Handwritten Digit Recognition Using CNN

Nithyapriya S[1], Shanthini C[2], Suvalaxmi S[3], Shalini P[4]

[1]Department of Artificial Intelligence and Data Science, Bannari Amman Institute of Technology
[2]Department of Artificial Intelligence and Data Science, Bannari Amman Institute of Technology
[3]Department of Computer Science and Engineering, Bannari Amman Institute of Technology
[4]Department of Electronics and Communication Engineering, Bannari Amman Institute of Technology

---***---

Abstract - This project explores the application of Convolutional Neural Networks in Handwritten digit recognition. Exploiting the widely used MNIST dataset, we proposed a deep learning model that encompasses the Ensemble model that is capable of predicting and recognizing the handwritten digits and achieves remarkable precision. The dataset contains 70000 images in total, where 55000 images are for training, 5000 images for validating, and 10000 images for testing. By using CNN, we surpassed the traditional way of training the algorithm with machine learning. By incorporating robust pre-processing techniques and innovative training ideas and strategies, our model showcases resilience to real-world problems and challenges which includes reading the postal codes in the written mail, reading the digits in the handwritten checks, digits drawn on the mobile touch panel, etc. To better extract the features of the complex handwritten digits, bagging is used from the Ensemble techniques. The objective of this project is to find a better optimizer to work with ensemble models, which encompasses the merging of optimizers with the deep learning models.

Key Words: Handwritten digits, CNN, Ensemble Model, Optimizers, MNIST.

1.INTRODUCTION

Handwritten Digit Recognition is 'Hello World' of object detection, object recognition or pattern recognition. The MNIST dataset is used where the dataset is constructed from a number of scanned documents which were available in the NIST (National Institute of Standards and Technology). This is the small history of MNIST name formation or Modified NIST. Handwritten digits, in particular, pose a unique challenge. They exhibit vast diversity in style, size, structure, making the task of recognition an intricate puzzle. A variety of documents have been used to take the images for the dataset, then they were normalized and centralized in their sizes. This makes it easy for the users of MNIST dataset to directly work with the model building rather than data cleaning or requirement satisfaction. Each image in the dataset is 28x28 pixels with 784 pixels in total. The dataset itself has a standard split of 60000 images for training and then 10000 images for testing which in total consists of 70000 handwritten digit images. The digits are numbered from 0 to 9, sum of 10 classes. With this we can develop a product which could automatically recognize the Arabic numerals in different scenarios with the help of computers. For training the computers, Convolutional Neural Networks have revolutionized the way of understanding the images. With the ability to learn complex patterns and structures, CNN's can interpret the strokes and curves of the Arabic numerals. In addition to CNN, this paper introduces a concept of Ensemble learning, a strategy which encompasses the collection of multiple CNN models. Instead of a single model result, it employs an ensemble of CNNs, which are trained independently and to work collectively. This approach will add a layer of resilience, accuracy and adaptability to the recognition process. The project also illustrates the entire journey of data to intelligence. By beginning with the vast data of handwritten digits, each digit has a unique expression of creativity by humans. The data have been collected from the MNIST dataset. For the preprocessing process, we should clean and preprocess the data to make it suitable for the deep learning model. The common step for the preprocessing includes resizing of images, normalization of pixel values and splitting the dataset into training and the test set data, rotation of the images, shifting of images in horizontal and vertical manner randomly, zoom of digits randomly, encoding of digits. The process of feature extraction is taken care of by the Convolutional Neural Network model itself, where it helps to identify the relevant patterns and the characteristics in the images. The next step is to select the model, where model selection is an iterative process. It involves experimenting with multiple model architectures, hyperparameters, regularization techniques etc., . In this project, we started with CNN models like VGG16, ResNet, LeNet - 5 models. And then based on the computation time and the efficiency, LeNet – 5 is selected. With LeNet-5, the optimizers were included. The Adam and the Stochastic Gradient Descent optimizers are used independently. Then they were collaborated in separate phases like the first part of training with Adam and the second part of training is with Stochastic Gradient Descent. And then the ensemble model has been selected in the final part where the LeNet-5 model along with the optimizers were involved. Then hyperparameters like batch size, bag size, epochs, learning rates were tuned accordingly. Regularization techniques were implemented to avoid overfitting of the model. Dropout, L1, L2, early stopping is some of the regularization techniques. Techniques of bagging, boosting, and stacking can be applied in the case of ensemble models. The trained model is then evaluated with the metrics like accuracy, precision, recall, F1-Score, and confusion matrices. Cross validation can be used to get a better estimate of model performance. For the model's performance factors like computation time, complexity, training time and resources are considered. By visualizing the trained model, visualization can help us to know better about the parameters.

