JOURNAL OF SOFT COMPUTING AND DATA MINING
e-ISSN: 2716-621X

JSCDM

Vol. 5 No. 1 (2024) 79-90
https://publisher.uthm.edu.my/ojs/index.php/jscdm

Handwritten Digit Classification Using Deep Learning Convolutional Neural Network

Eman Ahmed Khorsheed[1]*, Ahmed Khorsheed Al-Sulaifanie[2]

[1] *Computer Science,*
College of Science, Nawroz University, Duhok, IRAQ
[2] *Electrical and Computer Engineering,*
College of Engineering, University of Duhok, Duhok, IRAQ

*Corresponding Author: eman.khorsheed@nawroz.edu.krd
DOI: https://doi.org/10.30880/jscdm.2024.05.01.007

Article Info

Received: 1 December 2023
Accepted: 25 April 2024
Available online: 21 June 2024

Keywords

Handwritten digit recognition (HDR), deep learning, convolutional neural networks (CNNs), classification, and feature extraction.

Abstract

Due to the wide range of handwriting styles among individuals and the low image quality of the handwritten text, accurate handwriting detection has been a difficult challenge in computer vision. This is because static feature analysis of the text images is frequently insufficient to account for these factors. The accuracy of recognizing different handwriting patterns has recently progressively increased because of the introduction of machine learning, particularly convolutional neural networks (CNNs). This study uses various filter sizes to create a deep CNN model to increase the handwritten digit recognition rate. The proposed model's multi-layer deep structure includes a fully linked layer (also known as a dense layer) for classification and one convolution and activation layer for feature extraction. The proposed methodology has an average classification accuracy of up to 99.5% on the MNIST dataset.

1. Introduction

The task of handwriting recognition is a cornerstone challenge in both computer vision and image recognition realms. Crafting algorithms and models that can accurately decipher and classify handwritten characters or symbols remains an ongoing hurdle in today's technology landscape. The versatility of this technology spans a broad spectrum, encompassing critical functions such as the digitization of historical documents and empowering smart devices to interpret and understand user-generated input [1].

Central to this technological advancement is the profound impact of Convolutional Neural Networks (CNNs) [1][2][3], a revolutionary technique inspired by the human brain's visual recognition process. Similar to how humans teach children to recognize objects by exposing them to numerous images, CNNs operate on the premise of automatically extracting hierarchical features from raw pixel data. This unique ability has positioned CNNs as a linchpin in the realm of deep learning, especially in tasks that rely on visual inputs, such as images or handwritten text.

CNNs' unparalleled strength lies in their capability to discern intricate patterns and relationships among pixels. In the context of handwriting recognition, where understanding the spatial arrangement of individual pixels is crucial for accurate interpretation, CNNs excel [4]. By systematically analyzing and learning from vast datasets, these networks can automatically deduce essential features, allowing them to distinguish nuances and variations in handwritten characters with remarkable accuracy.

Moreover, CNNs' hierarchical learning approach enables them to progressively analyze images in multiple layers, extracting increasingly abstract and complex features. This innate ability to learn representations at different levels of abstraction empowers CNNs to understand and categorize handwritten content effectively.

In essence, the prowess of Convolutional Neural Networks lies not only in their ability to autonomously learn from raw data but also in their capacity to unravel intricate patterns and spatial relationships within images. These networks have significantly transformed the landscape of handwriting recognition, establishing themselves as indispensable tools in accurately deciphering and classifying handwritten content.

Recently, there has been an increasing fascination with utilizing contemporary machine learning (ML) techniques for handwriting categorization. These approaches have shown exceptional performance and the ability to apply learned knowledge to new instances. By integrating the aforementioned handwritten traits with conventional classification methods like k-nearest Neighbour (KNN) and histogram of dimensional gradient (HOG) traits [5], the statistical classification model [6], support vector machine (SVM) [7], and clustering [8], the accuracy of classification and recognition has been greatly enhanced. The formative work by Li Deng [9] laid the foundation for modern handwriting recognition using CNNs. They introduced the MNIST dataset and trained a multi-layer neural network, demonstrating the effectiveness of deep learning for handwritten digit recognition. In [10] the authors proposed a combination of deep features and handcrafted for the recognition of handwritten numerals. The researchers in [11] proposed a deep CNN model with a fast-converging rate in training to improve the recognition rate and the overall accuracy of the developed model, which can reach 99.4% on the MNIST dataset. A framework is constructed to identify handwritten numbers, as detailed in [12]. The methodology depends on the extraction of distinctive characteristics and the integration of different classifiers using algebraic fusion. The MNIST dataset is used to train a CNN model to extract features. The results suggest that the fusing of the model reaches a minimum accuracy of 98%.

The authors in [13] used the Deeplearning4j (DL4J) framework to identify handwritten numbers. The most favorable result was attained by using two convolutional layers. The first layer comprises 32 filters with a window size of 5×5, while the subsequent layer utilizes 64 filters with a window size 7×7. Consequently, they have formulated a model that attains a precision rate of 99.21%. This model has exceptional performance in terms of both accuracy and efficiency. Savita et al. [14] conducted a study on various parameters of a CNN to devise an improved approach for accurately identifying handwritten numbers. The elements that need to be considered are the number of layers, kernel size, padding, receptive field, dilution, and stride size. The authors introduced an innovative CNN structure that integrates techniques such as data preprocessing, data augmentation, receptive field, normalization, optimization, and regularization to enhance the precision of recognizing handwritten digits. The result achieved a recognition accuracy of 99.87% for an MNIST dataset. The authors in [15] suggested using the Efficient system to use an enhanced CNN structure for High Dynamic Range (HDR) photos. The suggested framework outperforms state-of-the-art techniques in terms of recognition accuracy with 99.83%. Kim, Y et al. [16] suggested a deep CNN architecture with a rapid convergence rate during training Rectified Linear Units (ReLU) activation function to enhance the accuracy of recognizing the MNIST handwritten digit dataset.

