Study and Develop a Convolutional Neural Network for MNIST Handwritten Digit Classification

Chapter · July 2022

DOI: 10.1007/978-981-19-1142-2_32

6 authors, including:

Brijeshkumar Y. Panchal
Sardar Vallabhbhai Patel Institute of Technology, Vasad (GTU)

Study and Develop a Convolutional Neural Network for MNIST Handwritten Digit Classification

Disha Jayswal, Brijeshkumar Y. Panchal, Bansari Patel, Nidhi Acharya, Rikin Nayak, and Parth Goel

Abstract The goal of this analysis has been on the development of handwritten digit recognition with the use of the MNIST dataset. In the latest days, the identification of handwritten digits has become a challenging research topic in machine learning. Due to physically formed digits having varying lengths, widths, orientations, and positions. It may be utilized in several ways, such as the amount and signature on bank checks, the location of postal and tax papers, and so on. This research used CNN for recognition. Total four steps followed by pre-processing, feature extraction, training CNN, classification, and recognition. Along with its great higher accuracy, CNN outperforms other methods in detecting essential characteristics without the need for human intervention. On top of that, it incorporates unique levels of convolution and pooling processes. Through CNN, 97.78% accuracy was obtained.

Keywords Convolutional neural network (CNN) · Handwritten digit recognition · MNIST dataset · Neural network

D. Jayswal · B. Y. Panchal (✉) · B. Patel · P. Goel
Department of Computer Science and Engineering, Faculty of Technology and Engineering (FTE), Devang Patel Institute of Advance Technology and Research (DEPSTAR), Charotar University of Science and Technology (CHARUSAT), Changa, India
e-mail: panchalbrijesh02@gmail.com

N. Acharya
Department of Computer Engineering, Faculty of Technology and Engineering (FTE), Devang Patel Institute of Advance Technology and Research (DEPSTAR), Charotar University of Science and Technology (CHARUSAT), Changa, India

R. Nayak
V. T. Patel Department of Electronics and Communication Engineering, CHARUSAT Space Research and Technology Center, Charotar University of Science and Technology, (CHARUSAT), CHARUSAT Campus, Changa 388421, India

P. K. Singh et al. (eds.), *Proceedings of Third International Conference on Computing, Communications, and Cyber-Security*, Lecture Notes in Networks and Systems 421,
https://doi.org/10.1007/978-981-19-1142-2_32

1 Introduction

Handwritten digit identification is a difficult task. The aspect that complicates the situation is the natural diversity in syntaxes at various times. As a result, developing a general recognition system process of tracking numbers and compositions produced by a variety of authors is also not possible. Yet, one of the really difficult issues for this assignment is identifying the most useful characteristics with the strong discriminating capacity to increase accuracy rate while minimizing potential. This is an important task for which conventional databases exist, allowing alternative techniques to be tested and verified.

CNN has recently emerged as one of the useful approaches, gambling a key function in more than a few of new fulfillment and traumatic ML knowledge of packages consisting of mission ImageNet item identification, photo segmentation, and face recognition. As a result, here researcher selected CNN for handwritten digit recognition.

Recognition is recognizing or differentiating an object or a person from previous experiences or learning. So, by this, it can be made out easily that handwritten recognition is recognizing or identifying the digits of any document [1]. The MNIST handwritten digit classification problem is a well-known dataset utilized in computer vision and deep learning.

The MNIST dataset is utilized as a database of different handwritten digits. MNIST is a vast database of handwritten numeric or digits that are utilized to train and test machine learning algorithms. The training and testing images in this dataset total 60,000 and 10,000 photos, respectively.

The following are how the paper is structured: Sect. 2 has CNN modeling for the classification of the handwritten digit. Section 3 has MNIST dataset, Sect. 4 has literature survey, and Sect. 5 has experimental results and discussion, then conclusion and future work.

2 CNN Modeling for Classification of Handwritten Digits

In artificial intelligence, CNNs are a type of feed-forward neural network frequently employed for image recognition. CNN takes data in the form of multidimensional arrays as input. It performs admirably when dealing with enormous amounts of tagged data. The receptive field is what CNN uses to extract every piece of the input image. It assigns weights to each neuron based on the significance of the receptive field. As a result, each neuron can tell itself apart from the others. The architecture of layers in CNN.

