iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 10 **Issue:** VI **Month of publication:** June 2022

DOI: https://doi.org/10.22214/ijraset.2022.44111

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 10 Issue VI June 2022- Available at www.ijraset.com

Classification of MNIST Image Dataset Using Improved Convolutional Neural Network

Neela Chattyopadhyay[1], Hrittika Maity[2], Bipsa Debnath[3], Debanjana Ghosh[4], Rupak Chakraborty[5], Sourish Mitra[6], Rafiqul Islam[7], Nirupam Saha[8]
Department of Computer Science and Engineering, Guru Nanak Institute of Technology, Kolkata, West Bengal, India

Abstract— Convolutional Neural Network (CNN) holds the current research interest in the ever-evolving image classification field. Accurate classifying the image data with minimum of time is highly desired. But the traditional CNN architecture often fails to generate the appropriate outcome for large dataset. So, a modified approach of CNN is proposed here which is the combination of data augmentation and batch normalization embedded with CNN. Now a days identifying or classifying digits accurately with variety of modes is really a task of challenge. The advantages of the proposed approach are noted when it is applied to the popular MNIST dataset used for digit classification. The proposed approach has been compared with some existing techniques and results infer that the validation, training loss and testing accuracy of the proposed approach are more superior as compared to the state-of-art approaches.

Keywords— CNN, Data Augmentation, Batch Normalization, MNIST dataset

I. INTRODUCTION

During contemporary times where digitization has revolutionized the frontier of modern computing and data processing, handwriting recognition systems is an important sector. Extracting information from the various available sources, and converting them into digital formats is an important process, as processing digital data is both cost-effective and less time-consuming. Handwritten recognition systems find applications in advanced fingerprint verification systems used for personal identification and investigation [2]. The fingerprints left on various surfaces are collected as specimens by the security departments to be used in the investigation in crime scenarios. They are used in the identification of automobile license plates, bank documents, archaeological manuscripts, medical prescriptions, and the list goes on [1]. The need for highly accurate, cost and power-efficient recognition systems is hence a necessity.

The influence of deep learning architectures has brought numerous changes in the field of AI and ML. Deep learning architectures mimic the working of the human brain in processing and recognizing data. It involves autoencoders, ANNs, recurrent neural networks, convolutional neural networks, etc. A neural network is a network of artificial neurons or nodes. They have input layers, hidden layers, and an output layer. Each layer contains nodes that are connected to nodes in the previous layer. CNN has proved to be a boon to the recognition technology as its simplistic structure makes it very suitable for image processing and recognition. They successfully without any human supervision detect prominent features of an image through the multiple hidden layers. With the advent of higher computational power and GPU efficiency, convolutional neural networks have now become a powerful means to classify image data and continues to retain their popularity in this field.

CNN is well designed for the processing of data through multiple layered arrays [3]. The defining feature that differentiates CNN from any other ordinary neural network is that CNN gathers input as a 2D array and operates directly on the images rather than focusing on feature extraction. In the past years, almost all state-of-the-art algorithms in the field of image recognition are based on it. The spatial correlations existing within the input data are used by CNN. Each and every synchronic layer of a neural network connects some input neurons. This specific region is called the local receptive field which focuses on the hidden neurons that process the input data inside the mentioned field not realizing the changes outside the specific boundary. Some secured multimedia obtained techniques can also be found in the literature [22-24].

After all, detecting an individual's handwriting is a challenging task. CNN due to its several advantages has been used in several papers to do this task. In some papers, accuracy is as high as 98%-99%. In [4] Morph-CNN, where the counter-harmonic filter is combined with convolutional layers to perform morphological operations resulting in escalated feature maps to perform digit recognition on MNIST and SVHN datasets with accuracies of 99.32% and 92.56% respectively. Different hidden layers are added to CNN to note how different hidden layers affect the whole performance. Epochs are also added to it and different accuracies are noted to recognize handwritten digits in [5-6]. Higher accuracy of 99.21% is achieved in less computation time by improving CNN architecture in[7] using the MNIST dataset and DL4J Framework. CNN architecture is further improved and greater accuracy of 99.87% is gained in [1]. Here, the model is tested with different SGD optimization algorithms, culminating to attain this high recognition accuracy. Models of CNN being applied to vernacular languages are also popular now. In [8] LeNet-5 CNN was applied on MADBase database with Arabic handwritten images for handwritten character

