Handwritten Digit Recognition using CNN and Deep Learning

Neetu[1], Dr. Amandeep[2], Ankita Sharma[3]

M.Sc. Computer Science[1,3], Artificial Intelligence and Data Science, GJUS&T Hisar,

Assistant Professor[2], Artificial Intelligence and Data Science, GJUS&T Hisar, Email- jangraneetu79@gmail.com

Abstract— The use of intelligent systems has become much more common in recent years. Among these, handwritten digit recognition has become a key problem in the field of artificial intelligence and pattern recognition. This study examines the creation and assessment of models for handwritten numerical digit recognition using the MNIST dataset. SVM, MLP, and CNN are the three techniques. We also examine the effects of network width (number of filters per layer), depth (number of convolutional layers), and activation functions (such as ReLU, Tanh, and Sigmoid) on the learning capacity and generalisation ability of the CNN models. CNNs are useful for large-scale handwritten digit classification jobs because, according to experimental data, they provide recognition accuracy and faster processing times than typical machine learning models. In this project their is the use of CNNs and deep learning for handwritten digit recognition. We examine the fundamental architecture of CNNs, which consists of fully connected layers for final classification and decision-making, pooling layers to reduce the dimensions and translational invariance and convolutional layer for feature extraction using filters.

Keywords

Handwritten Digit Recognition, MNIST Dataset, Convolutional Neural Network (CNN), Support Vector Machine (SVM), Multi-Layer Perceptron (MLP), Deep Learning, Machine Learning.

I. INTRODUCTION

The recognition of handwritten digits is a traditional studied field in ML and DL. It refers to a system's ability to correctly identify and classify handwritten numeric digits (0–9) from various sources such as scanned documents, digital images, or touch enabled devices. These systems have immense practical value and are used in real-world applications like automated bank cheque processing, traffic number plate analysis, postal mail sorting, and digital note transcription.

Despite the progress in pattern recognition, recognizing handwritten digits is a challenging task due to the difference in human handwriting[2]. Factors such as different writing styles, angles, spacing, pressure, and noise in scanned images add complexity to this classification problem. Unlike traditional Optical Character Recognition (OCR) that focuses on typed or printed text, handwritten digit recognition must result for high variability and irregularities in input data[1]. This research explores and compares three major algorithms— SVMs, MLPs, and CNNs to understand their advantage in handwritten digit classification. The comparison focuses on different type of parameters like accuracy, training and testing time, error rates and computational complexity.

The goal is to identify which technique gives the best trade-off between precision and performance for deployment in real time applications. The MNIST dataset for handwritten digit recognition is used for training and testing of the model[3]. The classification task is thus framed as a 10 class problem (for digits 0–9), and performance is visualized using graphs generated by using Matplotlib. In real world use cases model accuracy plays a important role.

For example an error in recognizing the numerical amount on a cheque or an identification number on a form can lead to costly consequences. Hence, the importance of selecting the most accurate and efficient model cannot be enhanced.

In this paper a comprehensive understanding of the underlying techniques behind SVM, MLP, and CNN models are explained. It not only discusses their theoretical foundations but also presents the experimental implementation and evaluation.

Earlier methods mostly used manually designed rules and features, which didn't work well when they faced new or unfamiliar handwriting styles. With the rise of deep learning, especially Convolutional Neural Networks (CNNs), there has been a big improvement in how accurately these systems work. This study also examines a number of optimisation and architectural assessment methods that are crucial for creating high-performance CNN-based HDR systems.

In this an overview of three algorithms and the background of HDR are explained : -

A. Convolutional Neural Network (CNN):

Figure 1 shows a basic CNN model. CNNs are the most efficient design in image recognition due to their capacity to learn many hierarchies. Convolutional layers that use filters to extract features make up a conventional CNN. Pooling layers that preserve important information while reducing dimensionality. To avoid overfitting, dropout layers are used. The output layer is fully connected, and the classifier's final result is the maximum value of the output neurones [5].