In Fig. 1, we can see a simple CNN model 1. The input layer is the initial layer, with a 28-by-28-pixel input image. Then there's the convolution layer, which may combine with the input image to produce four feature maps. The pooling layer is the third layer. It calculates the input feature maps' local average or maximum. The next

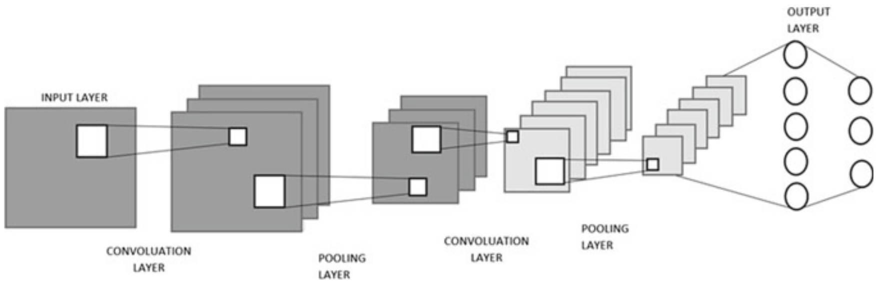

Fig. 1 Convolutional neural network

repeated except for the number and size of convolution kernels, the convolution layer, and the pooling layer work similarly to the preceding ones. Eventually, the output layer is a fully linked layer where the classifier's outcome is the output neurons' largest value [2].

2.1 Advantages of CNN

CNN is widely used replacing various other algorithms these days. As CNN works way better than various other algorithms because of its high computational efficiency, detects [3] the important features without any human supervision. Upon all this, it uses unique layers of convolution, and pooling operations have been inculcated.

3 Dataset MNIST

The MNIST dataset, which was published by Y. LeCun of New York University's Courant Institute, stands for Modified National Institute of Standards and Technology. It's made up of 60,000 squares of 2828-pixel grayscale [4] handwritten numbers ranging from 0 to 9. This dataset contains 60,000 training images and 10,000 testing images. For the test set, more than 250 different writers were picked for handwritten [5] data samples (Fig. 2).

4 Literature Survey

See Table 1.

Fig. 2 MNIST dataset

5 Experimental Result and Discussion

The basic idea of this paper is to get the best accuracy possible with the CNN algorithm. Hence, there is a long procedure [14] that acts behind it consisting of various steps as shown in Fig. 3.

The above figure illustrates the architecture diagram of the proposed system. It contains four stages starting from taking the input dataset and ending with giving the output of the recognized digit. The four stages are as follows: [1]

A. Pre-processing
B. Feature extraction
C. Training CNN
D. Classification and recognition.

A. Pre-processing

Various tasks on the input image must be completed during this pre-processing step. It is defined in such a way that binarization converts a grayscale image to a binary image [5].

Essentially, the training set photos will be thresholded into a binary image to reduce the amount of data [15].

B. Feature Extraction

Following the conclusion of the pre-processed images are now represented in matrix form, which includes pixels from extremely large images. It aids in obtaining the necessary digit information from photos. Feature extraction is the term for this activity. The data redundancy is removed at this stage [1].

C. Training CNN

Training starts from the very first layer which is the input layer where the MNIST dataset is a monochromic picture with the 28×28 size is taken. Then there's the convolution layer, which may combine with the input image to produce four feature maps. Next is the pooling layer, in which the pooling computation [16] will reduce the extension of the data.

Table 1 Literature survey

S. No.	Title	Year of publication	Method or algorithm or techniques	Accuracy (%)
1	An Efficient CNN Model for Automated Digital Handwritten Digit Classification [6]	April 2021	CNN architectures (training and validation), MNIST dataset	99.93
2	Comparative Analysis of Algorithms Used in Handwritten Digit [7]	June 2021	Decision tree, logistic regression, *k*-nearest neighbors (KNN), and deep learning algorithm CNN	86.6, 92.6, 96.89, 99
3	Handwritten Digit Recognizer using Deep Neural Network [7]	April 2021	Deep neural network	99.19
4	Convolutional neural network-based ensemble methods to recognize Bangla handwritten character [8]	June 2021	CNN	98.68
5	Evaluating Machine Learning Models for Handwriting Recognition-based Systems under Local Differential Privacy [9]	2021	Machine learning models	97
6	Hybrid CNN-SVM Classifier for Handwritten Digit Recognition [10]	2020	Convolutional neural networks (CNN) and support vector machine (SVM)	99.28
7	Handwritten Digit Recognition of MNIST dataset using Deep Learning state-of-the-art Artificial Neural Network and CNN (CNN) [11]	2021	Deep learning state-of-the-art artificial neural network (ANN) and convolutional neural network (CNN)	80