There are numerous CNN classification algorithms in the literature. However, most of these algorithms have not chosen the appropriate filter size. This paper investigates handwriting classification using CNNs with appropriate filter sizes. The purpose is to build a robust and accurate architecture. Moreover, it involves model architecture design, training, and evaluation. This work is expected to contribute to the field of handwriting recognition by highlighting the effectiveness of CNNs in accurately classifying handwritten digits. The trained model's performance on the validation dataset will demonstrate its ability to generalize and handle different writing styles well.

In the subsequent sections, the methodology, architecture design, training process, and results of the CNN-based handwriting classification method are detailed, providing a comprehensive understanding of the journey from raw pixel data to accurate character recognition in sections 2 to 4, respectively. Lastly, section 5 concludes the study.

2. Handwritten Digit Dataset

The recommended architecture is evaluated for classification accuracy using the MNIST (Modified National Institute of Standards and Technology) dataset [17] via many studies. The Digits dataset comprises 10,000 grayscale pictures of handwritten digits. The dimensions of each picture are 28 by 28 pixels, and each image is labeled with a number from 0 to 9 to indicate which digit it represents. Every picture has undergone rotation at a certain angle. Several photos and their corresponding labels are shown in Figure 1. The reason behind choosing the used dataset is that MNIST is a popular benchmark for assessing the performance of various machine learning methods, and it is frequently regarded as a beginner-friendly dataset in the fields of deep learning and computer vision.

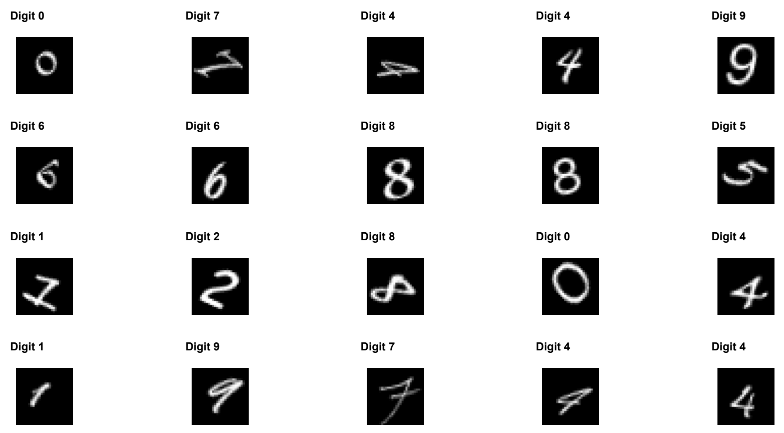


Fig. 1 *28-by-28-pixel grayscale image from the database*

Table 1 presents the distribution of the 10,000 synthetic handwritten digits' dataset.

Table 1 *The distribution of the 10,000 synthetic handwritten digits' dataset*

Label	0	1	2	3	4	5	6	7	8	9
Count	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000

3. Proposed CNN Architecture

This section presented a handwritten digit categorization and recognition model illustrated in Figure 2. This recognition model is based on a deep CNN applied to the MNIST dataset with a feature-mapped output layer. This proposed CNN model classifies digits between 0 and 9. The following subsections provide a full explanation of the suggested model.

The CNN is a biologically inspired trainable machine learning architecture that, like standard multi-layer neural networks, learns via experience. CNNs comprise multiple layers of overlapping tiling groupings of small neurons collaborating to reflect the original image better. CNNs are often used to recognize images and videos. A CNN model comprises an input, output, and intermediary hidden layers. These hidden layers consist of convolutional layers, responsible for extracting features from input data using filters, pooling layers that down-sample the extracted features, and fully connected layers, which perform classification or regression tasks [18]. Figure 2 visually depicts a suggested CNN architecture employing a filter size of n x n, showcasing how this convolutional network structure extracts and processes features from input data for tasks like image recognition and classification.

3.1 Convolution Layer

The core element within a CNN is the convolutional layer. The procedure involves applying convolution operations to specific regions of the input feature maps and generating output feature maps. The 28x28x1 input picture undergoes manipulation using 20 convolutional filters. The 2-D convolutional layer employs sliding convolutional filters to process the 2-D input. The dimensions of the picture output for each filter in the convolutional layer are M x M.

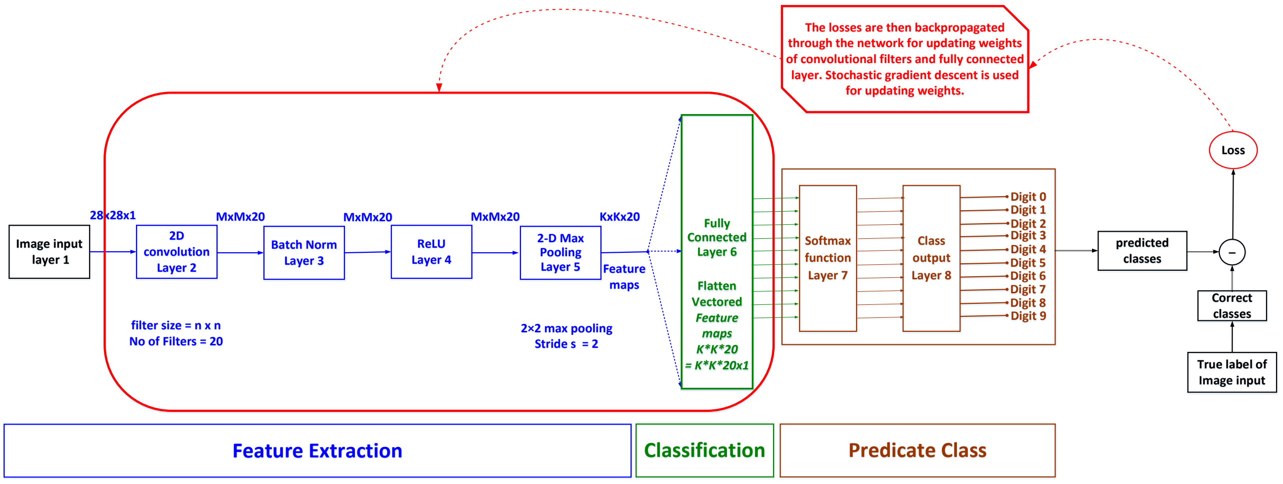


Fig. 2 *Proposed method*

Four filters with different n sizes are utilized. The number of feature maps and filter size are shown in Table 2. The convolution layer extracts most significant features in CNN. The convolution layer operates on the following principle: a sequence of filters with varying weights sweep over the input to extract features, and the output is an activation map (also known as a feature map). It is worth mentioning that the weights in CNN are learned through backpropagation, except initialization, when weights are assigned randomly.