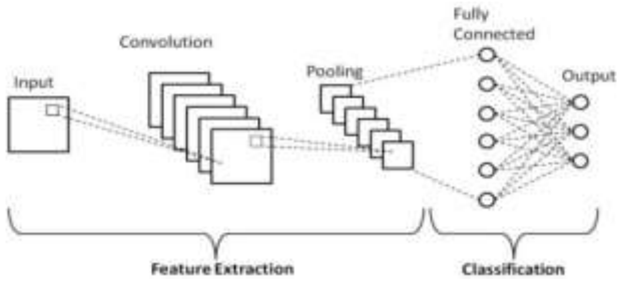


Fig. 1. Structure of CNN

B. Deep Belief Network (DBN):

A type of unsupervised learning algorithm is the Deep Belief Network, which generates probabilities [5]. There are several Restricted Boltzmann Machines (RBM) in it. By stacking several

RBM, DBN may extract more abstract features thanks to RBM's efficacious feature extraction technique [6]. Fig. 2 depicts a typical DBN structure.

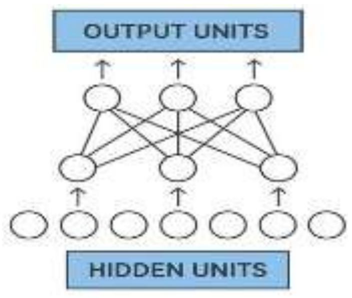


Fig. 2. Input and output units in DBN

C. Deep Neural Network (DNN):

A second set of test images is used to evaluate generalisation performance after the originally random weights of DNN are trained again and again to minimise the classification error on a set of labelled images for training [4]. The winner-take-all neurones in DNN's two-dimensional layers have overlapping receptive fields and shared weights.

A straightforward max pooling method divides layers into squared zones of and chooses a neuron which is active among all others in each sector to identify winning neurons given an input pattern [7,8].

Some layer winners feed the next layer in the hierarchy by representing a smaller, lower-resolution, down-sampled layer [8]. The method is based on the groundbreaking research of Hubel and Wiesel on the primary visual cortex of cats, which revealed that there are complex cells that execute down-sampling-like functions .(Fig. 3).

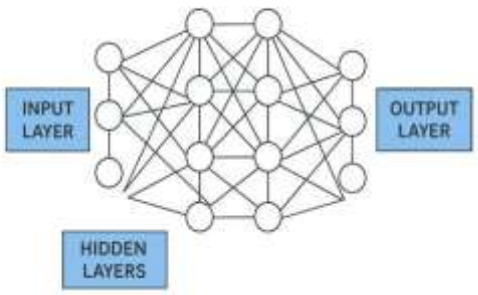


Fig. 3. The structure of DNN

II. LITERATURE REVIEW

The first approach to HDR were mostly the rule based technique and template matching methods.

These early systems based on hand coded rules to detect specific shapes of digits. These methods needed a good understanding of image structures to define the right features for each digit.

1. LeCun et al. (1998) – Gradient-Based Learning Applied to Document Recognition [9]

One of the first and most significant papers in handwritten digit recognition is this one. It introduced the LeNet-5 model, which is a type of Convolutional Neural Network (CNN). The MNIST dataset was used to train and test their model.

2. Cireşan et al. (2010) – Deep, Big, Simple Neural Nets Achieve State-of-the-Art Performance [10]

This paper focused on training large CNNs for digit recognition. Many layers was used by the authors trained deep networks. They didn't use GPUs, which was uncommon at the time. Despite that, they got high accuracy using smart tricks. Their models were simple but deep, meaning many layers.

3. Wan et al. (2013) – Regularization of Neural Networks using DropConnect [11]

This research introduced a new regularization technique

called DropConnect. It's like Dropout but instead of dropping activations, it drops weights. The authors used this method on CNNs for digit recognition. They showed that DropConnect improved generalization. Their model performed well on MNIST and CIFAR-10.

5. Hinton et al. (2006) – Reducing the Dimensionality of Data with Neural Networks [12]

This paper introduced deep belief networks (DBNs), which were early versions of deep learning models. It explained how neural networks can learn data representations without supervision. The authors used a layer-wise training method, where each layer is trained one at a time.