2. Related Works

[1] K. Swetha et al., In their work their main objective is to observe the variations of different algorithms by using the different hidden layers, epochs, and to compare the metrics of those models to choose the best out of them. By comparing the various algorithms like KNN, Regression, the authors concluded that Convolutional Neural Networks (CNN) will be the best algorithm to deal with the images. [2] Apaar Chadha et al., in their handwritten digit recognition using machine learning algorithms, works profoundly to find the patterns with different writing styles. Various algorithms can be used to recognize handwritten digits. It can be concluded that CNN provides maximum accuracy while recognizing/predicting handwritten digits. Accuracy of these traditional CNNs can be improved even more by removing the ensemble features and fine tuning the hyper parameters of the pure CNN architecture. This will also reduce the computational complexities and overall cost of implementing the model. GPU implementation of this technology can also be done alongside CPU implementation to improve efficiency by reducing computation time. Compute Unified Device Architecture (CUDA) on a GPU reduces training time which in turn improves the overall effectiveness of the proposed machine learning model. [3] Ayesha Siddiqa et al., investigated the random forest (RF) and convolutional neural network (CNN) algorithms for handwritten digit recognition on the MNIST dataset. Both the algorithms perform competitively with the state-of-the-art method on MNIST dataset and by comparing the algorithms, they have concluded that the CNN being the fastest. Using the MNIST dataset as a wind tunnel, they have performed some experiments with different preprocessing steps, segmentation, and feature extraction. They proposed an architecture which contains three main components: feature extraction, feature dimension transposition and output layer. The author utilizes the advantage of Dense Net's dense blocks to conduct the feature extraction. They have used a feed forward back propagation neural network having two hidden layers with 54-100-100-38 architecture is used for classification. [4] Fathma Siddique et al., have employed the variations of accuracies for handwritten digit recognition observed for 15 epochs by varying the hidden layers using CNN on the MNIST dataset using TensorFlow in python. They have used seven layered convolutional neural network (CNN) architecture. They have generated accuracy curves for six cases of various combinations of hidden layers for the different parameters and observed the maximum and minimum accuracies. With their observations they have concluded that the maximum accuracy in the performance was found to be 99.21% for 15 epochs in combination of Conv1, pool1, Conv2, pool2 with 2 dropouts and the minimum accuracy in performance was found to be for the combination of Conv1, pool1, Conv2, pool2 with 1 dropout with accuracy of 97.07%. The main conclusion of the paper, CNN shows better performance to attain better image resolution and noise processing. [5] Ritik Dixit et al., have implemented three models (SVM, MLP and CNN) for handwritten digit recognition using MNIST datasets based on both machine learning and deep learning algorithms, and they have compared these model's accuracy and execution speed to conclude the best one among the three models. The SVM was the one of the basic classifiers and that is why the algorithm is faster and it's not possible in the case of complex dataset and ambiguous images, for this case the training accuracy and execution speed get decreased. They have observed that the testing accuracy gets decreased in MLP. The authors have found that the CNN gave the most accurate results for handwritten digit recognition for both training and testing accuracy. So, this makes them conclude that CNN is best suited for any type of prediction that includes images as input data. They have also researched the execution time and concluded the cases which affect the execution time. The proposed condition was increasing the number of epochs without changing the configuration of the algorithm is useless due to limitation of some models and after a certain number of epochs the model starts overfitting the dataset and gives biased prediction as output. [6] Rabia karakaya et al., In their handwritten digit recognition using machine learning, tests were performed on the MNIST dataset with SVM, decision trees, Random Forests, ANN, KNN, K-Means algorithm. The success rates of the algorithms in the field of handwriting recognition were compared. Evaluations were made on the compared machine learning algorithms. The values obtained as a result of the study were compared, as SVM got around 90%, decision tree around 87%, random forest and ANN around 97% , K NN and K-means around 98%. [7]. Anchit Shrivastava et al., In their work they tried to find the best suitable method for the recognition process using KNN, SVM, LeNet-4, LeNet-5, Linear regression, CNN, etc., With these models they tried tuning the hyperparameters by increasing the number of layers, the accuracy and the computational time seems to be increased. Ensemble classifiers achieved high accuracy with least error rate. Tsehay Admassu Assegie, for his handwritten digit recognition, he employed a decision tree classification model for machine learning. The machine learning model's accuracy is 83.4% after being trained on a dataset containing 42000 rows and 720 columns from Kaggle. For recognizing handwritten digits, we employed Decision Tree classifiers and pixels from digit pictures as features vectors. The Kaggle repository was used by us for both the training and