Table 2 *The number of feature maps*

Filter size $n \times n$	**2D convolutional Output:** $M \times M$, $M = 28 - n + 1$	**2x2 Max pooling** L **= pool size =2,** $s = 2$ **Output:** $K \times K$, $K = \frac{M-L}{s} + 1$	**Flatten Feature maps**
3x3	26x26x20	13x13x20	3380x1
5x5	24x24x20	12x12x20	2880x1
7x7	22x22x20	11x11x20	2420x1
9x9	20x20x20	10x10x20	2000x1

3.2 Batch Normalization

The batch normalization layer accelerates CNN training and decreases network initialization sensitivity [19]. The Batch Normalization layer transforms the signal S(k) of length L as follows: first determines the mean μ and the variance σ^2 of the activation values across the batch. Batch normalization uses re-centering and rescaling to normalize the layers' inputs, making neural network training faster and more reliable. It then normalizes the activation vector Sn (k). As a result, the output of each neuron follows a standard normal distribution through the batch.

The BN finally calculates the layer's output So(k) by employing a linear transformation with γ and β, two trainable parameters as in eq.1 [19]:

$$S_o(k) = \gamma\, S_n(k) + \beta \tag{1}$$

Where γ and β are two parameters that can be learned via backpropagation.

This phase enables the model to choose the most favorable distribution for each hidden layer by modifying two parameters: γ, which controls the standard deviation correction, and β, which allows for bias adjustment, pushing the curve either to the right or left side. The suggested CNN architecture incorporates batch normalization before the activation function (ReLU).

3.3 Rectified Linear Unit

When the input is negative, as seen in eq.2 [20], the ReLU output is 0. ReLU instead returns the same value x, when the input is positive:

$$f(x) = f(x) = \begin{cases} 0, \; for \;\; x < 0 \\ x, \; for \;\; x \geq 0 \end{cases} \tag{2}$$

The rectified linear function introduces non-linearity to the neural network [21]. The ReLU layer employs a threshold operation on each input element, assigning a value of zero to every element that is less than zero. The batch normalization and ReLU layers have no impact on the input dimensions.

3.4 Max-pooling Layer

The process of down-sampling occurs through a 2-D max-pooling layer, which partitions the input into rectangular regions for pooling and subsequently identifies the maximum value within each region [22]. The pooling regions move with a step size, referred to as the stride (s), which is fixed at a value of 2. Following the application of the 2-D max-pooling layer, the image size becomes M x M, resulting in a feature map of dimensions KxKx20 for this specific 2-D max-pooling layer.

3.5 Fully Connected Layer

A fully connected neural network is one in which each neuron performs a linear transformation to the input vector using a weights matrix [20]. Consequently, every input of the retrieved features affects every output of the output vector, as all potential connections between layers are present. In this manner, the fully connected layer performs multi-classification based on these extracted features. The fully connected layer receives the output feature maps from the 2-D max-pooling layer, which are flattened before being fed into it. The *KxKx20* feature maps transformed into a one-dimensional flattened array of *K*K*20 x 1* local features. A fully connected layer, multiple a learned weight matrix *Wij* of dimension *K*K*20 x 10* by input array flatten K*K*20 x 1 and then adds a bias vector. Therefore, the output of a fully connected layer is the vector of size *1x10*.

3.6 Softmax Function

The softmax layer receives the output from the fully connected layer and transforms it into probabilities. In a multi-class problem, softmax assigns each class to a decimal probability, and the sum of these probabilities is equal to 1.0. This permits the output to be interpreted directly as a probability while each output value is between 0 and 1. In other words, the softmax function can rescale the elements of an N-dimensional input tensor to a zero-to-one range and sum to one. The function of softmax activation can be mathematically represented in eq.3 [23]:

$$\theta(Z)_i = \frac{e^{Z_i}}{\sum_{k=1}^{M} e^{Z_k}} \tag{3}$$

where z_i refers to the weighted sum of the i-th output neuron, while M is the number of output neurons.

The final outputs are classified into 10 MNIST dataset categories (numbers 0 to 9) by the class output layer, with each category corresponding to a different probability distribution.

For training, the true label is used to join the predicted classes to calculate loss using the object function. Backpropagation of the losses across the network is then used to update the convolutional filter weights and fully connected layers. The moment stochastic gradient descent (msgd) is used to update weights.

4. Experimental Results

In this paper, the Digits dataset from Matlab is used. It contains 10,000 synthetic grayscale images of handwritten digits. Each image is 28 × 28 pixels (784 features) and is labeled with the digit it represents (0-9). A specific angle has rotated each image. The used Digits dataset was divided into 8000 grayscale image samples (images) for training (60%) and 2000 for validation (testing) (40%).

The training results of the proposed CNN architecture are performed using 20 filters in the convolution layer to generate feature maps. The size of filters is set to 3x3, 5x5, 7x7, and 9x9 with padding zero p=1. A single-size filter is selected for the training process.

Fig.3 shows the image processing of the convolution layer in the case of a 5x5 filter. The feature map consists of 24x24x20 pixel images. This figure shows the various changes to the input image (Digit 8) caused by the convolution filter.