6. Wan et al. (2013) – Regularization of Neural Networks using DropConnect [13]

In this research, the authors introduced a regularization technique called DropConnect, which is a variation of the popular Dropout method. In Dropout, random neurons are turned off during training to prevent overfitting. But in DropConnect, random weights are set to zero instead.

Applications like automated cheque clearing, postal mail sorting, form scanning, and handwriting input on mobile devices all rely on the system's ability to understand handwriting from different people. Interestingly, some modern HDR models are now reaching or even surpassing human-level accuracy in specific cases.

II.
III.

III. METHODOLOGY

A. Preprocessing

There is currently no need for a noise reduction strategy because every image in the database is clear and noise-free. However, we must eliminate noise from the photos in a real system. There is a possibility that any paper will have optical noise. The shapes of the characters may not always be distinct, particularly in handwritten writings. Preprocessing is therefore required.

In order to remove one-bit mistakes and provide a clean edge, we will first apply an erosion using three-by-three structural components. Next, 2 X 2 elements are used to dilate the characters For ease of computation, we have transformed our image from RGB to greyscale in Figure 2 by dividing the 255 pixel value to obtain a number between 0 and 1[14]. Additionally, we are transforming the data from category values into binary or vector forms during this procedure. One-hot encoding: A categorical label is converted into a binary vector using one-hot encoding, in which all of the indexes are "0" and only the label's index is "1." The number "3" becomes [0, 0, 0, 1, 0, 0, 0, 0, 0, 0], for instance.

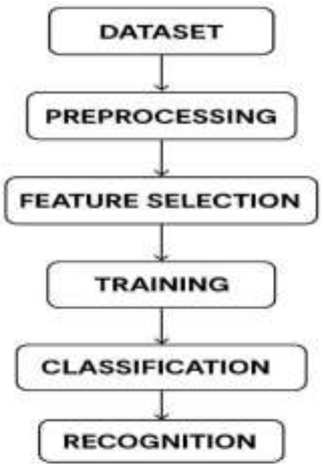


Fig. 9. A block diagram of model.

B. Convolutional neural network

The CNN model comes after the preprocessing is finished. Four hidden layers make up CNN, which aids in the extraction of features from photos and makes predictions [15] .
Images and other data having a grid-like layout can be processed by CNNs, a specific type of neural network. It has the essential advantage of being able to identify important highlights without the need for human management.

1. Convolutional Layer
A convolution layer in transforms the input data using a convolution operation. The convolution process creates a feature map by summing the results of element wise multiplications carried out by a filter (or kernel) that moves over the input data. The network can identify patterns in the input images, including edges, textures, and shapes.

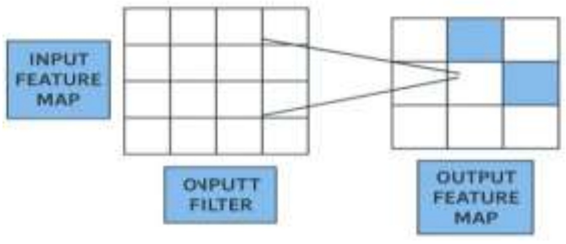


Fig. 5 Convolutional Layer

1. Relu Layer
The activation function receives the feature maps, much like it would in a typical artificial neural network. They are fed into a rectifier function, which, in the event that the input value is less than zero, returns zero; if not, it returns the input value.

$$R(z) = \begin{cases} z & > 0 \\ 0 & \leq 0 \end{cases}$$

2. Pooling Layer
Using this layer the most significant features from a region are chosen like when choosing the average or strongest signal from a tiny data block. The two most popular forms of pooling are average pooling, which takes the average value, and max pooling, which takes the highest value.

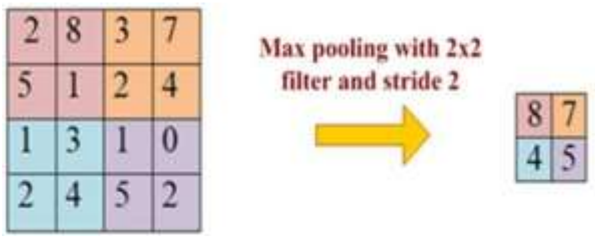


Fig. 6 Pooling Layer

3. Fully connected Layer
In this layer every neuron in one layer is connected to every neuron in the next layer. It is present at the end of the network and is mainly used for final decision making or classification. In this layer, all the extracted features from previous layers are combined and processed to give the final output.
It does the implementation of CNN model in which all the layers which are required are present.

C. Training of Model
After building the model we train our raw data onto it so that we can evaluate it. In training we check that the model is working on our data or not. First we train the dataset on training dataset and than test it on testing dataset. The number of epochs means the number of iterations on the dataset on training and by these epochs we check the accuracy of our model on given dataset.

Training data shape: (60000, 28, 28)
Test data shape: (10000, 28, 28)

We'll convert them into one-hot vectors like this:

5 → [0 0 0 0 0 1 0 0 0 0]

D. Classification & Recognition
A recognition system's decision-making phase makes use of the features that were extracted in the earlier phase. In classification, a feed forward neural network is used. It has backpropagation also. In output layer a digits is detected from the given raw handwritten image and give as output , while the hidden layers employ the log sigmoid activation function. After training, an expert compares the Digits to determine how accurate the tip .

IV. RESULTS

After iterations of the number of epochs the model is trained and give the score of accuracy and loss accuracy means the percentage of the model to understand the patterns of the data and loss means the percentage of the missing patterns of features from the data .
Our model get 99 percent accuracy on the training dataset.

Accuracy - how may data points are correctly predicted/ total data points correct predictions/total no of predictions

Recall - out of actual 1's how many are correctly predicted TP/TP+FN actual positive class

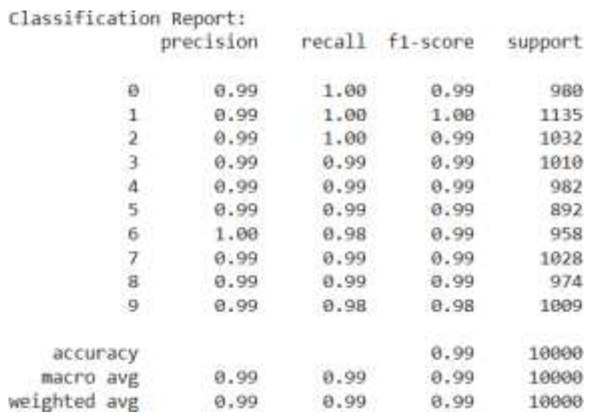
Precision - out of predicted 1's how many are correctly predicted TP/TP+FP predicted positive class

F1-score is the harmonic mean of precision and recall, provides a balance between the two.

False Positive Rate(FPR):
[0.00110865 0.00067682 0.00122658 0.00166852 0.000998 0.00142732
0.00022119 0.00089166 0.00155107 0.00144589]
False Match Rate (FMR): 0.0101

Test accuracy: 0.9899
Test loss: 0.0279

.



```
Classification Report:
              precision    recall  f1-score   support

           0       0.99      1.00      0.99       980
           1       0.99      1.00      1.00      1135
           2       0.99      1.00      0.99      1032
           3       0.99      0.99      0.99      1010
           4       0.99      0.99      0.99       982
           5       0.99      0.99      0.99       892
           6       1.00      0.98      0.99       958
           7       0.99      0.99      0.99      1028
           8       0.99      0.99      0.99       974
           9       0.99      0.98      0.98      1009

    accuracy                           0.99     10000
   macro avg       0.99      0.99      0.99     10000
weighted avg       0.99      0.99      0.99     10000
```