recognition. In [9],92.72% testing accuracy was reached during digit recognition on NumtaDB, a Bangla digit database using CNN. Deep CNN is used in Devanagari handwritten digit recognition using the CPAR-2012 and CVPR-ISI datasets in [10].
The problem of classifying images from MNIST dataset is formulated in Section II. Our CNN model is proposed in Section III. The results observed are discussed in Section IV and our approach is compared is made with other models and other approaches. The conclusion and future scope of the paper is eventually discussed in Section V.

II. PROBLEM FORMULATION

Our objective is to build an efficient CNN model which detects handwritten digits of MNIST dataset with high accuracy and minimal loss as much as possible. Convolutional Neural Network over the years, owing to its versatility has proven to be the best solution to solve this problem.

A. Convolutional Neural Network

Multiple layers form a CNN. We have the input layer at first followed a set of layers finally ending with the output layer. The main layers and their characteristics are described below:

1) *Convolutional Layer:* It is the base layer and main building block of CNN. A filter is placed on the image resulting in a convolved feature map, analogous to watching a specific section of an outdoor scene through a window, which makes specific features prominent. Extracting of image features is the important use of this layer [12]. On convoluting the m × m filter over the n × n input neurons of the input layer, (n-m+1) x (n-m+1) is delivered as output. These filters or kernels help in extracting specific low-level features from the input high-level image. We get the sum of dot products whenever convolution occurs at a (x, y) coordinate in space between the input and 2-D kernel [11]

$$convolution_{x,y} = \Sigma p_i q_i \quad (1)$$

Here p_i refers to the convolutional kernel weights whereas q_i stands for the values of correspondingly spatial extent in the input map (or the output of the preceding layer). Scalar bias will be added to obtain the output for convolutional layer [11]

$$z_{x,y} = convolution_{x,y} + bias \quad (2)$$

2) *Relu:* It is used to add non-linearity in order to gain a generalized solution. The output of the convolution layer is added with a bias value (element wise) and then it is passed through an activation function. It operates on element z, where any negative value is scaled to zero [11],

$$g(z) = z \text{ if } z \geq 0, \quad 0 \text{ if } z < 0. \quad (3)$$

Without it, data being fed into each layer would lose the dimensionality that we want to maintain. It also helps in reducing the vanishing gradient problem [14]. It does not change the size of its input. The other activation functions are sigmoid, tanh, leaky ReLu etc.

3) *Pooling Layer:* This layer reduces the sample size of a particular feature map. It works by reducing the dimension of the sample which helps in increasing the computational performance and it also decreases the chances of over fitting. It also introduces non-linearity. A pooling operator runs on individual feature channels, merges nearby feature values into one, by the application of a suitable operator. Two of the most commonly used pooling techniques are max-pooling and average-pooling [13].
4) *Fully Connected Layer:* For performing classifications there will be a need of several fully connected layers in CNN architecture. In a fully connected layer, the neurons are attached to all the neurons in the preceding layer like any ANN or MLP. Fully connected layers combine features from convolution and from pooling layers to generate a probable class score for the classification of the input images. Nonlinear activation function may be used in a fully connected layer to enhance the performance of the network. At present, the classification layer uses the activation function 'softmax' for classifying the generated features of the input image received from the previous layer into various classes based on the training data.

B. Data Augmentation

Data augmentation is a method to increase the dataset's size as it results in more efficiency to a deep learning model. The traditional approach involves creating new images by performing traditional transformations like rotations, zoom, sheer, shift, coloration, filling, etc., over the images. Involving mostly affine transformations, the original image takes the form [15]:

$$y = Wx + b \quad (4)$$

Increasing the size of data helps the model combat noisy images and overfitting [16]. In [17] unsupervised data augmentation method is used. In Keras, ImageDataGenerator class is used to perform data augmentation and arguments of relevant data augmentation operation are passed through it [18]. Some of the important operations are:

1) *Rotation:* Images are rotated randomly by specifying the degree to the rotation_range argument.
2) *Zoom:* Images are zoomed in and zoomed out by specifying the range value. For zoom in the value would be less than 1 and for zoom out, it would be greater than 1.
3) *Shift:* The Image pixels are moved in either horizontally or vertically. width_shift_range argument is used for the horizontal shift and height_shift_range is used for the vertical shift.
4) *Fill Mode:* There are several points outside of the input image's boundaries. This operation is performed to fill those points by specifying the filling mode and the fill_mode argument is used for it. Default fill mode is 'nearest' which is applied here.