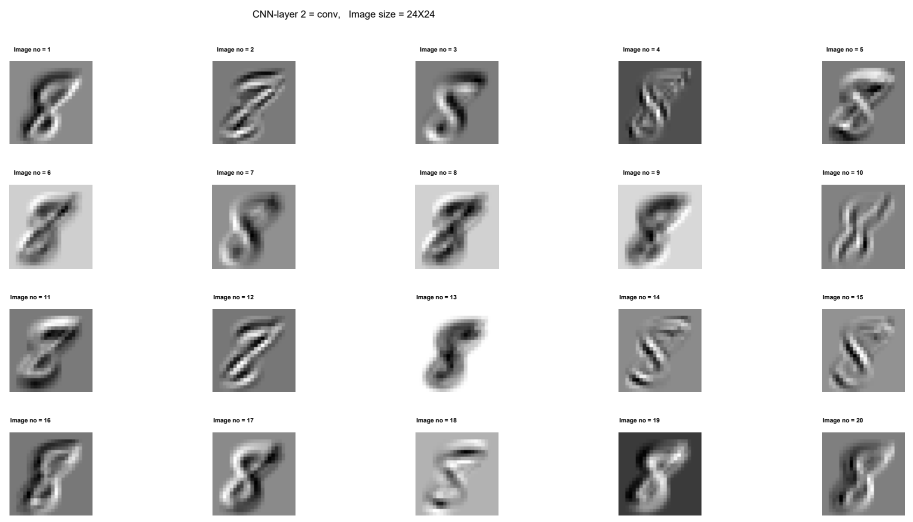


Fig. 3 *Image processing of convolution layer results with padding 1 for 5x5-filter size*

As illustrated in Fig.4, the ReLU function is then applied to the feature map from the convolution layer.

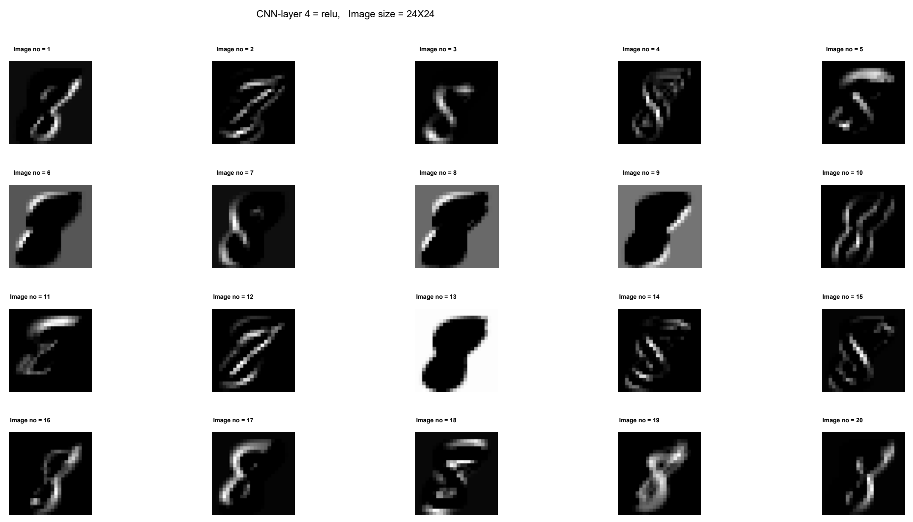


Fig. 4 *The results of image processing of the ReLU layer on the feature map from previous layers for 5 x 5 filter size*

The max-pooling technique generates reduced-resolution feature maps that retain the most crucial features from the original feature map. Hence, max pooling reduces the number of parameters, lowers computational cost, and prevents overfitting. In the proposed CNN system, the 2 × 2 kernel of the max-pooling layer is used with a stride of 2 and applied to the spatial dimensions of the input. The outcomes of photographs after max-pooling are shown in Figure 5. Figure 5 illustrates the fifth outcome, showcasing the pictures generated by the ReLU layer after the max-pooling procedure. Each picture is a scaled-down version of the preceding image, with dimensions of 12x12 pixels, which is half the size of the previous image. This issue exemplifies how much the pooling layer may reduce the necessary resources.

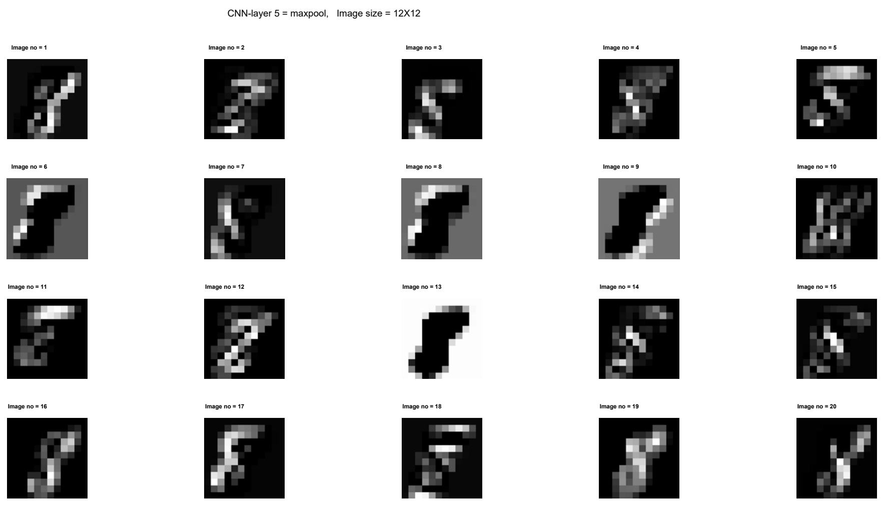


Fig. 5 *The results of images after max pooling for 5 x 5-filter size*

The convolutional and pooling layers play a pivotal role in generating multiple smaller images, mirroring the convolution filters employed in the process. This transformation effectively converts the original input image into a diverse array of compact feature maps, each highlighting distinct aspects. Figure 6 serves as a visual representation, showcasing the learned coefficients attributed to 20 trained convolution filters, all sized at 5x5. These filters, essentially pixelated images, exhibit grayscale representations where the varying shades denote component values, with brighter tones corresponding to higher values. Within the CNN architecture, these filters have proven notably effective in discerning and extracting pivotal features from the MNIST image dataset, underscoring their significance in the network's operations and feature identification.