testing datasets, and the results of the experiment demonstrate the effectiveness of the decision tree classifier in recognizing handwritten numbers.

3. Methodology

A. Dataset Acquisition:

MNIST is the widely used dataset in the field of computer vision, pattern recognition and deep learning. It stands for 'Modified National Institute of Standards and Technology'. The dataset consists of handwritten digits from various documents, which includes digits from 0 to 9. Each digit is stored as an image with 28x28 pixel grayscale images. The dataset consists of 60,000 training images and 10000 testing images, where this split is used to evaluate how well a model can generalize from the training data to a new, unseen data. The labels of the digits are of 10 classes from 0 to 9. There are many variations of images in the dataset, including binarized pixels with either 0 or 1, rotated images of digits, etc. The MNIST dataset can be easily accessed from various open-source libraries like TensorFlow, Keras.

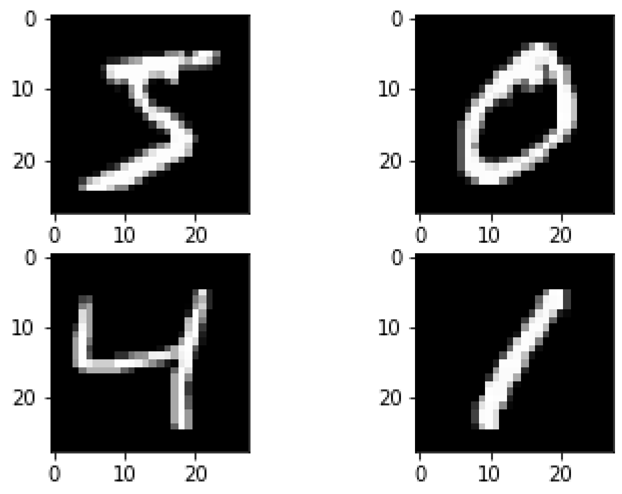

Fig -1: Sample data

B. Data Preprocessing:

Data Preprocessing is a crucial step in Handwritten digit recognition as it should prepare the raw data - handwritten digit into a data compatible for the model to train. This technique is used to increase the quality of the image by denoising it. The main technique which is involved is resizing images to a consistent dimension of 28x28, normalizing the pixel values to grayscale pixels like 0 or 1 by dividing the pixel values by 255, converting the colored images to grayscale images and the augmenting of data by applying random transformations like flips or rotations. In this project, the reshaping of a 3D array of data to 2D array data should be done since the neural network model will expect the 2D array of data. The reshaping is done for both the training and the testing data. Label encoding can also be done numbered from 0 to 9 which is of 10 classes. By preprocessing, the data is being reshaped, augmented, and normalized as per the requirements of the model.