Apart from creating more data, it also helps in the generalization of CNN models so that these trained models can perform better with real-world examples. GAN (Generative Adversarial Networks) is another popular method where styled transformation occurs when the input image is styled with its 6 different image styles [15].

C. Batch Normalization

It is tough to train deep neural networks with many layers. Overfitting might occur. Moreover, the distribution of inputs converts at each layer at the training time. It happens due to the parameter's conversion of previous layers. It decelerates the entire training process. This incident is known as the "*internal covariate shift" [22]. Batch normalization is a method to stabilize the training process via normalizing the inputs at each layer for every training mini-batch. This technique minimizes the epochs' number and increases the model's accuracy.*

In CNN, suppose a layer gets output from preceding layer x. Then the layer applies an affine transformation to achieve y = Wx + b. Let us consider after that it produces an output h(y) which is considered as input of the next layer. Let each component are designated by $y = [y_1, y_2, \cdots, y_n]^T$. Now h(y) can be written as $h(y)= [h(y_1), h(y_2)..., h(y_n)]^T$ [23]. Batch normalization is based on the transformation [23]:

$$\hat{y}_i \leftarrow \frac{y_i - F[y_i]}{\sqrt{F[(y_i - F[y_i])^2]}} \quad (5)$$

Here, $F[y_i]$ *is mean and* $\sqrt{F[(y_i - F[y_i])^2]}$ is standard deviation of random variable y_i. Mean and standard deviation are estimated with a batch of training images at the time of training.

D. Dataset

The MNIST database or Modified National Institute of Standards and Technology database is a subset of a larger dataset available from the NIST database introduced by LeCun [19] in 1998. The images dataset in MNIST database is a combination of two of NIST's databases one of which consist of handwritten digits of high school students and the other dataset contains handwritten digits of employees of the United States Census Bureau.

This dataset contains around 70,000 images in total, out of which contains 48,000 images are being used for training the model, around 12,000 images for validating the model and 10,000 images are being used to test the model. Out of this training data half of the data has been taken from NIST's training dataset and the second half from NIST's testing dataset [20]. A portion of this training set has been used for validation while training our model. The train-validation-test split of this dataset for building our model is described in Table 1.

Table 1. Train-Validation-Test Split.

Split category	No. of images
Train set	48000
Validation set	12000
Test set	10000

Each and every image is grayscale of dimension 28 x 28 pixels, representing the digits 0-9. Each element in this vector is denoted in either zero or one to represent the intensity of the pixels [21].

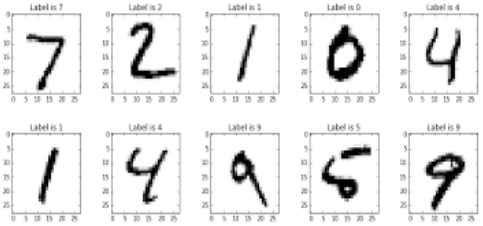


Fig. 1. Image samples from the MNIST database.

III.PROPOSED MODEL

Our proposed architecture consists of 9 layers, with 1 input layer and output layer, 6 inner layers and 1 dense layer as described in Fig 2. The input layer takes 28x28px images from MNIST. Inner layers consist of 2 sets of 2 convolutional layers, each followed by 1 max-pooling layer. The first set of convolutional layers contain 32 filters each and the size of its filter is 3x3. The next set of convolutional layers contain 64 filters each and the size of its filter size is 3×3. The popular rectified linear unit (ReLu) function, helpful in avoiding the vanishing gradient problem [14] is the activation function for all inner layers. The pool size of each of the max pooling layers is 2×2. The dense layer after flattening, has 256 neurons and ReLu activates it. Finally, the softmax activation function is applied over the last dense layer to classify the given input digit image among the 0-9 categories. The "deep-ness" of the model would enable us to extract more prominent features from the MNIST digits and reach a higher testing accuracy. But the deeper the model is, the chances of overfitting also increase. Dropouts of 40% is hence added after every convolution-pooling set and a dropout of 50% is added after the first dense layer. Dropout is a regularisation technique which randomly ignores the given number of nodes and helps to reduce overfitting during training.