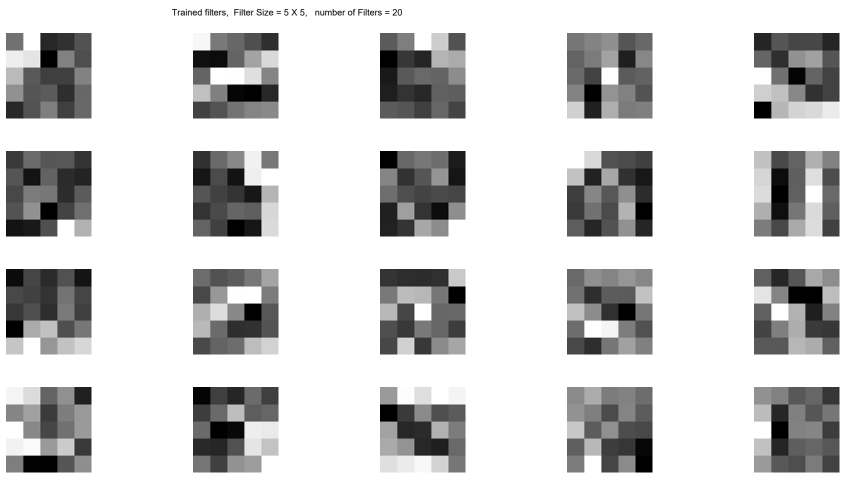

Trained filters, Filter Size = 5 X 5, number of Filters = 20

Fig. 6 *Trained 20 convolutional filters with size 5x5*

The image becomes a 24x24 feature map once it is processed with the 5x5-convolution filter. The layer produces 20 feature maps for each of the 20 convolution filters. Each feature map is reduced to 12x12 by the pooling layer using the 2x2 mean pooling process.

The fully connected layer collects feature mappings from previous convolutional layers and calculates the image's potential classification score in the input layer. In this layer, feature maps are converted into one-dimensional vectors. Based on training data, Fig. 7 represents the plot of 10 one-dimensional vectors after the image transformation.

Z = [-8.3014 -0.7979 -1.8114 -2.5943 0.4308 0.1967 4.0065 -4.1120 12.7905 2.4657]

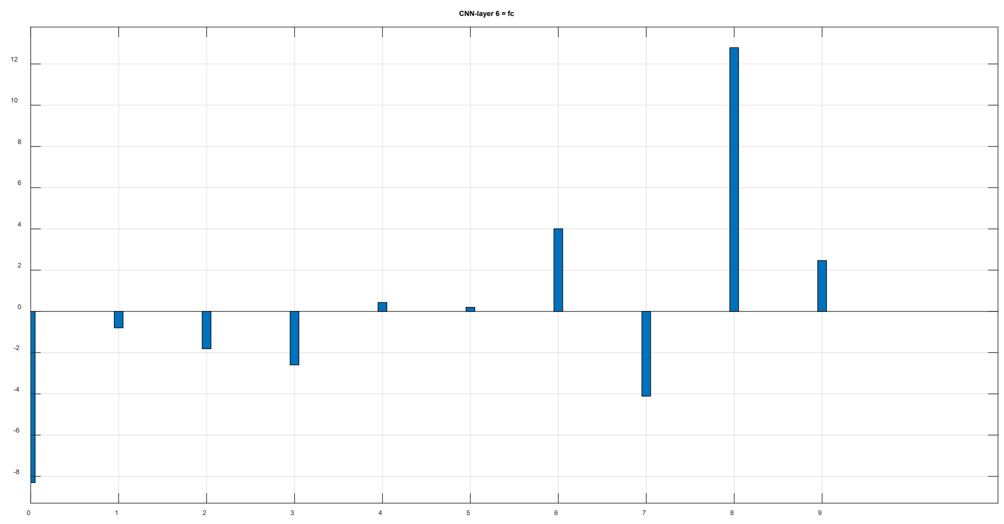
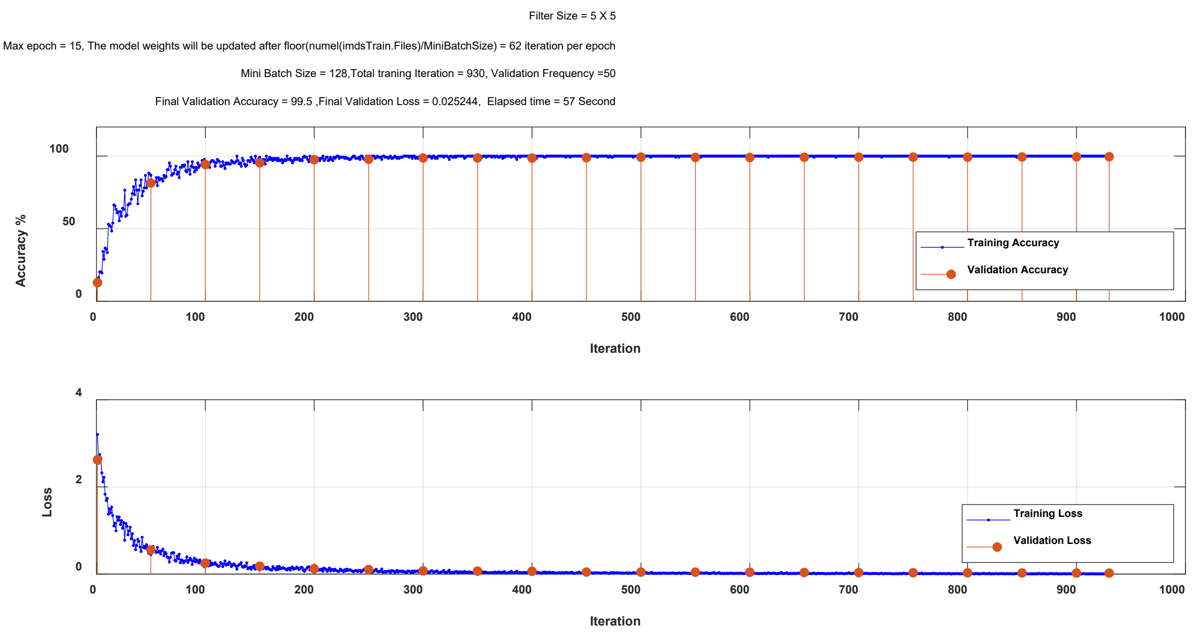


Fig. 7 *The plot of fully connected layer result of 10 one-dimensional vectors for 5 x 5-filter size*

The CNN architecture is trained using stochastic gradient descent with momentum (sgdm) algorithm, using default starting weights and bias settings. The initial learning rate is set at 0.01. A predetermined limit of epochs is applied to evaluate the training performance and efficiency. The epochs are limited to 1, 5, 10, 15, and 20. By increasing the number of epochs, more reliable outcomes may be anticipated. The training is conducted on a solitary CPU.

Once the training of the network starts, a training progress plot pops up, as shown in Fig. 8. The parameter of Fig. 8 is 5x5-filter size, 15 epochs, 62 iterations per epoch, and 128 mini-batch size.


Filter Size = 5 X 5

Max epoch = 15, The model weights will be updated after floor(numel(imdsTrain.Files)/MiniBatchSize) = 62 iteration per epoch

Mini Batch Size = 128,Total traning Iteration = 930, Validation Frequency =50

Final Validation Accuracy = 99.5 ,Final Validation Loss = 0.025244, Elapsed time = 57 Second

Fig. 8 *Training progress accuracy and loss. Filter size 5x5, MaxEpoch = 15, mini batch size = 128, iteration per epoch = 62, validation frequency = 50. Final validation accuracy (VA) = 99.5 and elapsed time of training = 59 second*

The loss and accuracy results achieved by training and validation of the proposed CNN design are shown in Table 3 for the different filter sizes at each maximum epoch. Table 3 demonstrates that CNN performs best with 99.5 % validation accuracy with the filter size 5 x 5 when MaxEpoch is 15. The degradation in accuracy occurred for other filters of other sizes because the effect of neighboring pixels depends on the size of the window. For instance, using a 5x5 filter size becomes more suitable for the shape of the numbers, and almost all the points fall within this range.

Table 3 *The loss, accuracy, and time elapse results achieved of training the proposed CNN architecture*

Filter size	Epoch	Iitreation	TL	VL	TA %	VA %	T CPU sec
3	1	62	0.7400	0.6500	75	77.3000	6.5000
3	5	310	0.0600	0.0900	99.2200	97.9500	22.2500
3	10	620	0.0100	0.0500	100	98.7000	39.8300
3	15	930	0.0100	0.0400	100	98.8500	63.5500
3	20	1240	0.0100	0.0400	100	99.1000	73.6600
5	1	62	0.4500	0.5000	85.1600	83.4500	7.2300
5	5	310	0.0500	0.0800	100	98.2000	19.6900
5	10	620	0.0200	0.0400	100	99.1000	39.3600
5	15	930	0.0100	0.0300	100	99.5000	59.0800
5	20	1240	0.0100	0.0300	100	98.9000	77.6600
7	1	62	0.5200	0.6600	82.8100	78	8.6300
7	5	310	0.0500	0.0800	100	98.2000	23.0200
7	10	620	0.0200	0.0500	100	99.1000	36.3600
7	15	930	0.0100	0.0300	100	99.1000	65.4400
7	20	1240	0.0100	0.0400	100	98.9500	69.0900
9	1	62	0.6000	0.5900	79.6900	80.9500	11.2500
9	5	310	0.1200	0.1000	98.4400	97.9000	27.5600
9	10	620	0.0200	0.0400	100	99.1500	49.5000
9	15	930	0.0100	0.0400	100	99.3000	72.3800
9	20	1240	0.0100	0.0300	100	99.3500	93.5900

The distribution of the final validation accuracy performance of the proposed CNN architecture is illustrated in Fig.9 along five maximum epochs with four different filter sizes. The graph depicts the impact of five maximum number of epochs with four different filter sizes model accuracy.

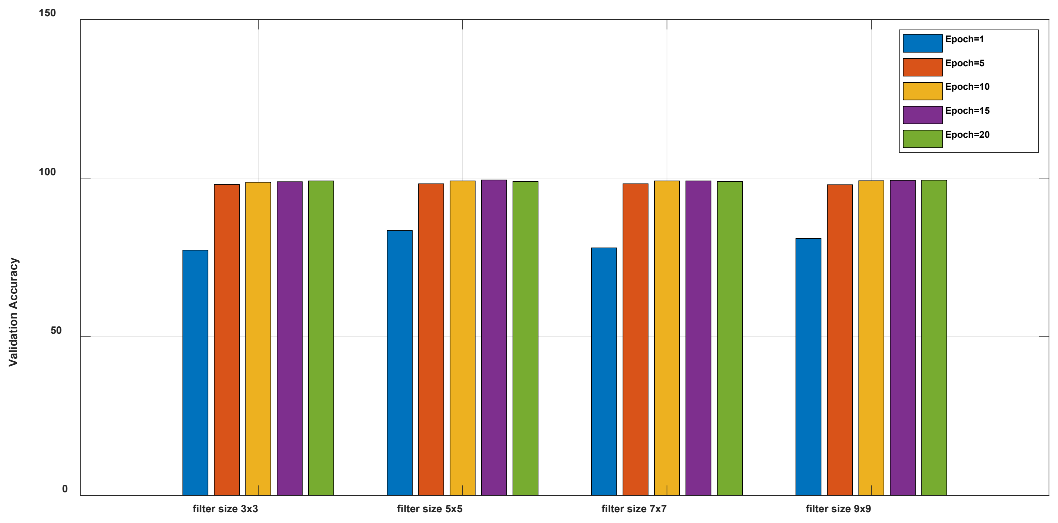


Fig. 9 *Illustrating the impact of filter size and maximum epoch on validation accuracy performance using digits' dataset*

Figure 10 depicts the confusion matrix for the proposed neural network. The suggested CNN architecture's classification performance is evaluated using a confusion matrix. A confusion matrix (sometimes called an error matrix) is a table that summarizes the outcomes of a classification model. The number of correct and wrong estimations is totaled and burnt away by class. The confusion matrix is made up of four categories: true negative (TN), false negative (FN), true positive (TP), and false positive (FP). The confusion matrix (CM) aids in the proper determination of other outcomes. CM contains two primary points. One is the true label, which saves all stored true values. This marker is located on the x-axis. Another is the predicted label, which gives results from trained algorithms for comparison with the original real value. This label is located on the y-axis. Among all of the values, the true positive part produces the proper result. The true positive component can be determined on a graph where values with the same label on both axes (x and y) match with the help of CM.