C. Feature Extraction and classification of Digits:

This is the key stage of the project, where these stages are essential to convert the raw data images to meaningful information and to make clear predictions about which digit the given image represents. The data should be preprocessed to extract the features. As the preprocessing is already done by reshaping the dataset, normalizing the images and data augmentation. In the neural networks, feature extraction is integrated with the model itself. Using deep learning models like CNN, which can learn the features from the raw pixels. CNNs have the ability to automatically extract the relevant features during the model training process.

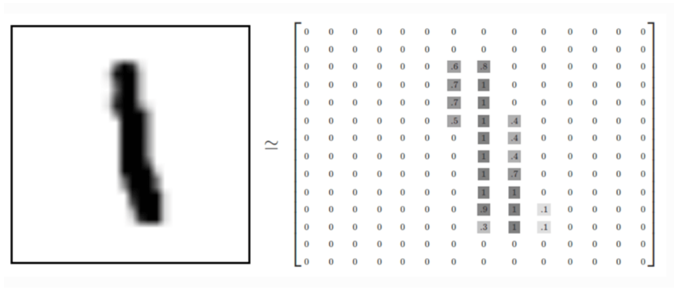

Fig -2: Pixel values

The layers of CNN can do the feature extraction by itself. The first layer is the convolutional layer, where the valuable features from the raw image are extracted. It has several filters which can perform the convolution operation. The output of the convolution layer will be a feature map, which in turn then acts as an input to the ReLU layer, where the pixels are standardized.

$$f(x) = \max(0,x)$$

Fig -3: ReLU formula

$$\sigma(\vec{z})_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}}$$

Fig -4: SoftMax Formula

The third layer is the pooling layer, where the down sampling operation can be done to reduce the size of the feature map. This would result in the pooled feature map. Pooling layer uses filters to identify and extract the different parts of the images like the edges and corners. After the pooling layer, flattening of the image is done, which would result in a single continuous linear vector from the 2D array. And the classification of the image is done by the model's classifier. This is how the usual CNN works. In this project, we implemented an ensemble model with the LeNet-5 base model. The architecture of the model will be starting with a convolutional layer which is 2D, responsible for learning the features in the images. It could have 6 kernels or filters to extract the feature map. And the extracted feature map is then sent into the ReLU layer, which induces the non-linearity in the model. It expects the grayscale 2D array of pixels as the input. The convolutional layer is then followed by the MaxPooling layer. MaxPooling reduces the dimensions of the feature map with a 2x2 pooling window or pooling kernel. After the pooling layer, another convolutional layer is added with the same ReLU activation function with 16 filters or kernels. After this another MaxPooling layer is added to reduce the dimensions. Then the flattening layer is added which transforms the 2D features into 1D vector where this also prepares the data for the fully connected layer. This 1D vector is given into a dense layer with the ReLU activation function and then another dense layer with the same ReLU activation function with a different number of neurons. The last dense layer is the output layer which classifies the digit into 10 classes so it consists of 10 neurons, the activation function is SoftMax, commonly used in the output layer also for the multiclass classification problems. Like this about ten base models are trained with different parameters like epochs, batch size to form the ensemble model. For each base model compilation, Adam optimizer is used as they can improve the performance of the neural network. Since it is a multi-class classification problem, the loss parameter is calculated with categorical cross entropy. Also, early stopping callback is defined, which could monitor the validation loss during the training and stops if the loss doesn't improve for the specified number of epochs, which is termed as 'patience'. Then the base model is trained with the specified hyperparameters where 20% of the training data is randomly sampled. After training the base models, predictions on the test set of data is done, which results in collection of predictions.

