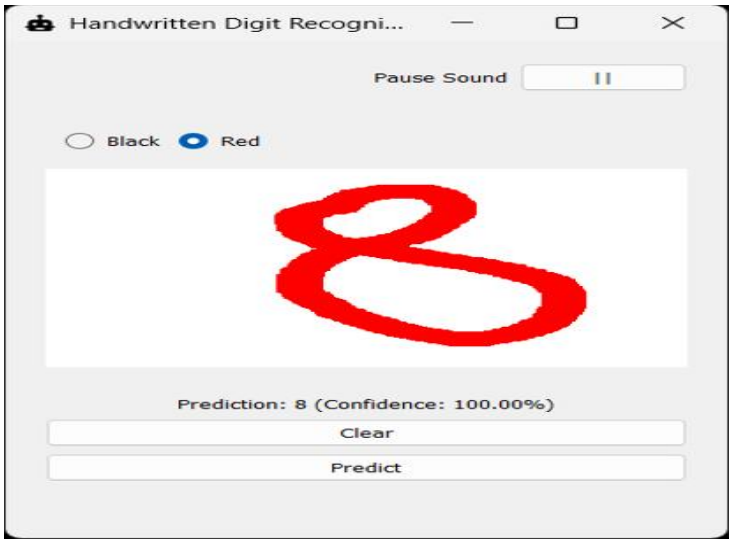

Figure 10.2. 3:- Output for identified 8 Red colour with accuracyC

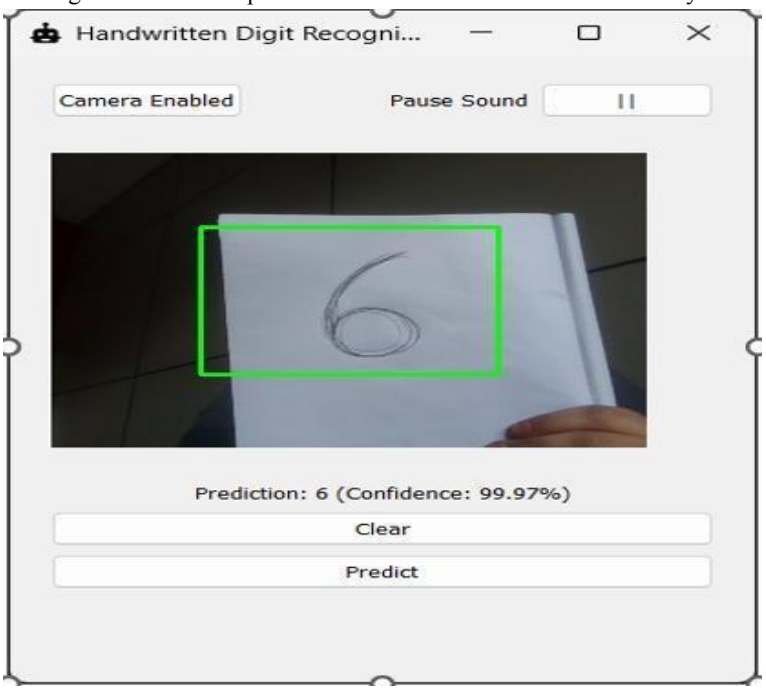

Figure 10.2. 4:- Capturing 6 Digit With Real Time Camera

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 4, January 2026
Impact Factor: 7.67

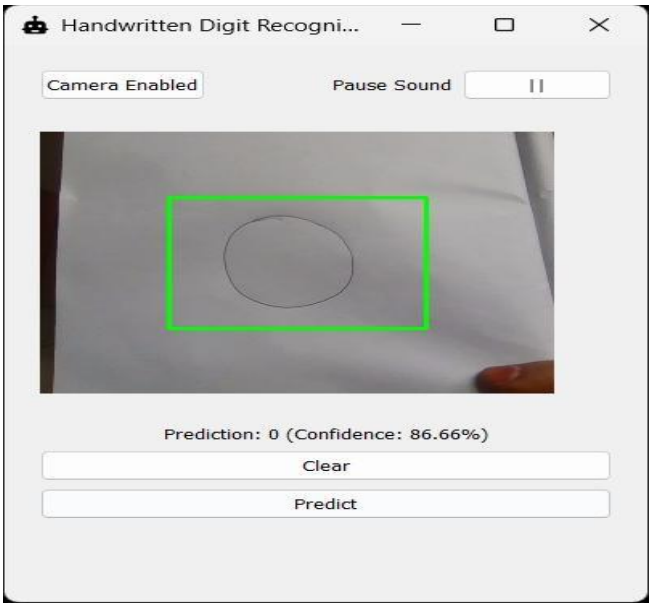

Figure 10.2. 5:- Capturing Digit 0 With Real Time Camera

CONCLUSION

There are problem with machine to recognize the handwritten digit. So, this system will provide a CNN model which can recognize digit and also provide GUI to make user friendly environment, Because of the CNN this system has become more reliable, less complex and easy to understand as well as GUI gives window to user, by which user can draw the digit on it. This project will help to reduce machine efforts to recognize handwritten digits. This project is design to make learning process easy and efficient.

REFERENCES

[1]. Gao, Zan, Yinming Li, and Shaohua Wan. "Exploring deep learning for view-based 3D model retrieval." ACM transactions on multimedia computing, communications, and applications (TOMM) 16.1 (2020): 1-21.

[2]. Gao, Zan, et al. "Adaptive fusion and category-level dictionary learning model for multiview human action recognition." IEEE Internet of Things Journal 6.6 (2019): 9280-9293.

[3]. Herrera-Pereda, Raidel, et al. "A review on digital image processing techniques for in-vivo confocal images of the cornea." Medical Image Analysis 73 (2021): 102188.https://doi.org/10.1016/j.media.2021.102188

[4]. Janeliukstis, Rims, and Xiao Chen. "Review of digital image correlation application to large-scale composite structure testing." Composite Structures 271 (2021): 114143.

[5]. Lowe, Sheila R. "Pattern recognition in handwriting." Advances in Pattern Recognition and Artificial Intelligence. 2022. 77-95. https://doi.org/10.1142/9789811239014_0005.