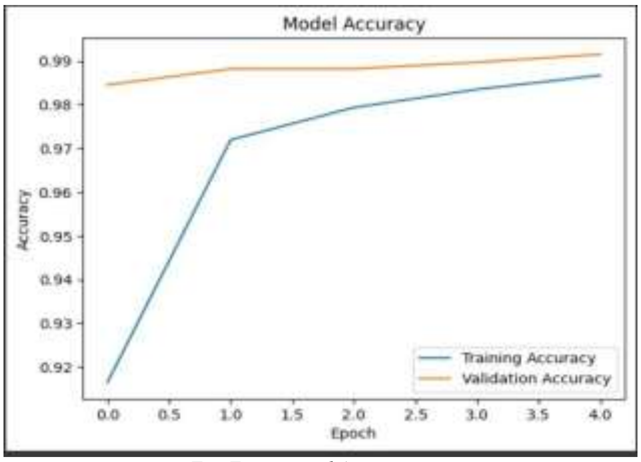
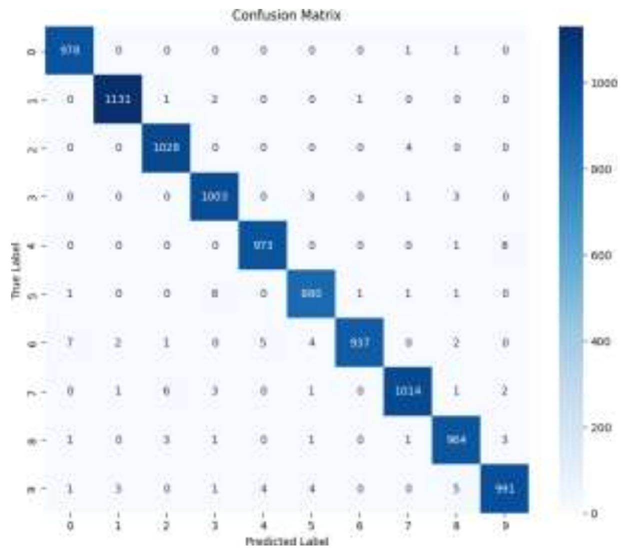


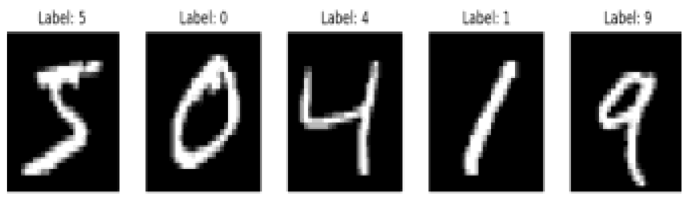
Fig: Training and Accuracy

Confusion matrix - is a 2*2 matrix that gives count of TP,TN,FP,FN . Confusion matrix visually demonstrates where the model confuses certain digits with others.

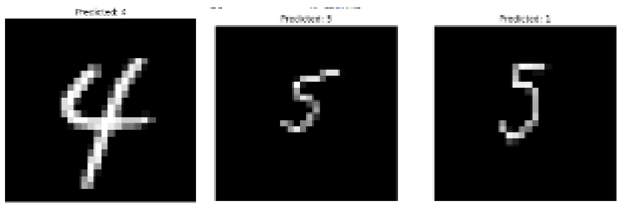


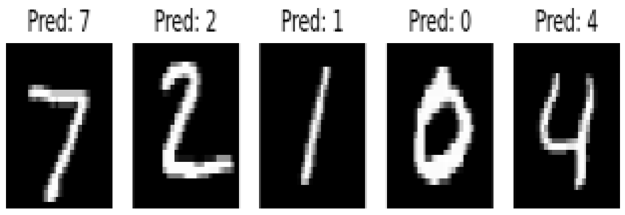
V. PREDICTION

This is the final part of the project where we will check the prediction of our model is right or not. And also check the percentage at which it correctly predicting the give raw handwritten image .
The image is passed through the CNN. Each layer of the CNN extracts complex features. The final layer of the model is typically a dense layer with softmax activation, which outputs a probability distribution over all possible digit classes (0 through 9). The result with the highest probability is selected as the predicted digit.



These are the predicted images from the images from the test dataset I have used in the model.





These are the images predicted images by the model from the testing dataset and the image from user side. The accuracy of the model is checked on the images of testing dataset and also on my own custom images handwritten digits.