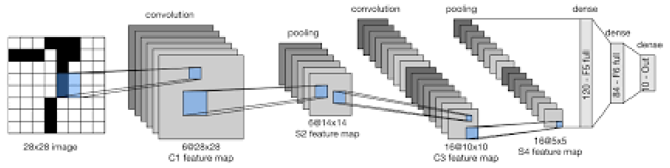

Fig -5: Architecture of LeNet - 5 (Architecture 1)

After training with the Adam optimizer, it switches to the SGD (Stochastic gradient descent) optimizer by recompiling the model. It fine tunes the model by exploring the optimization strategies. Here the fusion of the optimizers has been implemented which in turn increase the accuracy of the model. For the ensemble prediction, majority voting is used to combine the predictions from all the base models, the class which receives the most votes from the base model can be selected as a final prediction.

$$\text{Precision} = \frac{TP}{TP + FP}$$

$$\text{Recall} = \frac{TP}{TP + FN}$$

Fig -6: Formula for precision and recall

$$F1\ score = 2 * \frac{Precision\ * Recall}{Precision + Recall}$$

Fig -7: Formula for F1 Score

Then the final evaluation is done by comparing the predictions of ensemble predictions with the true labels. The various evaluation parameters are involved in comparing the performance of the models. This is referred to as accuracy of the model, precision, recall, f1 score, where the accuracy is 99.15%, precision is 98.87%.

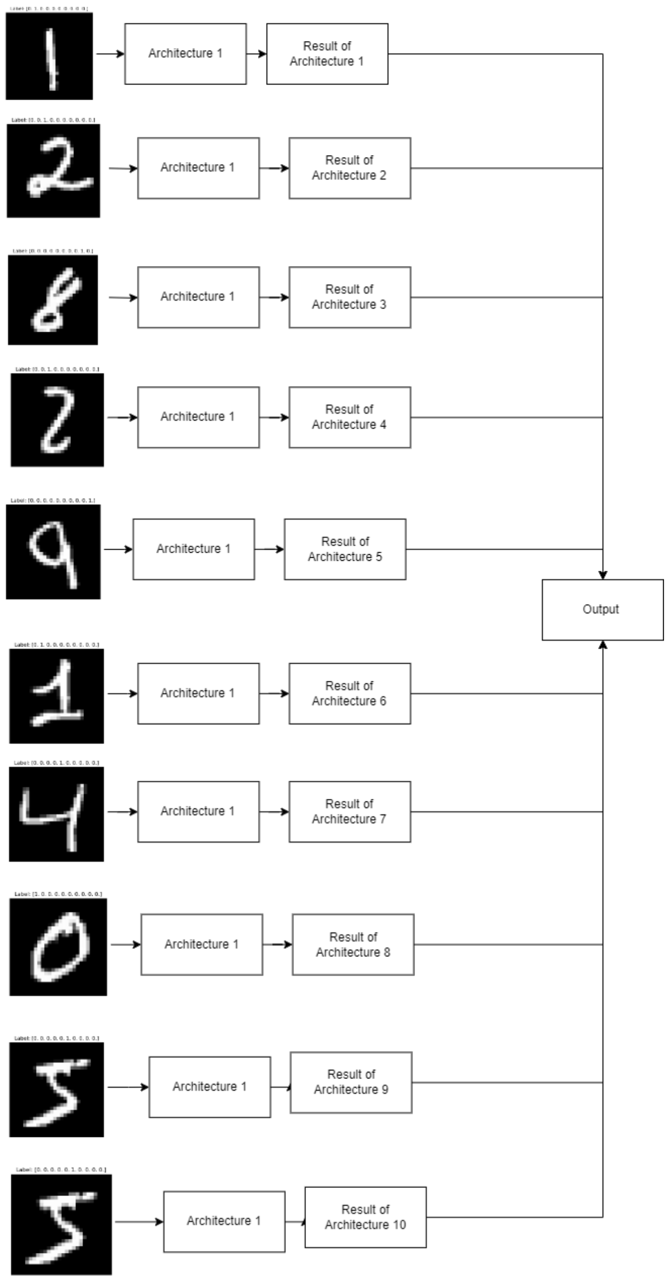

Fig -8: Proposed Model - Architecture

Table -1: Metrics Comparison Table

Model	Accuracy	Precision	Recall	F1 Score
LeNet-5	98.52%	0.9890	0.9890	0.9890
LeNet-5 - Adam	98.97%	0.9882	0.9881	0.9881
LeNet-5 - SGD	98.76%	0.9861	0.9861	0.9861
LeNet-5 - Adam - SGD (Ensemble)	99.15%	0.9887	0.9887	0.9887

The above table illustrates the difference between the model's evaluation metrics independently.