(continued)

Table 1 (continued)

S. No.	Title	Year of publication	Method or algorithm or techniques	Accuracy (%)
8	Handwritten Digit Recognition with Feed-Forward Multi-Layer Perceptron and Convolutional Neural Network Architectures [12]	2020	Feed-forward multi-layer perceptron and convolutional neural network architectures	97.44, 98.76
9	Handwritten Digit Recognition by Deep Learning for Automatic Entering of Academic Transcripts [13]	2020	Deep learning for automatic entering of academic transcripts	98.01
10	Implementation of CNN for Handwritten Digit Recognition [14]	2020	FPGA implementation of CNN	97.57

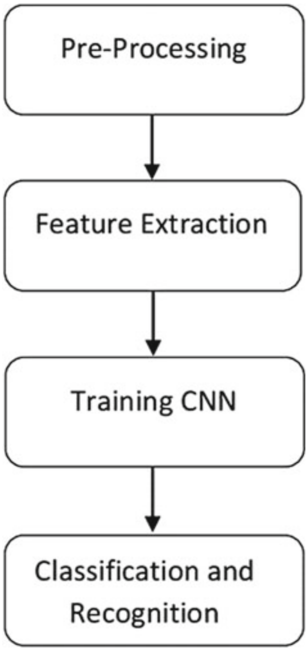

Fig. 3 Block diagram of proposed work

```
Epoch 1/10
118/118 [==============================] - 2s 13ms/step - loss: 1.0223 - accuracy: 0.6974 - val_loss: 0.2267 - val_accuracy: 0.9325
Epoch 2/10
118/118 [==============================] - 1s 11ms/step - loss: 0.2398 - accuracy: 0.9308 - val_loss: 0.1549 - val_accuracy: 0.9528
Epoch 3/10
118/118 [==============================] - 1s 11ms/step - loss: 0.1704 - accuracy: 0.9510 - val_loss: 0.1232 - val_accuracy: 0.9624
Epoch 4/10
118/118 [==============================] - 1s 11ms/step - loss: 0.1301 - accuracy: 0.9617 - val_loss: 0.1037 - val_accuracy: 0.9678
Epoch 5/10
118/118 [==============================] - 1s 11ms/step - loss: 0.1054 - accuracy: 0.9694 - val_loss: 0.0949 - val_accuracy: 0.9702
Epoch 6/10
118/118 [==============================] - 1s 11ms/step - loss: 0.0859 - accuracy: 0.9738 - val_loss: 0.0832 - val_accuracy: 0.9737
Epoch 7/10
118/118 [==============================] - 1s 10ms/step - loss: 0.0736 - accuracy: 0.9778 - val_loss: 0.0828 - val_accuracy: 0.9735
Epoch 8/10
118/118 [==============================] - 1s 10ms/step - loss: 0.0619 - accuracy: 0.9819 - val_loss: 0.0773 - val_accuracy: 0.9760
Epoch 9/10
118/118 [==============================] - 1s 11ms/step - loss: 0.0580 - accuracy: 0.9823 - val_loss: 0.0759 - val_accuracy: 0.9752
Epoch 10/10
118/118 [==============================] - 1s 11ms/step - loss: 0.0446 - accuracy: 0.9865 - val_loss: 0.0726 - val_accuracy: 0.9769
```

Fig. 4 Training CNN and the improved accuracies during each epoch

Again comes another set of convolution layer and pooling [17] layers which have similar operation patterns, except for the fact that the numeral amount and size of convolution kernels. Our final layer that is the output layer which is fully connected layer as the name suggests it combines all the neuron to produce and output where the result of the classifier is the maximum value of output neurons [2].

One model has been constructed; it is required to build and apply it. During the fitting phase, the algorithm will go over the dataset and grasp the relationships. This will educate many more times as specified along with the procedure. We've established ten epochs in our example. Throughout the process, the CNN model will learn as well as making errors. There is indeed a cost for [18] each error made by the model, which is reflected in the lower number for each epoch. In summary, by the conclusion of the last epoch, the model should provide the least amount of losses or as much precision as feasible (Fig. 4).

D. Classification and Recognition

Figure 5 is the tabular representation of the confusion matrix which shows digits 0–9 as class 1–10 respectively that mean class 1 corresponds to 0, class 2 corresponds to 1, class 3 corresponds to 2, and goes on. Vertical rows represent classifier results, and horizontal columns represent our true data. Here classifier results mean the value of digit which is recognized by the classifier and truth data is the actual value of the digit. The diagonally mentioned figure shows the number of correct predictions of the classifier corresponding to true values. Let's understand it better by taking cell of (class 1(0), class1(0)) which gives 970, which means that there are 970 right predictions of zeroes. Now for considering values other than diagonal shows that how many values are predicted as wrong and what digit they have been predicted. Let's understand it better by taking cell of (class 9(8), class 7(6)) which gives 6, it means that true value 8 has been wrongly predicted as 6 for 6 times (Fig. 5).

Figure 6 is the graphical representation of our model accuracy corresponding to the number of epochs taken. The numeral epoch is a hyperparameter that interprets the number of times that the learning algorithm will work through the whole dataset of training. And here we can see a general pattern that as the number of epochs for training and testing dataset increases corresponding to that accuracy also increases.

Truth data

Classifier results	Class 1	Class 2	Class 3	Class 4	Class 5	Class 6	Class 7	Class 8	Class 9	Class 10	Classification overall	User's accuracy (Precision)
Class 1	970	0	8	0	2	2	4	2	3	3	994	97.586%
Class 2	0	1122	2	1	0	0	2	5	1	5	1138	98.594%
Class 3	1	2	1005	2	0	0	0	12	1	1	1024	98.145%
Class 4	0	2	5	990	0	7	1	4	3	3	1015	97.537%
Class 5	0	0	1	0	957	2	2	1	3	8	974	98.255%
Class 6	2	0	1	3	0	868	2	0	1	2	879	98.749%
Class 7	4	3	3	0	11	8	947	0	6	1	983	96.338%
Class 8	1	0	3	5	2	1	0	994	1	5	1012	98.221%
Class 9	1	6	4	6	1	3	0	3	951	7	982	96.843%
Class 10	1	0	0	3	9	1	0	7	4	974	999	97.497%
Truth overall	980	1135	1032	1010	982	892	958	1028	974	1009	10000	
Producer's accuracy (Recall)	98.98%	98.855%	97.384%	98.02%	97.454%	97.309%	98.852%	96.693%	97.639%	96.531%		

Overall accuracy (OA): 97.78%

Kappa[1]: 0.975

Fig. 5 Confutation matrix of CNN classifier

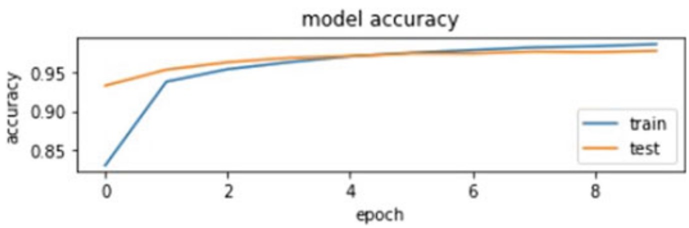

Fig. 6 Model accuracy of proposed work

Figure 7 is the graphical representation of our model loss corresponding to the number of epochs taken. The numeral epochs are a hyperparameter that interprets the number of times that the learning algorithm will work through the whole dataset of training. And here we can see a general pattern that as the number of epochs for training and testing dataset increases corresponding to that loss decreases.

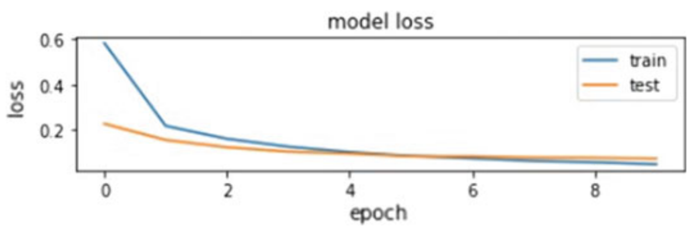

Fig. 7 Model loss of proposed work

6 Conclusion

It was to be found that the algorithm CNN which has given the accuracy of 97.78% works better than various other algorithms. CNN provides tremendous computing efficiency while also detecting significant traits without the need for human intervention. CNN being a special architecture to detect complex features in data gives us the convenience to perform recognition. Upon all this, it also has one of the unique features of combining various convolution and pooling layers which help us to improve accuracy. CNN models can now run on any device, making them globally appealing by combining two layers of convolutions and pooling each.

7 Future Work

Future efforts can take a look at the effectiveness in gaining in-depth knowledge and put in it to greater complicated problems under image recognition. Such that an easy-to-use application can be created for mobile phones or pc which can take in input and recognize it and gives us the identity of the digit input.

The obtained following outcomes can be made way more detailed and accurate by using numerous amounts of convolution layers and a huge number of hidden neurons. And also, accuracy can be increased by using some hybrid model which consists of more than one algorithm combinedly. In the future, this project can be inculcated with real-time data using real-time handwritings of humans.

References

1. Rudraswamimath, V.R., & Bhavanishankar, K. (2019). Handwritten digit recognition using CNN. *International Journal of Innovative Science and Research Technology, 4*, 182–187.
2. El Kessab, B., Daoui, C., Bouikhalene, B., Fakir, M., & Moro, K. (2013). Extraction method of handwritten digit recognition tested on the MNIST database. *International Journal of Advanced Science and Technology, 50*, 99–110.
3. LeCun, Y., Jackel, L., Bottou, L., Brunot, A., Cortes, C., Denker, J., Drucker, H., Guyon, I., Muller, U., Sackinger, E., Simard, P. & Vapnik, V. (1995). Comparison of learning algorithms for handwritten digit recognition. In *International Conference on Artificial Neural Networks*.
4. Abu Ghosh, M. M., & Maghari, A. Y. (2017). A comparative study on handwriting digit recognition using neural networks. In *International Conference on Promising Electronic Technologies (ICPET)* (pp. 77–81).
5. Bohara, M., Patel, K., Patel, B., & Desai, J. (2021, September). An AI based web portal for cotton price analysis and prediction. In *3rd International Conference on Integrated Intelligent Computing Communication and Security (ICIIC 2021)* (pp. 33–39). Atlantis Press.
6. Athila, V. A., & Chandran, A. S. (2021). *Comparative analysis of algorithms used in handwritten digit recognition*.
7. Nikesh, G. S., Amruth, T., Reddy, B. A., Rajashekhar, K., & Surya, N. J. (2021). *Handwritten digit recognizer using deep neural network*.

8. Shahid, A. R., & Talukder, S. *Evaluating machine learning models for handwriting recognition-based systems under local differential privacy*.
9. Ahlawat, S., & Choudhary, A. (2020). Hybrid CNN-SVM classifier for handwritten digit recognition. *Procedia Computer Science, 167*, 2554–2560.
10. Beohar, D., & Rasool, A. (2021, March). Handwritten digit recognition of MNIST dataset using deep learning state-of-the-art artificial neural network (ANN) and convolutional neural network (CNN). In *2021 International Conference on Emerging Smart Computing and Informatics (ESCI)* (pp. 542–548). IEEE.
11. Harikrishnan, A., Sethi, S., & Pandey, R. (2020, March). Handwritten digit recognition with feed-forward multi-layer perceptron and convolutional neural network architectures. In *2020 2nd International Conference on Innovative Mechanisms for Industry Applications (ICIMIA)* (pp. 398–402). IEEE.
12. Nouri, H. E. (2020, October). Handwritten digit recognition by deep learning for automatic entering of academic transcripts. In *Proceedings of the Computational Methods in Systems and Software* (pp. 575–584). Springer.
13. Xiao, R., Shi, J., & Zhang, C. (2020, June). FPGA implementation of CNN for handwritten digit recognition. In *2020 IEEE 4th Information Technology, Networking, Electronic and Automation Control Conference (ITNEC)* (Vol. 1, pp. 1128–1133). IEEE.
14. Hossainm M. A., Ali, M. M. (2019). Recognition of handwritten digit using convolutional neural network (CNN). *Global Journal of Computer Science and Technology: D Neural and Artificial Intelligence, 19*, 27–33
15. Vinjit, B. M., Bhojak, M. K., Kumar, S., & Nikam, G. (2021). Implementation of handwritten digit recognizer using CNN. In *Workshop on Advances in Computational Intelligence at ISIC*.
16. Singh, M., & Rahul (2020). Handwritten digit recognition using machine learning. *International Research Journal of Engineering and Technology (IRJET), 07*, 921–925
17. Biswas, A., & Islam, M. S. (2021). An efficient CNN model for automated digital handwritten digit classification. *Journal of Information Systems Engineering and Business Intelligence, 7*(1), 42–55.
18. Gope, B., Pande, S., Karale, N., Dharmale, S., & Umekar, P. (2021). Handwritten digits identification using mnist database via machine learning models. *IOP Conference Series: Materials Science and Engineering, 1022*, 1–12.