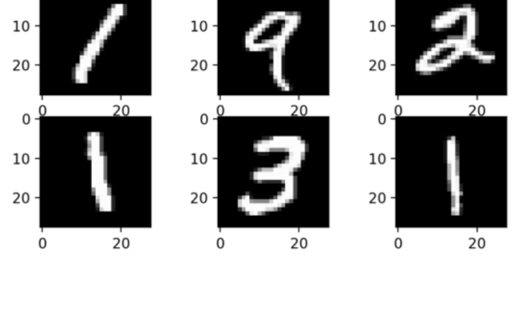
HANDWRITTEN DIGIT RECOGNITION USING NEURAL NETWORK

R S Monish
Computer Science and Engineering
CHENNAI INSTITUTE OF TECHNOLOGY
Chennai ,India
rsmonish.cse2023@citchennai.net

ABSTRACT - Handwritten digit recognition is a fundamental problem in computer vision and pattern recognition, with significant applications in areas such as postal mail sorting, bank check processing, and form data entry automation. Traditional machine learning approaches have been moderately successful, but the advent of neural networks has significantly enhanced performance. This paper presents a neural network-based approach using Convolutional Neural Networks (CNNs) to achieve high accuracy in recognizing handwritten digits. Using the MNIST dataset, which contains 60,000 training and 10,000 testing images of digits from 0 to 9, the proposed system achieves an accuracy of over 99%. The results demonstrate the effectiveness of deep learning models in understanding and recognizing complex patterns, making it suitable for real-world applications.

1. INTRODUCTION

Handwritten digit recognition has been a widely studied problem due to its diverse applications and challenges. Despite its apparent simplicity, variations in writing styles, sizes, and orientations make it a complex task for traditional methods. Early approaches used techniques such as feature extraction combined with machine learning classifiers like k-Nearest Neighbors (k-NN) and Support Vector Machines (SVM). However, these methods struggled with achieving high accuracy due to their reliance on hand-engineered features and inability to generalize well to new data. The introduction of deep learning models, specifically Convolutional Neural Networks (CNNs), has revolutionized this field. CNNs have shown exceptional performance in extracting spatial hierarchies and patterns, making them a preferred choice for image-related tasks. This paper explores the use of CNNs for recognizing handwritten digits and evaluates its effectiveness using the widely-used MNIST dataset.



2. LITERATURE SURVEY

Over the years, various methods have been proposed for handwritten digit recognition. Early approaches relied on traditional machine learning algorithms, such as k-Nearest Neighbors (k-NN), Support Vector Machines (SVM), and Decision Trees, which were effective to a certain extent but required complex feature extraction techniques. These methods often struggled with varying handwriting styles and noise in the data. With the advent of neural networks, researchers began to focus on using multi-layer perceptrons (MLPs) to model non-linear relationships. However, MLPs were limited by their inability to capture spatial information in images. Convolutional Neural Networks (CNNs) addressed this limitation by introducing convolutional layers that automatically learn spatial hierarchies of features. LeNet-5, proposed by Yann LeCun in 1998, was one of the first CNN architectures for digit recognition, setting a benchmark for future research. Subsequent models like AlexNet, VGGNet, and ResNet further improved accuracy and efficiency, making CNNs the state-of-the-art for image classification tasks.

3. SYSTEM DESIGN

The system is designed using a multi-layer Convolutional Neural Network (CNN) architecture. The input to the network is a 28x28 grayscale image from the MNIST dataset.

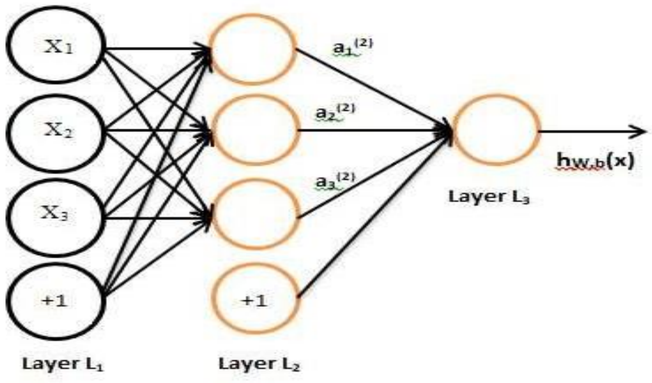
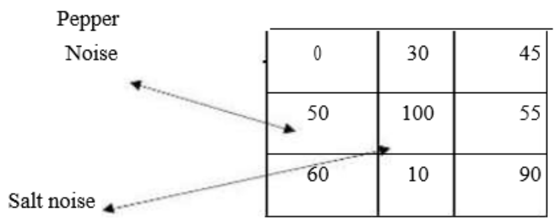


Fig-3.1 Neural Network Model

The architecture consists of multiple convolutional layers followed by ReLU activation functions and pooling layers to reduce dimensionality and retain important features. This is followed by fully connected dense layers that map the extracted features to the output labels (digits 0-9). Dropout layers are used to prevent overfitting, and a softmax layer at the end provides the probability distribution for each class. The model is trained using the cross-entropy loss function and optimized using the Adam optimizer.