From Fig.10, We can see that the main diagonal comprises more than 99% of the values (diagonal components) representing the number of points where the true label equals the predicted label. On the other hand, only about 1% of the values (off-diagonal components) representing mislabeled elements are located outside of the main diagonal.

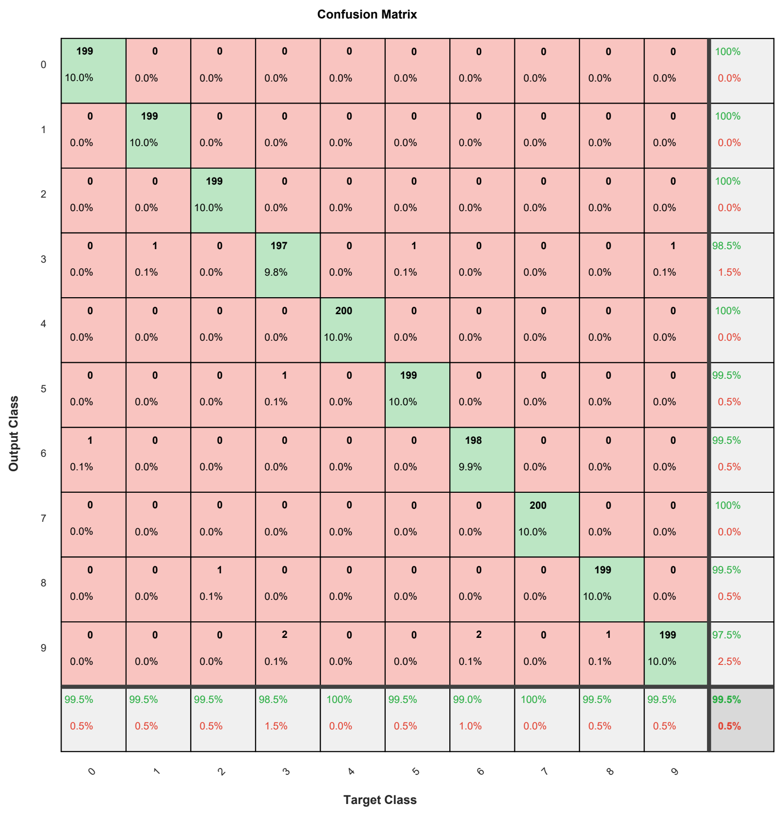


Fig. 10 *Confusion matrix graph for filter size 5 x 5 the number of epochs is 15*

5. Discussion

This segment examines the performance of CNN models in handwritten digit recognition tasks, comparing them to previous research and emphasizing the accuracy reached in a current study. The extract mentions Table 3, which most likely summarises past research on handwritten digit recognition with CNN models. Accuracy levels reported in these trials range from 98% to 99.3%. However, it warns that many studies may lack explicit evaluation procedures, meaning that the methodologies employed to measure model performance were not rigorous or transparent enough. The CNN model produced in the current project has significantly improved accuracy. It is close to the greatest recorded accuracy of 99.5%. This means that the model created for this study works extraordinarily well at recognizing handwritten digits.

Table 3 *Comparison of CNN results with existing studies*

Research Paper	Accuracy Rate (%)
Hybrid CNN-SVM Classifier for Handwritten Digit Recognition (CNNSVM) [24]	99.2
Handwritten Digit Recognition System Using CNN [25]	98.0
Implementation of Handwritten Digit Recognizer using CNN [26]	99.1
Handwritten Digit Recognition using Machine and Deep Learning Algorithms (CNN) [27]	99.3
Hand Written Digit Recognition Using Machine Learning (CNN) [28]	95.0
Hand Written Digit Recognition Using Machine Learning (CNN) [29]	98.4
The proposed method	99.5

Importantly, the excerpt emphasizes that the current project's CNN model was assessed using rigorous cross-validation methods to minimize overfitting. Cross-validation is a technique for determining how well a model will generalize to an independent dataset. Using this technique, the project assures that the reported accuracy is reliable and not exaggerated by overfitting. The comparison indicates that the CNN model created in the current project beats other existing models for handwritten digit recognition. Regardless of the minimal difference in accuracy, the rigorous review procedure increases confidence in the existing model's performance. This suggests that the project's approach could be a substantial development in the field of handwritten digit recognition.

Generally, this work offers comprehension of the performance of CNN models in handwritten digit recognition tasks, highlighting the importance of rigorous evaluation methods and emphasizing the superior performance of the CNN model established in the current project.

6. Conclusion

CNN uses deep learning, is one of the best approaches to the significant problem of handwritten digit classification in the rapidly developing field of technology. This paper completes an effective architecture for the MNIST dataset classification. This study explains the process of developing handwriting digit recognition and classification of the handwriting digit dataset. The deep learning CNN architecture is proposed to improve the training and validation results. After comparing the different combinations of filter sizes, the proposed architecture performance is evaluated for overall accuracy, which reaches 99.5% using a 5x5-filter size MaxEpoch of 15. Moreover, it was found that after epoch 5 it does not contribute much to the accuracy of the classifier. For future work, we believe it is worthwhile to take additional steps to improve our model's performance in learning and extracting local features in the hidden layers precisely and how to increase recognition capacity in the fully linked layers to avoid mislabeling issues. Furthermore, the suggested approach can be applied to different datasets.

Acknowledgement

The author would like to acknowledge the Nawroz University, Computer Science, and the University of Duhok, Electrical and Computer Engineering, Duhok, Iraq.

Conflict of Interest

Authors declare that there is no conflict of interests regarding the publication of the paper.

Author Contribution

All authors have contributed significantly to the research and writing of this paper, each bringing their unique expertise and insights to ensure the quality and comprehensiveness of the final manuscript.