VI. CONCLUSION

The first major step in this project involved preparing the dataset for training. By using preprocessing techniques like normalisation, reshaping, and one-hot encoding, the data was processed by CNN model.
The first important step in this project is preparing the dataset for training of the model. By using preprocessing techniques like normalisation, reshaping, and one-hot encoding, the data was processed by CNN model.
The model followed a complete end-to-end learning pipeline—from raw pixel input to final digit classification—which highlighted the advantages of deep learning in reducing human effort while increasing reliability and accuracy. The confusion matrix in particular helped to identify digit pairs that were misclassified, giving insights into areas where further improvement was needed.
Beyond the technical aspects this project was also a valuable learning experience. It helped build a deeper understanding of

deep learning workflows and how theoretical concepts from textbooks are applied in actual implementations.

The model achieved over 98% accuracy, which is in line with results seen in high-quality academic studies. The entire model from data collection using the MNIST dataset to preprocessing, model building, training, evaluation, and finally deployment was executed in a structured manner using platforms like Google Colab. The training process was optimised using GPU support.

VII. REFERENCES

[1] X. X. Niu and C. Y. Suen, "A novel hybrid CNN–SVM classifier for recognizing handwritten digits," *Pattern Recognition*, vol. 45, pp. 1318–1325, 2012.

[2] Z. Li *et al.*, "Handwritten digit recognition via active belief decision trees," in *Proc. 35th Chinese Control Conf. (CCC)*, 2016, IEEE.

[3] Y. LeCun, C. Cortes, and C. J. Burges, *The MNIST database of handwritten digits*, 1998.

[4] "Investigating Resource Allocation Techniques and Key Performance Indicators (KPIs) for 5G New Radio Networks: A Review," *Int. J. Comput. Netw. Appl. (IJCNA)*, 2023. (Scopus CiteScore: 1.3)

[5] "Secure and Compatible Integration of Cloud-Based ERP Solution: A Review," *Int. J. Intell. Syst. Appl. Eng. (IJISAE)*, vol. 11, no. 9s, pp. 695–707, 2023. (Scopus CiteScore: 1.46)

[6] "Ensemble Learning based malicious node detection in SDN based VANETs," *J. Inf. Syst. Eng. Bus. Intell.*, vol. 9, no. 2, Oct. 2023. (Scopus)

[7] "Security in Enterprise Resource Planning Solution," *Int. J. Intell. Syst. Appl. Eng. (IJISAE)*, vol. 12, no. 4s, pp. 702–709, 2024. (Scopus CiteScore: 1.46)

[8] "Secure and Compatible Integration of Cloud-Based ERP Solution," *J. Army Eng. Univ. PLA*, vol. 23, no. 1, pp. 183–189, 2023. (Scopus)

[9] "Secure and Compatible Integration of Cloud-Based ERP Solution," *J. Army Eng. Univ. PLA*, vol. 23, no. 1, pp. 183–189, 2023. (Scopus)

[10] Z. Zhao and L. Chen, "Conversational AI with personalized speech," in *Proc. AAAI Conf. Artif. Intell.*, vol. 36, no. 10, pp. 11154–11162, 2022.

[11] M. Jain and A. Tiwari, "Chatbots with cloned voices for customer engagement," *J. AI Appl.*, vol. 5, no. 2, pp. 45–52, 2023.

[12] T. George, M. Raman, and K. Das, "AI narrators in journalism and publishing," *MediaTech J.*, vol. 7, no. 4, pp. 14–22, 2022.

[13] J. Tan, Z. He, X. Wang, and J. Yamagishi, "A survey on neural voice cloning," *arXiv preprint*, arXiv:2109.00252, 2021.

[14] S. España-Boquera, M. J. Castro-Bleda, J. Gorbe-Moya, and F. Zamora-Martínez, "Improving offline handwritten text recognition with hybrid HMM/ANN models," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 4, pp. 767–779, Apr. 2014.

[15] M. Ahmed, A. G. Rasool, H. Afzal, and I. Siddiqi, "Improving handwriting-based gender classification using ensemble classifiers," *Expert Syst. Appl.*, vol. 85, pp. 158–168, 2017.