4. IMPLEMENTATION

The implementation of the model is done using Python with the TensorFlow/Keras framework. The MNIST dataset, which is readily available in Keras, is split into training and testing sets. The CNN is built using sequential layers, starting with convolutional layers that extract features from the images.



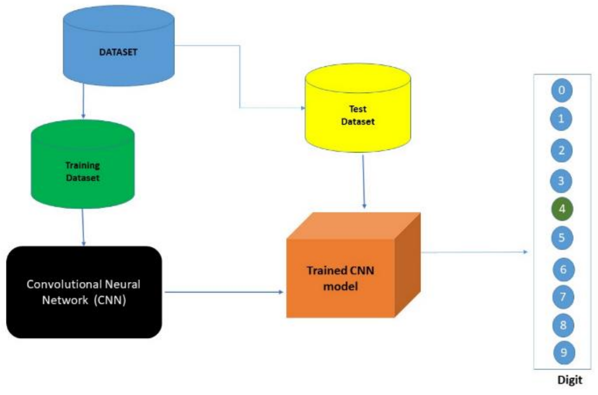
0	30	45
50	100	55
60	10	90

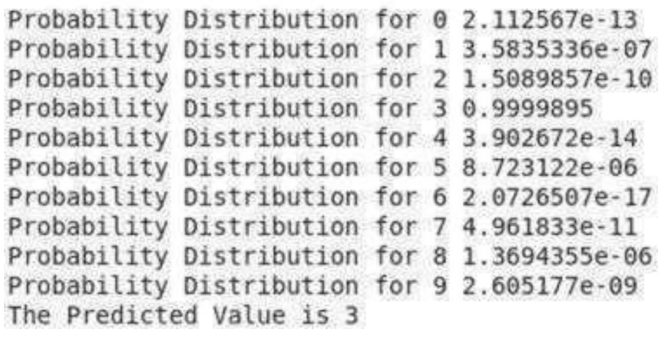
Table 4.2 Type of Noises

These layers are followed by pooling layers to reduce the feature map size, and dropout layers are added to prevent overfitting. After feature extraction, fully connected layers are used for classification.

The training process is carried out for 20 epochs with a batch size of 128. Data augmentation techniques, such as random rotations and shifts, are applied to increase the robustness of the model. After training, the model is evaluated using the test set, and the results are analysed.

Image, you provided, which appears to be a flowchart illustrating the process of training a Convolutional Neural Network (CNN) for digit recognition. It shows the data flow from a dataset to training and testing datasets, ultimately leading to a trained CNN model that predicts digits from 0 to 9.





```
Probability Distribution for 0 2.112567e-13
Probability Distribution for 1 3.5835336e-07
Probability Distribution for 2 1.5089857e-10
Probability Distribution for 3 0.9999895
Probability Distribution for 4 3.902672e-14
Probability Distribution for 5 8.723122e-06
Probability Distribution for 6 2.0726507e-17
Probability Distribution for 7 4.961833e-11
Probability Distribution for 8 1.3694355e-06
Probability Distribution for 9 2.605177e-09
The Predicted Value is 3
```

5. FLOWCHART

A flowchart is a type of diagram that represents a workflow or process.