To further increase test accuracy Data augmentation (DA) and batch normalization (BN) is added. Data Augmentation applied over the input data diversifies the dataset more and enables the model to learn newer low-level and high-level features. Images have been rotated by 10 degrees, zoomed by 5%, shifted randomly in between -5% to +5% and many more operations are applied. Batch normalization raises the independency of the model, further fulfilling our aim of attaining a higher accuracy. It is added twice, each one after every set of convolutional layers before max-pooling. To arrive at a more accurate good model, we train the model over 4 cases with different combinations of usage of data augmentation and batch normalization as in Table 2. The overall algorithm followed is the following:

Proposed Model Algorithm:

Perform data pre-processing on the MNIST dataset
Perform the train-validation-test split as per section 2.4
for i=1 to 4 /* i is the model for each of the 4 cases */
ClassifyMNIST(i , train data, validation data, test data)
/* 1 - No DA, No BA; 2 – Only DA;
3 - Only BN; 4 - Both DA and BN */
Compare and plot training-validation accuracy of 4 cases
Compare and plot training-validation loss of 4 cases

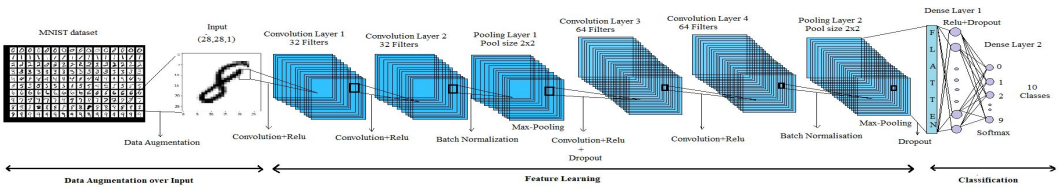


Fig. 2. 9 layered CNN model with both data augmentation and batch normalization.

Adam optimizer, the popular gradient descent optimisation algorithm with cross entropy as its loss function. Being the child of gradient descent with momentum and RMSProp algorithm, it enjoys their mutual benefits resulting in memory-efficiency and high performance of the model.

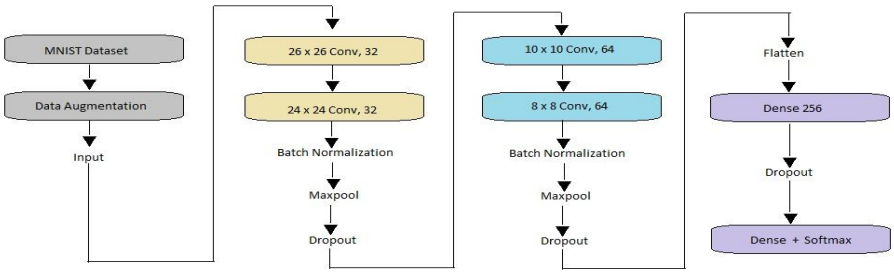


Fig. 3. Flowchart of the working methodology of case 4 with both data augmentation and batch normalization.

IV. RESULTS AND DISCUSSION

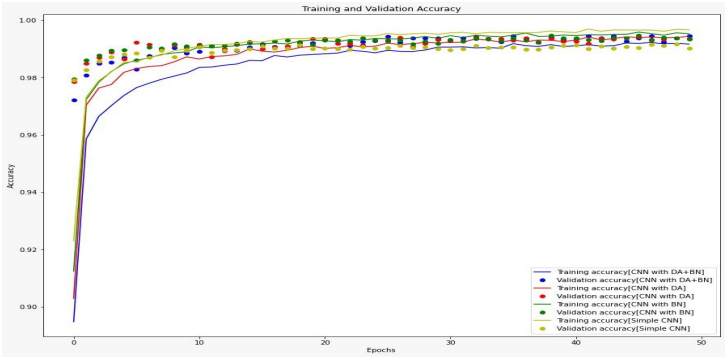
We use the training set to train the model, the validation set tunes the parameters accordingly during the training and finally we check the accuracy of our model with the test set for all the 4 cases. The epochs are 50 and batch size is 32 for all 4 cases. The proposed model was implemented using Keras of the TensorFlow framework on Google Collab.