References

[1] Ahlawat, S., & Choudhary, A. (2020). Hybrid CNN-SVM Classifier for Handwritten Digit Recognition. Procedia Computer Science, 167, 2554–2560. https://doi.org/10.1016/j.procs.2020.03.309

[2] Zebari, D. A., Haron, H., Sulaiman, D. M., Yusoff, Y., & Othman, M. N. M. (2022, December). CNN-based Deep Transfer Learning Approach for Detecting Breast Cancer in Mammogram Images. In 2022 IEEE 10th Conference on Systems, Process & Control (ICSPC) (pp. 256-261). IEEE

[3] Abdullah, D. M., Abdulazeez, A. M., & Sallow, A. B. (2021). Lung cancer prediction and classification based on correlation selection method using machine learning techniques. *Qubahan Academic Journal*, *1*(2), 141-149.

[4] Mohammed, S. H., & Çinar, A. (2021). Lung cancer classification with convolutional neural network architectures. *Qubahan Academic Journal*, *1*(1), 33-39.

[5] Barnouti, N. H., Abomaali, M., & Al-Mayyahi, M. H. N. (2018). An efficient character recognition technique using K-nearest neighbor classifier. International Journal of Engineering & Technology, 7(4), 3148-3153.

[6] Khanday, O. M., & Dadvandipour, S. (2021). Analysis of machine learning algorithms for character recognition: a case study on handwritten digit recognition. Indonesian Journal of Electrical Engineering and Computer Science, 21(1), 574-581.

[7] Abdullah, D. M., & Abdulazeez, A. M. (2021). Machine learning applications based on SVM classification a review. *Qubahan Academic Journal*, *1*(2), 81-90.

[8] Gogulamudi, S., Pinnela, V. K., Pathuri, L. S. T., & Borra, R. (2020). Handwritten Digit Recognition by using Pattern Recognition & Consensus Clustering. International Journal of Innovative Technology and Exploring Engineering (IJITEE), 9(6), 2263-2267.

[9] Li Deng. (2012). The MNIST Database of Handwritten Digit Images for Machine Learning Research. IEEE Signal Processing Magazine, 29(6), 141–142. https://doi.org/10.1109/MSP.2012.2211477

[10] Michael Nielsen. (2015). Neural networks and deep learning (Vol. 25). Determination press.
[11] Haijian Shao, E. M. M. Z. X. D. and S. Z. (2023). MNIST Handwritten Digit Classification Based on Convolutional Neural Network with Hyperparameter Optimization. Intelligent Automation & Soft Computing, IASC, 36(3), 3596–3606.
[12] Patrice Y. Simard, D. S. J. C. P. (2003). Best Practices for Convolutional Neural Networks Applied to Visual Document Analysis. Proceedings of the Seventh International Conference on Document Analysis and Recognition (ICDAR 2003).
[13] S. Ali, Z. S. M. A. Z. S. and T. M. (2019). An efficient and improved scheme for handwritten digit recognition based on convolutional neural network. SN Applied Sciences, 1(9), 1–9.
[14] Savita Ahlawat, A. C. A. N. S. S. and B. Y. (2020). Improved Handwritten Digit Recognition Using Convolutional Neural Networks (CNN). Sensors, 20(12), 3344.
[15] Ahmed, S. S., Mehmood, Z., Awan, I. A., & Yousaf, R. M. (2023). A novel technique for handwritten digit recognition using deep learning. Journal of Sensors, 2023.
[16] Kim, Y., Ohn, I., & Kim, D. (2021). Fast convergence rates of deep neural networks for classification. Neural Networks, 138, 179-197.
[17] LeCun, Y. (1998). The MNIST database of handwritten digits. *http://yann. lecun. com/exdb/mnist/*.
[18] Qian, L., Bai, J., Huang, Y., et al. (2024). Breast cancer diagnosis using evolving deep convolutional neural network based on hybrid extreme learning machine technique and improved chimp optimization algorithm. *Biomedical Signal Processing and Control*, *87*, 105492.
[19] Salih, B. M., Abdulazeez, A. M., & Hassan, O. M. S. (2021). Gender classification based on iris recognition using artificial neural networks. *Qubahan Academic Journal*, *1*(2), 156-163.
[20] Zebari, N. A., Mohammed, C. N., et al.. (2024). A deep learning fusion model for accurate classification of brain tumours in Magnetic Resonance images. *CAAI Transactions on Intelligence Technology*.
[21] Sulaiman, D. M., Abdulazeez, A. M., et al. (2022). An attention-based deep regional learning model for enhanced finger vein identification. *Traitement du Signal*, *39*(6), 1991.
[22] Ibrahim, D. A., Zebari, D. A., Mohammed, H. J., & Mohammed, M. A. (2022). Effective hybrid deep learning model for COVID-19 patterns identification using CT images. *Expert Systems*, *39*(10), e13010.
[23] Zebari, D. A., Sadiq, S. S., & Sulaiman, D. M. (2022, March). Knee Osteoarthritis Detection Using Deep Feature Based on Convolutional Neural Network. In 2022 International Conference on Computer Science and Software Engineering (CSASE) (pp. 259-264). IEEE.
[24] Ahlawat, S., & Choudhary, A. (2020). Hybrid CNN-SVM classifier for handwritten digit recognition. *Procedia Computer Science*, *167*, 2554-2560.
[25] GOH, K. Y., & Ab Ghafar, A. S. B. (2021). Handwritten Digit Recognition System using Convolutional Neural Network (CNN). *Progress in Engineering Application and Technology*, *2*(1), 578-592.
[26] 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*.
[27] Scientific, L. L. (2024). HANDWRITTEN DIGIT RECOGNITION USING MACHINE LEARNING. *Journal of Theoretical and Applied Information Technology*, *102*(5).
[28] Praneeth, A. V. S. R., Pappu, L. R., Kumar, G. P., Bhatta, A. S., Chowdary, B. R., & Shishwan, G. S. (2022). Hand Written Digit Recognition Using Machine Learning. *International Journal of Research in Engineering, Science and Management*, *5*(1), 59-62.
[29] Sharma, M., Sindal, P. S., & Baskar, M. (2023). Handwritten digit recognition using machine learning. In *Proceedings of Data Analytics and Management: ICDAM 2022* (pp. 31-43). Singapore: Springer Nature Singapore.