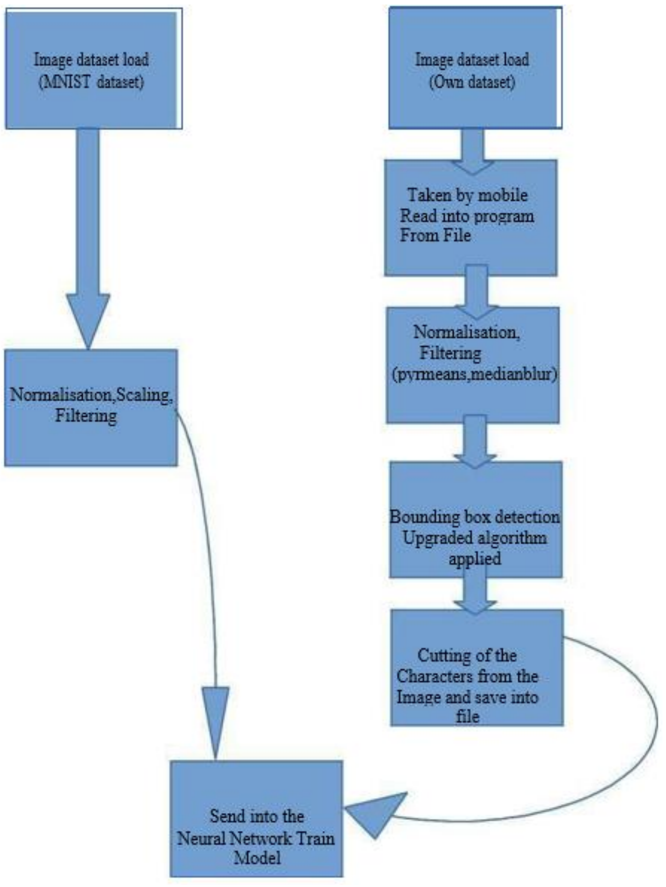


Image dataset load (MNIST dataset)
Image dataset load (Own dataset)
Taken by mobile Read into program From File
Normalisation, Filtering (pyrmeans,medianblur)
Normalisation,Scaling, Filtering
Bounding box detection Upgraded algorithm applied
Cutting of the Characters from the Image and save into file
Send into the Neural Network Train Model

Fig 5.1 Implementation flowchart

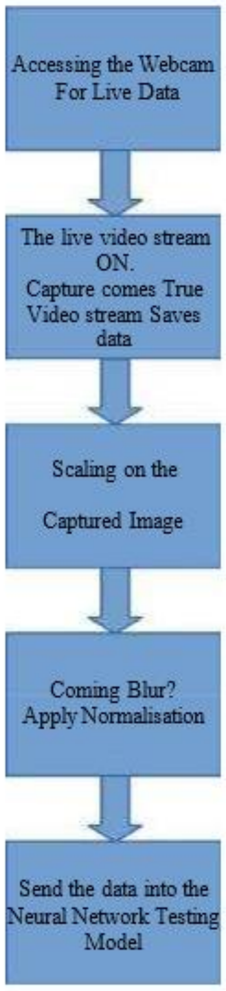

Accessing the Webcam For Live Data
The live video stream ON. Capture comes True Video stream Saves data
Scaling on the Captured Image
Coming Blur? Apply Normalisation
Send the data into the Neural Network Testing Model

Fig 5.2 Image Processing for Testing data

This Flowchart figure 5.2 shows the Image Processing part for the testing data. Testing dataincludes the part of the Accessing the webcam and start the video stream on which an imagedataset will be tested

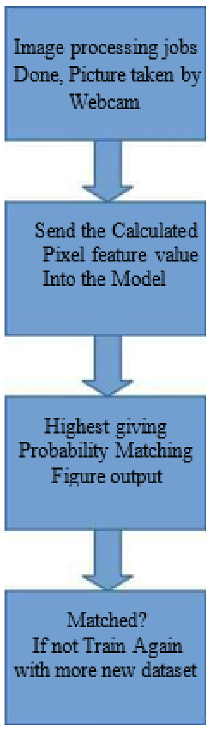


Image processing jobs Done, Picture taken by Webcam
Send the Calculated Pixel feature value Into the Model
Highest giving Probability Matching Figure output
Matched? If not Train Again with more new dataset

Fig 5.5 Neural Network Model for Testing Data

Neural Network Model gets the prediction from the testing image. Thanks to the Image processing part for the testing image that actually plots the data in the image into the pixels calculated the probabilistic values by sending it into the neural network. If matched then it is a successful one if not then it is an error. May be training will be done with it later. The figure 5.5 depicts the flow of the processes.

6. CHALLENGES FACED

During the implementation, several challenges were encountered, including overfitting due to the relatively small size of the MNIST dataset and high variance in writing styles. Overfitting was addressed using dropout and data augmentation techniques. Another challenge was the selection of optimal hyperparameters, such as learning rate, batch size, and number of layers. Finding the right balance between model complexity and training time was crucial. Additionally, training deep networks can be computationally expensive, and managing resource limitations required the use of GPU acceleration.

- **Handwriting Variability**: Diverse styles and intra-class variations.
- **Noise & Quality**: Background noise and inconsistent contrast.
- **Size & Orientation**: Scale and rotation differences.
- **Segmentation Issues**: Connected digits and irregular padding.
- **Class Ambiguity**: Similar-looking digits (e.g., '4' vs. '9').
- **Generalization**: Limited diversity for real-world adaptation.
- **Feature Extraction**: High dimensionality and selecting relevant features.