Table 2. Performance of CNN for 4 different cases.

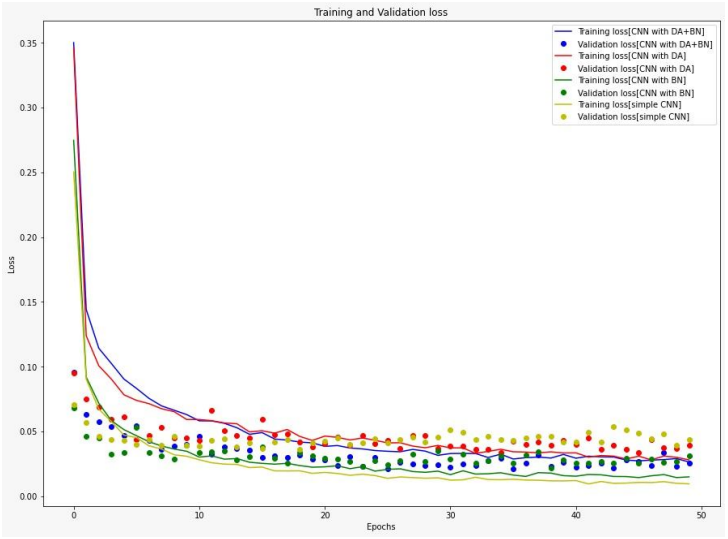
Case	DA or BN	Training Accuracy (%)	Validation Accuracy (%)	Test Accuracy (%)	Training Time (s)
1	No DA, No BN	99.66	99.02	99.20	6688
2	Only DA	99.25	99.20	99.28	6700
3	Only BN	99.31	99.17	99.22	6692
4	Both DA and BN	99.26	99.46	99.68	6693

Table 3. Other CNN models and their approaches.

Paper	Approach	Accuracy (%)
[4]	Morph CNN with counter harmonic filter	99.29
[6]	CNN (TensorFlow Framework)	99.21
[7]	CNN (DL4J Framework)	99.21
Our model	CNN with DA and BN	99.68



(a)



(b)

Fig. 4. (a) Training and Validation Accuracy (b) Training and Validation Loss Graphs of the 4 cases.

We compared the effectiveness of augmentation and normalization techniques both individually as well as combined together in our CNN model as shown in Fig 4 and Table 2. Case 1 shows the results for our bare CNN model without data augmentation and batch normalization. We achieve a high final training accuracy of 99.66% which decreases significantly in validation and test sets. This shows that despite getting high train accuracy the model isn't able to cope similarly with unseen data and higher test loss.

In case 2 we added data augmentation to our previous CNN architecture in case 1 in hopes to diversify the dataset and improve the intuition of the model. We got a 99.25% train accuracy, which is lesser than the training accuracy from case 1 showing we have successfully increased the difficulty/diversity of the dataset. We get a similar validation accuracy of 99.205 which is better than that of case 1 and quite close to the training accuracy of case 2 suggesting a reduction in overfitting. The test accuracy came as 99.28% which is slightly higher than the train accuracy.

Similarly in case 3, we used the batch-normalization technique to modify our case 1 model. Case 1 had signs of overfitting, so we added a normalization layer after each set of two convolution layers. We achieved a training accuracy of 99.31%, again less than the case 1 training accuracy. Validation accuracy was 99.17%, which is better than that of our base model. The final test accuracy was 99.22% which is an improvement from case 1 test accuracy and quite close to train accuracy for case 3 proving that the technique can reduce the overfitting and stabilize the training process to enhance the model's capability to handle unseen.