Charts:

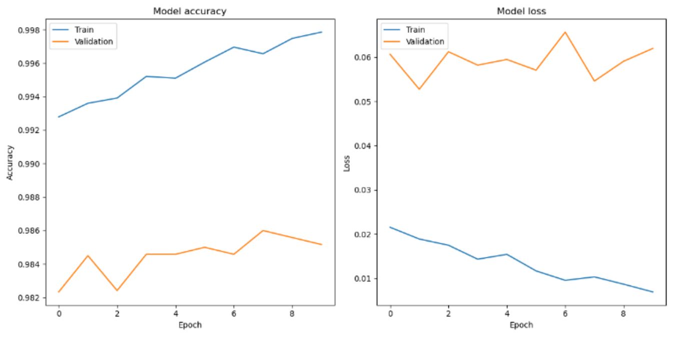

Fig -9: LeNet – 5 (Model Accuracy and Loss)

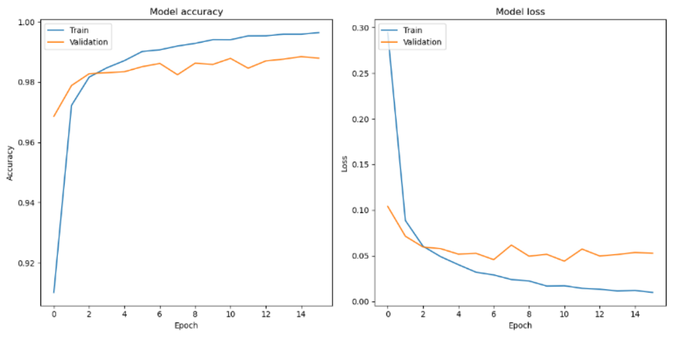

Fig -10: LeNet – 5 with Adam (Model Accuracy and Loss)

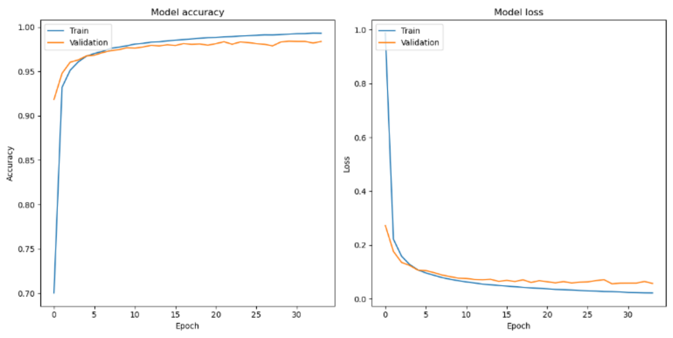

Fig -11: LeNet – 5 with SGD (Model Accuracy and Loss)

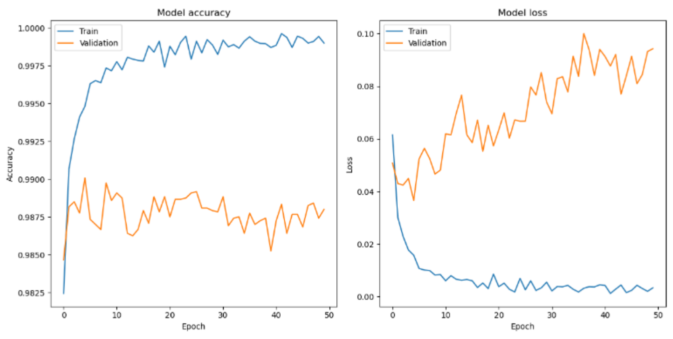

Fig -12: LeNet – 5 with Adam and SGD (Ensemble) (Model Accuracy and Loss)

The above graphs illustrate the training and validation accuracy and the loss of the model.

4. CONCLUSIONS

In computer vision and deep learning, handwritten digit recognition is a well-known issue. It entails teaching computer software how to recognize and categorize handwritten numerals, which typically range from 0 to 9. It's essential to correctly preprocess the dataset before training a model for this task. Resizing photos, standardizing pixel values, and cleaning noisy data are some examples of preprocessing procedures. To guarantee that the models can learn efficiently and perform properly, these steps are crucial. CNNs are a kind of deep learning architecture that have excelled in jobs involving images. They are effective at recognizing handwritten digits because they can automatically identify pertinent elements from the data. Convolutional layers are used by CNNs to identify patterns and feature hierarchies in images. LeNet-5, VGG16, and ResNet are three particular neural network architectures that are mentioned in the paragraph. In the field of computer vision, these models are well-known and frequently applied. The accuracy of digit identification can be considerably impacted by selecting the proper model architecture. Optimizers are techniques used in neural network training to reduce the loss function and increase the precision of the model. According to the paragraph, choosing the right optimizers can improve recognition accuracy. To enhance overall performance, ensemble learning combines the predictions of various machine learning models. One such collective The technique is bagging. According to the passage, using ensemble learning approaches can help the models perform better on digit identification tests. To enhance model performance, this idea calls for mixing various optimizers during model training. The model may be able to attain more accuracy by combining optimizers as opposed to utilizing conventional optimization methods. Overfitting, where the model gets overly specialized in learning the training data and performs badly on fresh, unknown data, is one potential problem with merging optimizers. The use of early stopping callbacks to solve this issue. Early stopping is a strategy that prevents overfitting by halting training when the model's performance on a validation dataset begins to deteriorate. The text emphasizes how moving from a traditional architecture to an ensemble model with a unified optimizer can lead to increased accuracy. This shows that handwritten digit identification is a discipline that is always changing as researchers and practitioners investigate new methods to enhance model performance.

5. REFERENCES

1. K. Swetha, Y. Hithaishi, N.L. Tejaswini, P. Parthasaradhi, P. V. Venkateswara Rao, "Handwritten digit recognition using OpenCV and CNN's, International Journal of Creative Research Thoughts(IJCRT) ,2021 IJCRT | Volume 9, Issue 6 June 2021 | ISSN: 2320-2882
2. Apaar Chadha, Gaurav Yadav, Keshav Ahlawat "Handwritten digit recognition system using machine learning" e-ISSN: 2582-5208.
3. Ayesha Siddiqa, Chakrapani D S "A recognition system for handwritten digits using cnn and random forest", International Journal of Creative Research Thoughts (IJCRT), Volume 10 Issue 10, October 2021,10.21275/SR211005113042 ISSN: 2319-7064.
4. Fathma Siddique, Shadman Sakib , Md. Abu Bakr Siddique, "Recognition of handwritten digit using CNN in python with TensorFlow and comparison of performance for various hidden layers".
5. Ritik Dixit, Rishika Kushwah, Samay Pashine, "Handwritten Digit Recognition using Machine and Deep Learning Algorithms" arXiv:2106.12614v1[cs.CV] 23June 2021.
6. Rabia Karakaya, Serap Cakar, "Handwritten digit recognition using Machine learning", Sakarya University Journal of Science 25(1), 65-71, 2021.
7. Anchit Shrivastava, Isha Jaggi, Sheifali Gupta, Deepali Gupta, "Handwritten Digit Recognition Using Machine Learning: A Review", 2019 2nd International Conference on Power Energy, Environment and Intelligent Control (PEEIC), 978-1-7281-1793-5/19/$31.00 ©2019 IEEE
8. Xiao-Xiao Niu n, Ching Y. Suen (April 2012), "A novel hybrid CNN–SVM classifier for recognizing handwritten digits", Elsevier, Vol. 45, Issue 4, Pages 1318-1325.
9. Dan ClaudiuCires ̧an, Ueli Meier, Luca Maria Gambardella, Jurgen Schmidhuber (March 2010), ̓Deep, Big, Simple Neural Nets for Handwritten Digit Recognition", arXiv, pp. 1-14.
10. Y. LeCun, L. D. Jackel, B. Boser, J. S. Denker, H. P. Graf, I. Guyon, D. Henderson, R. E. Howard, W. Hub, "Handwritten Digit Recognition : Applications of Neural Network Chips and Automatic Learning" NATO ASI series F: Computer and system sciences, Vol. 68, pp. 41-46.
11. T Siva Ajay (July 2017), "Handwritten Digit Recognition Using Convolutional Neural Networks" International Research Journal of Engineering and Technology (IRJET), Vol. 04, Issue 07, pp. 2971-2976.
12. Li Deng (November 2012), "The MNIST Database of Handwritten Digit Images for Machine Learning Research", Best of the web series, IEEE signal processing magazine, pp. 141-142.

13. Rafael M. O. Cruz, George D. C. Cavalcanti and Tsang Ing Ren (2010), "Handwritten Digit Recognition Using Multiple Feature Extraction Techniques and Classifier Ensemble." 17th International conference on systems, signals and image processing (IWSSIP), pp. 215-218.
14. Yoshihiro Shima, Meisei, Yumi Nakashima, Michio Yasuda, Meisei (2017), "Pattern Augmentation for Handwritten Digit Classification based on Combination of Pre-trained CNN and SVM", 6th international Conference on informatics, Electronics and vision (ICIEV) and 7th International Symposium on Computational medical and health technology (ISCMHT).
15. Matthew Y.W. Teow Artificial Intelligence Lab (21 October 2017), "Understanding Convolutional Neural Networks Using A Minimal Model for Handwritten Digit Recognition", 2017 IEEE 2nd International Conference on Automatic Control and Intelligent Systems (I2CACIS 2017), Kota Kinabalu, Sabah, Malaysia, pp. 167-172.
16. Dan ClaudiuCires ̧an, Ueli Meier, Luca Maria Gambardella, Jurgen Schmidhuber (March 2010), ̈Deep, Big, Simple Neural Nets for Handwritten Digit Recognition", arXiv, pp. 1-14.
17. Vineet Singh, Sunil Pranit Lal, "Digits recognition using single layer neural Network with principal component analysis", Computer Science and Engineering (APWC on CSE), 2014 Asia-Pacific World Congress IEEE, 4-5 Nov 2014.
18. Retno Larasati, Hak Keung Lam, "Handwritten digits recognition using ensemble neural networks and ensemble decision tree", Smart Cities, Automation's Intelligent Computing Systems (ICON-SONICS), 2017 International Conference on, 8-10 Nov. 2017.
19. L. Bottou, C. Cortes, "Comparison of Classifier methods a case study in handwritten digit recognition", Pattern Recognition, 1994. Vol. 2 Conference B; Image Processing, Proceedings of the 12th IAPR International. Conference IEEE, 06 August 2002.
20. Hongjian Zhan, Shujing Liu, Yue Lu Shanghai (August 2018), "Handwritten Digit String Recognition using Convolutional Neural Network", 24th International Conference on Pattern Recognition (ICPR), pp. 3729-3734.
21. Seiichi Uchida, Shota Ide, Brian Kenji Iwana, Anna Zhu (2016), "A Further Step to Perfect Accuracy by Training CNN with Larger Data", 15th International Conference on Frontiers in Handwriting Recognition, 405-410
22. Alessandro Giusti, Dan C. Cires□an, Jonathan Masci, Luca M. Gambardella, Jurgen Schmidhuber (2010), "Fast image scanning with deep max-pooling convolutional neural networks" Published in 17th IEEE conference on image processing, pp. 4034-4038.
23. Vladimir Golovko, MikhnoEgor, AliaksandrBrich, and AnatoliySachenko (October 2016), "A Shallow Convolutional Neural Network for Accurate Handwritten Digits Classification" 13th international conference, PRIP, Minsk, Belarus, pp.77-85
24. D. Bouchain, 'Character Recognition using convolutional neural networks', Institute for neural information processing, 2006.
25. Liu, C., Fujisawa, H., "Handwritten digit recognition: Benchmarking of state-of-the-art techniques," Pattern Recognition, 2003.