The Challenges facein Handwritten Handwritten Digit Recognition
1
2
3
Varability in in handwritine styles
Noise and distotion in dignit shapes
Ocoss occlusation of poor lighting cualtys
Noise and dit poorring condiions
5
4
Oclustions of similol dingrss
Ocintnrration of pmor digits
Misintsrprration of similol dig65

7. TESTING

The model was tested using the MNIST test dataset, which contains 10,000 images.

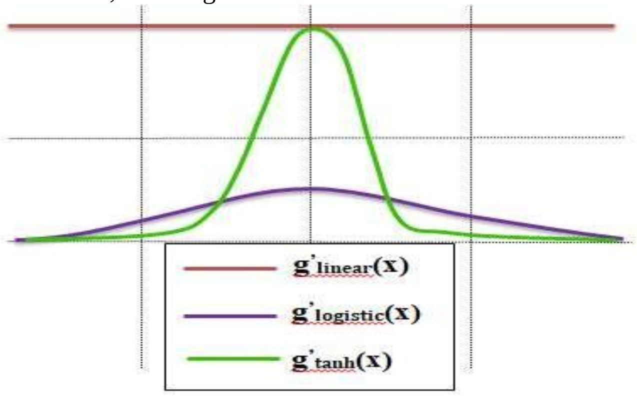


Fig – 4.2 Types of activation functions over graph

accuracy, precision, recall, and F1-score. Confusion matrices
insights into which digits were most frequently confused. The performance of the CNN model was compared against traditional machine learning models, demonstrating a significant improvement in accuracy and robustness.

Testing checks if actual results align with expected outcomes and ensures software is defect-free. It involves evaluating components to identify errors and gaps in requirements.

1) *Unit Testing:*

It focuses on individual software components, typically performed by programmers. Successful unit tests indicate that all modules are functioning correctly, allowing further development.

2) *Functional Testing:*

It verifies that the software meets its functional requirements through Black Box Testing. It includes checking inputs, workflows, and recognition accuracy, while also calculating errors.

3) *Integration Testing:*

It assesses the interactions between combined software modules and their components. It ensures that all parts work together seamlessly and identifies any issues in the integrated system.

4) *System Testing:*

It evaluates the complete system, including software and tools, against all requirements. It is tested across various platforms, ensuring compliance with specified functionalities.

5) *Acceptance Testing:*

It determines if the system meets business needs and is ready for delivery. Stakeholder feedback is used to assess the project's acceptance by delivery partners.

8. RESULTS

The CNN model achieved an accuracy of 99.2% on the MNIST test set, outperforming traditional approaches.

Total Tested Samples	Contour Removal Algorithm	Updated Contour Removal Bounding Box	Accuracy
134	56	119	88.8
Noise Cacellation			
120	110	116	96.6

The model showed strong generalization and was able to recognize digits accurately, even with variations in style and
noise. Misclassification primarily occurred in digits with very similar shapes (e.g., 3 and 8).

Fig-7.1 Detection of the image

Compared to classical methods like k-NN and SVM, which typically achieve accuracies around 96-97%, the proposed CNN model demonstrated superior performance.

9. DISCUSSION

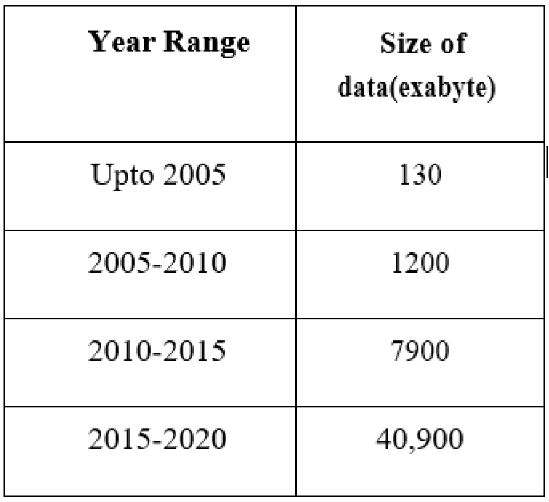
The results demonstrate that Convolutional Neural Networks (CNNs) are highly effective for handwritten digit recognition, owing to their ability to hierarchically learn features through convolutional layers. This capability allows CNNs to extract low-level patterns, such as edges and corners, and build higher-level abstractions like loops and curves, which are essential for capturing complex spatial relationships in the digits. In contrast, traditional machine learning models often require manual feature extraction and struggle to achieve similar performance.

Despite this success, the current model's accuracy could be further enhanced through the use of deeper architectures or ensemble methods that combine multiple models to reduce error rates. A notable limitation, however, is the reliance on the relatively simple MNIST dataset, which lacks real-world variations such as different backgrounds, lighting conditions, and more diverse handwriting styles. This simplicity means that while the model performs exceptionally well on MNIST, its ability to generalize to more complex scenarios may be limited.

To address this, future work should focus on extending the approach to more challenging datasets like EMNIST, which includes a broader set of handwritten characters, or exploring handwritten text recognition tasks. Such extensions would improve the model's robustness and practical applicability.

10. CONCLUSION

This is a project on OCR (Optical Character Recognition). This project is non-fundable project and designed with full of interest which also includes some outer concept on statistical modeling and optimizer technique.



Year Range	Size of data(exabyte)
Upto 2005	130
2005-2010	1200
2010-2015	7900
2015-2020	40,900

Fig 9.1 Range of data

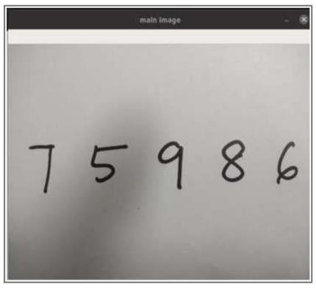


Fig -4 Standard input

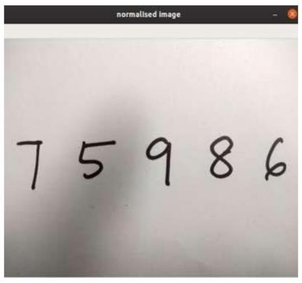
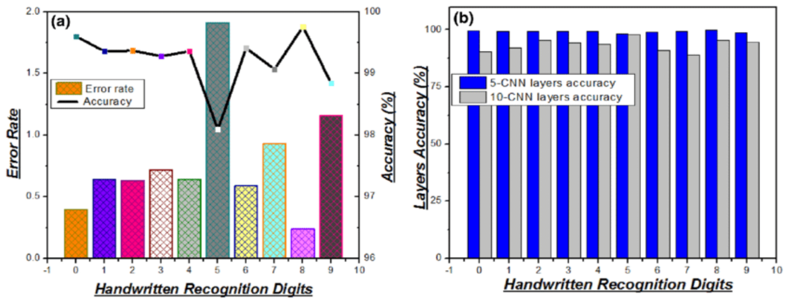


Fig -5 Handwritten Input for Training after minmax normalisation

This paper presented a neural network-based approach for handwritten digit recognition using Convolutional Neural Networks.

Fig-9.1 Test dataset result after 5th time



The proposed model demonstrated high accuracy on the MNIST dataset, making it a viable solution for practical applications. Despite the challenges faced during implementation, CNNs have proven to be a robust tool for image classification tasks. Future work could explore optimizing the model for real-world scenarios, such as recognizing handwritten text in natural images. This study reinforces the potential of deep learning models to achieve state-of-the-art performance in various image recognition tasks.

11. REFERENCES

1. Non-recursive Thinning Algorithms using Chain Codes Paul C K Miwok Department of Computer Science The University of Calgary Calgary, Canada T2N 1N4
2. A dynamic shape preserving thinning algorithm Louisa Lam and Ching Y. Suen Centre for Pattern Recognition and Machine Intelligence and Department of Computer Science, Concordia University, 1455 de Maisonneuve Blvd. W., Montrdal, Qudbec H3G 1MS, Canada
3. Object Contour Detection with a Fully Convolutional Encoder-Decoder Network Jimei Yang Adobe Research jimyang@adobe.com Brian Price Adobe Research bprice@adobe.com Scott Cohen Adobe Research scohen@adobe.com Honglak Lee University of Michigan, Ann Arbor honglak@umich.edu Ming-Hsuan Yang UC Merced mhyang@u
4. Contour Detection and Image Segmentation by Michael Randolph Maire B.S. (California Institute of Technology) 2003
5. Three-Dimensional Nonlinear invisible Boundary detection ,IEEE Transaction on Image Processing VassiliKovalev,J,Chen
6. Unconstrained OCR for Urdu using Deep CNN – RNN Hybrid Networks; Mohit Jain, Minesh Mathew et al.
7. Neural Network and Deep Learning by Michael Nielsen.
8. How to implement a Neural Network intermezzo 2, Peter Roelant’s(2016)