For case 4 we added both data augmentation and batch normalization to our base CNN model to see if we can combine the desired result from the individual usages. The goal of this test was to get an ideal architecture for digit recognition that goes beyond the usual 98% a simple CNN can achieve in training and make the model more intuitive towards handling unseen data. We sent augmented data into the CNN model and the normalization layers were place after each set of two convolution layers like in case 3. We achieved a training accuracy of 99.28% and a 99.46% overall validation accuracy. The validation accuracy is much better than the previous cases, showing that the training process has stabilized and reduced overfitting. The final test accuracy was found to be 99.68% for case 4, which is quite an improvement over our base CNN model and a bit higher than the training accuracy of the case 1 model. Thus, from the results, DA+BN performed better than the rest so we append it to our model and use it as our proposed CNN.

Finally, in Table 3, we compared our model with three existing state-of-the-art CNN models, where our CNN clearly achieved a superior performance. In [4] MConv layer made using CHM filter and convolution layer is used. Alongside convolution this layer also performs morphology and images resulted from this layer are of better quality. These MConv feature maps along with pooling and dense layers perform the overall classification of MNIST and SVHN benchmark datasets where different combinations of MConv layers are used for different results of dilation and erosion. Their two erosion layers architecture on the MNIST dataset performs the best reaching the highest overall accuracy of 99.29%. In [6] they perform an experimental

comparison with different combinations of convolution, max-pooling, dropout, dense layers. Among the different 3 and 4 layer CNN models, they conclude that their 4-layer CNN model with convolution, pooling, convolution, pooling, dropout, dense, softmax combination works the best with a high overall accuracy of 99.21%. In [7] all operations of CNN modelling, feature extraction and classification is performed using the DL4J Framework. Their model reaches an average accuracy of 99.21%.

V. CONCLUSION

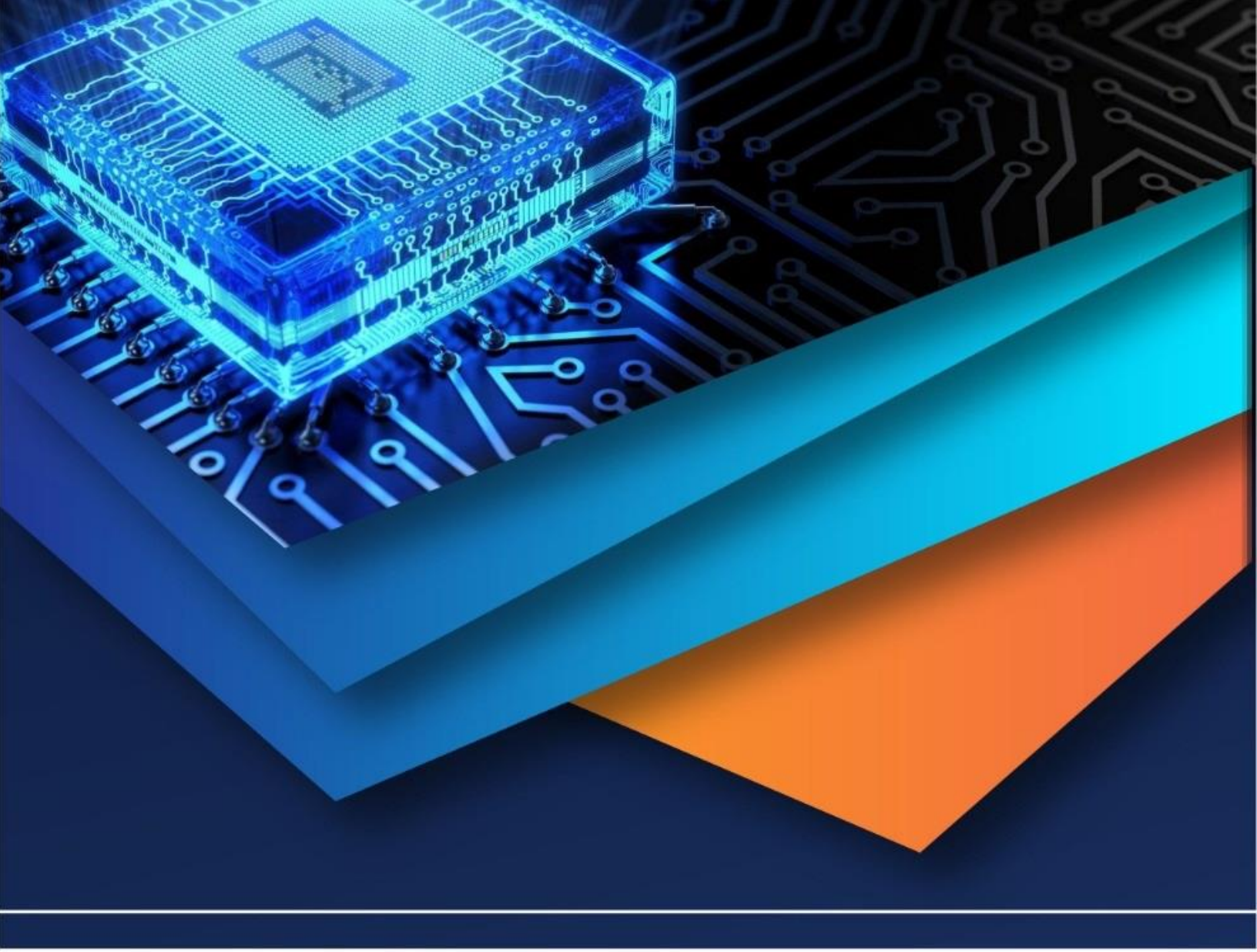
Handwritten digit classification is an important problem in the emerging world of technology, and deep learning's CNN is one of its best solutions. In this paper we successfully finalized an efficient architecture for classifying the MNIST dataset. After comparing the different combinations of data augmentation and batch normalization, we concluded that for a model with 4 convolution layers, with the first 2 layers having 32 filters and the later having 64 filters, the one using both data augmentation and batch normalization performs the best, with a high-test accuracy of 99.68%, higher than many of the existing techniques. Losses of all 4 models were compared as well.

In the future we wish to work on reducing the overall loss of the model as much as possible. Furthermore, we may extend our model to other architectures of CNN like hybrid CNNs, CNN with RNN, genetic algorithms and the other emerging modern techniques. More accurate and efficient CNN models can be built by experimenting with different optimization techniques, convolution layers, batch sizes etc.

REFERENCES

[1] Ahlawat, S., Choudhary, A., Nayyar, A., Singh, S., Yoon, B.: Improved handwritten digit recognition using convolutional neural networks (CNN). Sensors 20(12), 3344 (2020).
[2] Minaee, S., Azimi, E., Abdolrashidi, A.: FingerNet: Pushing the Limits of Fingerprint Recognition Using Convolutional Neural Network. arXiv:1907.12956v1.
[3] Pandya, B., Cosma, G., Alani, A. A., Taherkhani, A., Bharadi, V., McGinnity, T. M.: Fingerprint classification using a deep convolutional neural network. In: 2018 4th International Conference on Information Management (ICIM), pp. 86-91. IEEE (2018).
[4] Mellouli, D., Hamdani, T.M., Sanchez-Medina, J.J., Ayed, M.B., Alimi, A.M.: Morphological Convolutional Neural Network Architecture for Digit Recognition. IEEE Transactions on Neural Networks and Learning Systems 30(9), 2876-2885 (2019).
[5] Arif, R. B., Siddique, M. A. B, Khan, M. M. R., Oishe, M. R.: Study and Observation of the Variations of Accuracies for Handwritten Digits Recognition with Various Hidden Layers and Epochs using Convolutional Neural Network. In: 4th International Conference on Electrical Engineering and Information & Communication Technology, pp. 112-117. (2018).
[6] Siddique, F., Sakib, S., Siddique, M. A. B.: Recognition of Handwritten Digit using Convolutional Neural Network in Python with Tensorflow and Comparison of Performance for Various Hidden Layers. 5th International Conference on Advances in Electrical Engineering, pp. 541-546 (2019).
[7] Ali, S., Shaukat, Z., Azeem, M., Sakhawat, Z., Mahmood, T., Rehman, K.U.: An efficient and improved scheme for handwritten digit recognition based on convolutional neural network. SN Applied Sciences 1, 1125 (2019).
[8] El-Sawy, A., EL-Bakry, H., Loey, M.: CNN for Handwritten Arabic Digits Recognition Based on LeNet-5. In: Hassanien A., Shaalan K., Gaber T., Azar A., Tolba M. (eds) Proceedings of the International Conference on Advanced Intelligent Systems and Informatics 2016. AISI 2016. Advances in Intelligent Systems and Computing, vol. 533, pp. 566-575. Springer (2017).
[9] Shawon, A., Jamil-Ur Rahman, M., Mahmud, F., Arefin Zaman, M. M.: Bangla Handwritten Digit Recognition Using Deep CNN for Large and Unbiased Dataset. In: 2018 International Conference on Bangla Speech and Language Processing (ICBSLP), pp. 1-6. IEEE (2018).
[10] Singh, P., Verma, A., Chaudhari, N.S.: Deep Convolutional Neural Network Classifier for Handwritten Devanagari Character Recognition. In: Satapathy S., Mandal J., Udgata S., Bhateja V. (eds) Information Systems Design and Intelligent Applications. Advances in Intelligent Systems and Computing, vol. 434, pp. 551-561. Springer, New Delhi (2016).
[11] Huynh-The, T., Hua, C., Pham, Q., Kim, D.: MCNet: An Efficient CNN Architecture for Robust Automatic Modulation Classification. IEEE Communication Letters 24, 811-815 (2020).
[12] Saha, C., Faisal, R., Rahman, M.: Bangla Handwritten Digit Recognition Using an Improved Deep Convolutional Neural Network Architecture. In: International Conference on Advanced Information and Communication Technology 2020, IEEE, Dhaka (2020).
[13] Liu, K., Kang, G., Zhang, N., Hou, B.: Breast Cancer Classification Based on Fully-Connected Layer First Convolutional Neural Networks. IEEE Access 6, 23722-23732 (2018).
[14] Lin, G., Shen, W.: Research on Convolutional Neural Network based on improved Relu piecewise activation function. Procedia Computer Science 131, 977–984 (2018).
[15] Wang, J., Perez, L.: The Effectiveness of Data Augmentation in Image Classification using Deep Learning. arXiv:1712.04621v1 (2017).
[16] Mikolajczyk, A., Grochowski.: Data augmentation for improving deep learning in image classification problem. In: *International Interdisciplinary PhD Workshop (IIPhDW) 2018,* pp. 117-122. IEEE (2018).
[17] Shijie, J., Ping, W., Peiyi, J., Siping, H.: Research on data augmentation for image classification based on convolution neural networks. Chinese Automation Congress (CAC) 2017, pp. 4165-4170. IEEE (2017).
[18] Data Augmentation applying methodology, https://machinelearningmastery.com/how-to-configure-image-data-augmentation-when-training-deep-learning-neural-networks/, last assessed 2021/09/18.
[19] LeCun, Y.; Cortes, C.; Burges, C.J.C.: The MNIST Database of Handwritten Digits. (2012).
[20] Kussul, E., Baidyk, T.: Improved method of handwritten digit recognition tested on MNIST database. Image and vision computing 22(12), 971-981 (2004).
[21] MNIST page, https://web.stanford.edu/~hastie/CASI_files/DATA/MNIST.html, last assessed 2021/09/18.
[22] Ioffe, S., Szegedy, C.: Batch Normalisation: Accelerating Deep Network Training by Reducing Internal Covariate Shift. arXiv:1502.03167v3 (2015).
[23] Yuan, X., Feng, Z., Norton, M., Li, X.: Generalized Batch Normalization: Towards Accelerating Deep Neural Networks. Proceedings of the AAAI Conference on Artificial Intelligence, 33(01), 1682-1689 (2019).

[24] S. Namasudra, P. Roy, P. Vijayakumar, S. Audithan, B. Balusamy, "Time efficient secure DNA based access control model for cloud computing environment", Future Generation Computer Systems, 73, 90-105 (2017).
[25] S. Namasudra, R. Chakraborty, S. Kadry, G. Manogaran, B. S. Rawal, "FAST: Fast Accessing Scheme for Data Transmission in cloud computing", Peer-to-Peer Networking and Applications, 14, 2430-2442 (2021).
[26] P. Pavithran, S. Mathew, S. Namasudra, P. Lorenz, "A Novel Cryptosystem based on DNA cryptography and randomly generated mealy machine" 104, 102160(2021).



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY