

Classifying Handwritten Digits with Neural Network

Nikhil Subramanian ¹, Harensylak M.K ², Santanu Guha ³, Swapnil Mishra ⁴ Naveen Kumar Vaegae ⁶

¹ nikhil.subramanian2018@vitstudent.ac.in, ² harensylakm.k2018@vitstudent.ac.in, ³

santanu.guha2018@vitstudent.ac.in, swapnil.mishra2018@vitstudent.ac.in, ⁶ veenaveen@vit.ac.in

School of Electronics Engineering, Vellore Institute of Technology, Vellore 632014, India

Abstract:

In recent times, with the increase of Artificial Neural Network (ANN), deep learning has brought a dramatic twist in the field of machine learning by making it more artificially intelligent. Deep learning is remarkably used in vast ranges of fields because of its diverse range of applications such as surveillance, health, medicine, sports, robotics, drones, etc. In deep learning, Convolutional Neural Network (CNN) is at the centre of spectacular advances that mixes Artificial Neural Network (ANN) and up to date deep learning strategies. It has been used broadly in pattern recognition, sentence classification, speech recognition, face recognition, text categorization, document analysis, scene, and handwritten digit recognition. The goal of this paper is to observe the variation of accuracies of CNN to classify handwritten digits using various numbers of hidden layers and epochs and to make the comparison between the accuracies. For this performance evaluation of CNN, we performed our experiment using Modified National Institute of Standards and Technology (MNIST) dataset.

Keywords: *Handwritten digit recognition, Convolutional Neural Network (CNN), Deep learning, MNIST dataset, Epochs, Hidden Layers, Stochastic Gradient Descent, Backpropagation*

1. Introduction:

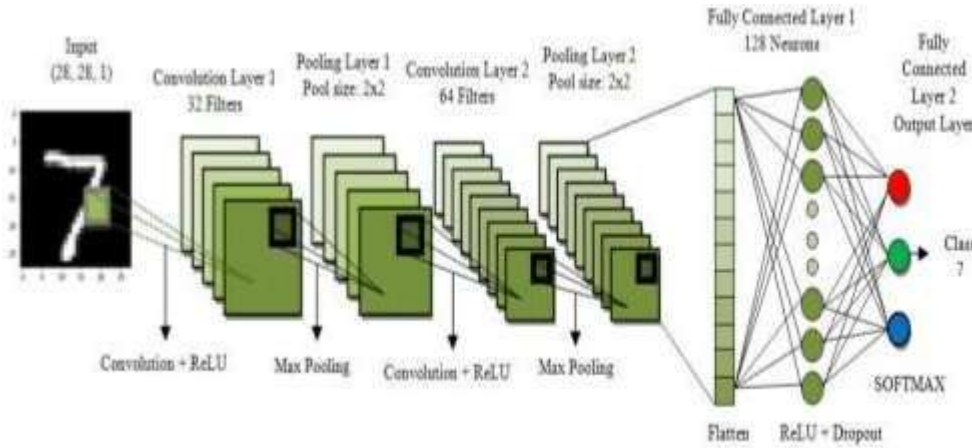
With time the numbers of fields are increasing in which deep learning can be applied. In deep learning, Convolutional Neural Network (CNN) is being used for visual imagery analysing. Object detection, face recognition, robotics, video analysis, segmentation, pattern recognition, natural language processing, spam detection, topic categorization, regression analysis, speech recognition, image classification is some of the examples that can be done using Convolutional Neural Network. The accuracies in these fields including handwritten digits recognition using Deep Convolutional Neural Networks (CNNs) have reached human level perfection. Mammalian visual systems' biological model is the one by which the architecture of the CNN is inspired. Cells in the cat's visual cortex are sensitized to a tiny area of the visual field identified which is recognized as the receptive field. It was found by D. H. Hubel et al. in 1962. The neocognitron, the pattern recognition model inspired by the work of

D. H. Hubel et al. was the first computer vision. It was introduced by Fukushima in 1980. In 1998, the framework of CNNs is designed by LeCun et al. which had seven layers of convolutional neural networks. It was adept in handwritten digits classification direct from pixel values of images. Gradient descent and back propagation algorithm is used for training the model. In handwritten recognition digits, characters are given as input. The model can be recognized by the system. A simple artificial neural network (ANN) has an input layer, an output layer and some hidden layers between the input and output layer. CNN has a very similar architecture as ANN. There are several neurons in each layer in ANN. The weighted sum of all the neurons of a layer becomes the input of a neuron of the next layer adding a biased value. In CNN the layer has three dimensions. Here all the neurons are not fully connected. Instead, every neuron in the layer is connected to the local receptive field. A cost function generates in order to train the network. It compares the output of the network with the desired output. The signal propagates back to the system, again and again, to update the shared weights and biases in all the receptive fields to minimize the value of cost function which increases the network's performance. The goal of this article is to observe the influence of hidden layers of a CNN for handwritten digits. We have applied a different type of

Convolutional Neural Network algorithm on Modified National Institute of Standards and Technology (MNIST) dataset using TensorFlow, a Neural Network library written in python. The main purpose of this paper is to analyse the variation of outcome results for using a different combination of hidden layers of Convolutional Neural Network. Stochastic gradient and backpropagation algorithm are used for training the network and the forward algorithm is used for testing.

2 Proposed Method

To recognize the handwritten digits, a seven-layered convolutional neural network with one input layer followed by five hidden layers and one output layer is designed and illustrated in figure 1.



The input layer consists of 28 by 28-pixel images which mean that the network contains 784 neurons as input data. The input pixels are grayscale with a value 0 for a white pixel and 1 for a black pixel. Here, this model of CNN has five hidden layers. The first hidden layer is the convolution layer 1 which is responsible for feature extraction from an input data. This layer performs convolution operation to small localized areas by convolving a filter with the previous layer. In addition, it consists of multiple feature maps with learnable kernels and rectified linear units (ReLU). The kernel size determines the locality of the filters. ReLU is used as an activation function at the end of each convolution layer as well as a fully connected layer to enhance the performance of the model. The next hidden layer is the pooling layer 1. It reduces the output information from the convolution layer and reduces the number of parameters and computational complexity of the model. The different types of pooling are max pooling, min pooling, average pooling, and L2 pooling. Here, max pooling is used to subsample the dimension of each feature map. Convolution layer 2 and pooling layer 2 which has the same function as convolution layer 1 and pooling layer 1 and operates in the same way except for their feature maps and kernel size varies. A Flatten layer is used after the pooling layer which converts the 2D featured map matrix to a 1D feature vector and allows the output to get handled by the fully connected layers. A fully connected layer is another hidden layer also known as the dense layer. It is similar to the hidden layer of Artificial Neural Networks (ANNs) but here it is fully connected and connects every neuron from the previous layer to the next layer. In order to reduce overfitting, dropout regularization method is used at fully connected layer 1. It randomly switches off some neurons during training to improve the performance of the network by making it more robust. This causes the network to become capable of better generalization and less compelling to overfit the training data. The output layer of the network consists of ten neurons and determines the digits numbered from 0 to 9. Since the output layer uses an activation function such as SoftMax, which is used to enhance the performance of the model, classifies the output digit from 0 through 9 which has the highest activation value. The MNIST handwritten digits database is used for the experiment. Out of 70,000 scanned images of handwritten digits from the MNIST database, 60,000 scanned images of digits are used for training the network and 10,000 scanned images of digits are used to test the network. The images that

are used for training and testing the network all are the grayscale image with a size of 28×28 pixels. Character x is used to represent a training input where x is a 784-dimensional vector as the input of x is regarded as 28×28 pixels. The equivalent desired output is expressed by y(x), where y is a 10-dimensional vector. The network aims to find the convenient weights and biases so that the output of the network approximates y(x) for all training inputs x as it completely depends on weight values and bias values. To compute the network performances, a cost function is defined, expressed by equation 1.

$$C(w, b) = \frac{1}{2n} \sum_x [y(x) - a^2]^2 \quad (1)$$

Where w is the cumulation of weights in the network, b is all the biases, n is the total number of training inputs and a is the actual output. The actual output a depends on x, w, and b. C (w,

b) is non-negative as all the terms in the sum is non-negative. Moreover, C (w, b) =0, precisely when desired output y(x) is comparatively equal to the actual output, a, for all training inputs,

n. To reduce the cost C (w, b) to a smaller degree as a function of weight and biases, the training algorithm has to find a set of weight and biases which cause the cost to become as small as possible. This is done using an algorithm known as gradient descent. In other words, gradient descent is an optimization algorithm that twists its parameters iteratively to minimize a cost function to its local minimum. The gradient descent algorithm deploys the following equations to set the weight and biases.

$$w^{new} = w^{old} - \eta \frac{\partial C}{\partial w^{old}} \quad (2)$$

$$b^{new} = b^{old} - \eta \frac{\partial C}{\partial b^{old}} \quad (3)$$

$$w^{new} = w^{old} - \eta \frac{\partial C}{\partial w^{old}} \quad (2)$$

$$b^{new} = b^{old} - \eta \frac{\partial C}{\partial b^{old}} \quad (3)$$

$$\delta^L = \frac{\partial C}{\partial a^{(L)}} \frac{\partial a^{(L)}}{\partial z^{(L)}} = \frac{1}{n} (a^{(L)} - 1) f'(z^{(L)}) \quad (7)$$

$$\delta^l = \frac{\partial C}{\partial z^{(l)}} = \frac{\partial C}{\partial z^{(l)}} \frac{\partial z^{(l+1)}}{\partial z^{(l)}} = \frac{\partial z^{(l+1)}}{\partial z^{(l)}} \delta^{l+1} = w^{l+1} \delta^{l+1} f'(z^l) \quad (8)$$

$$\frac{\partial C}{\partial b^{(l)}} = \delta^l \quad (9)$$

$$\frac{\partial C}{\partial w^{(L)}} = a^{l-1} \delta^l \quad (10)$$

3. Database and performance metrics

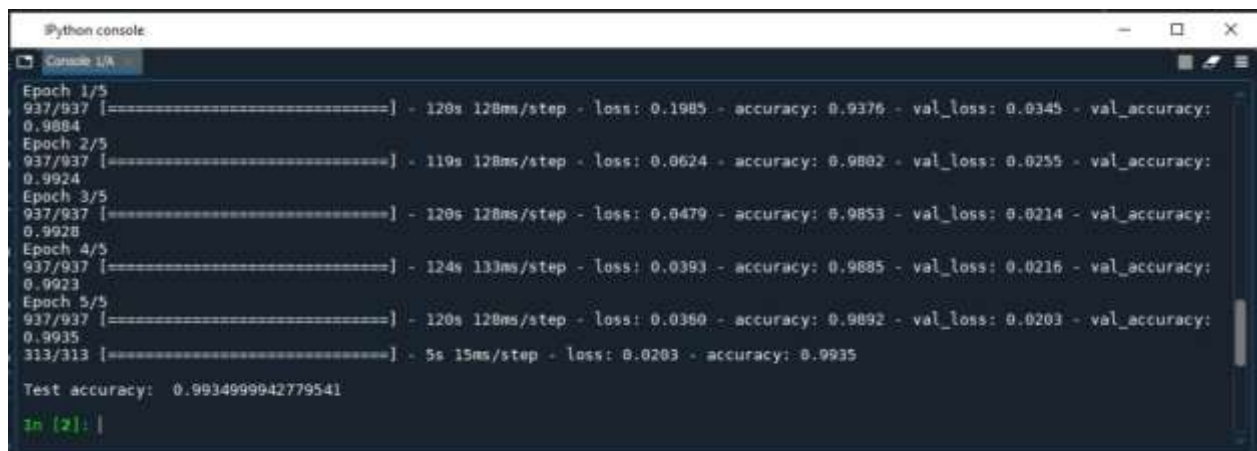
Modified National Institute of Standards and Technology (MNIST) is a large set of computer vision dataset which is extensively used for training and testing different systems. It was created from the two special datasets of National Institute of Standards and Technology (NIST) which holds binary images of handwritten digits. The training set contains handwritten digits from 250 people, among them 50% training dataset was employees from the Census Bureau and the rest of it was from high school students. However, it is often attributed as the first datasets among other datasets to prove the effectiveness of the neural networks. The database contains 60,000 images used for training as well as few of them can be used for cross-validation purposes and 10,000 images used for testing. All the digits are grayscale and positioned in a fixed size where the intensity lies at the centre of the image with 28×28 pixels. Since all the images are 28×28 pixels, it forms an array which can be flattened into $28 \times 28 = 784$ -dimensional vector. Each component of the vector is a binary value which describes the intensity of the pixel.

4. Results

In this section, CNN has been applied on the MNIST dataset in order to observe the variation of accuracies for handwritten digits. The accuracies are obtained using TensorFlow in python.

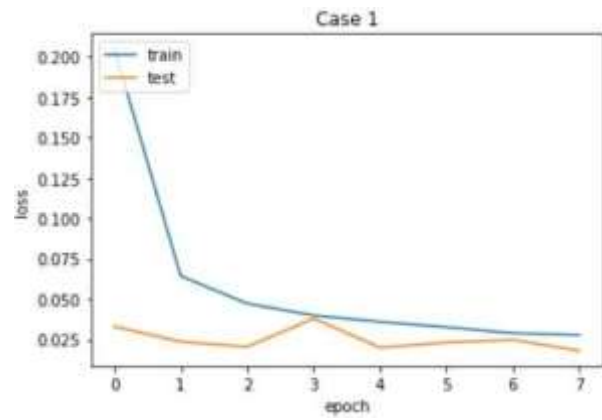
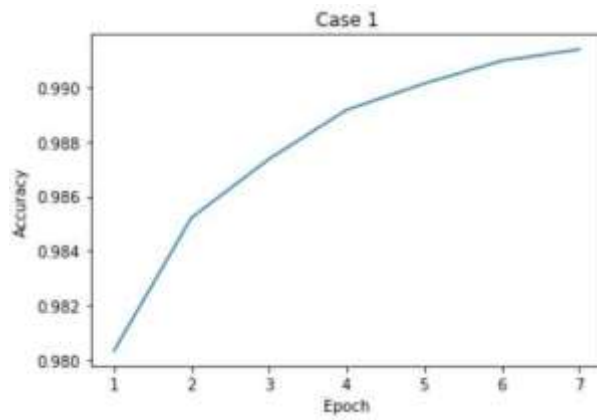
Table 1 illustrates the various accuracy and loss for the different epoch and NN layers in the model. For case 2, 4 and 6 an additional dense layer is added to the network with 256 Neurons.

Case	Number of Neural Network Layers	Steps per epoch(Batch Size)	Epochs	Accuracy	Loss
1	2	937	8	99.4	0.0208
2	3	937	8	99.24	0.0277
3	2	937	10	99.38	0.0195
4	3	937	10	99.51	0.0196
5	2	937	12	99.42	0.0248
6	3	937	12	99.32	0.0307

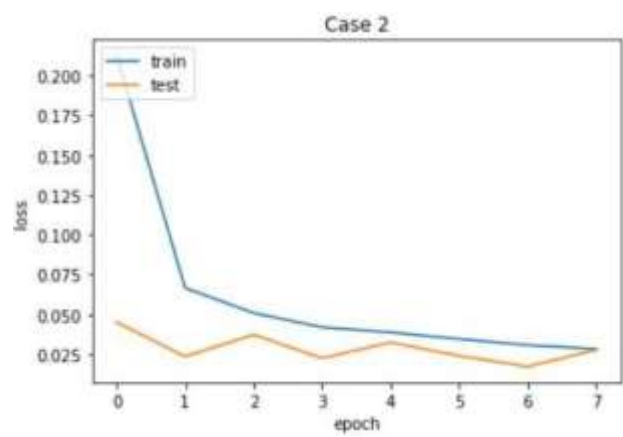
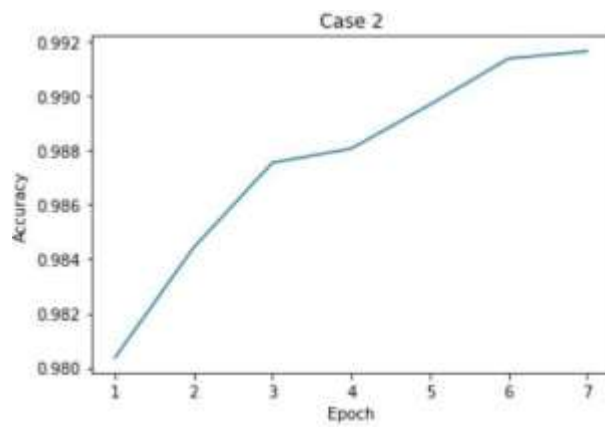


```
Python console
Console 1/A
Epoch 1/5
937/937 [=====] - 120s 128ms/step - loss: 0.1985 - accuracy: 0.9376 - val_loss: 0.0345 - val_accuracy: 0.9884
Epoch 2/5
937/937 [=====] - 119s 128ms/step - loss: 0.0624 - accuracy: 0.9802 - val_loss: 0.0255 - val_accuracy: 0.9924
Epoch 3/5
937/937 [=====] - 120s 128ms/step - loss: 0.0479 - accuracy: 0.9853 - val_loss: 0.0214 - val_accuracy: 0.9928
Epoch 4/5
937/937 [=====] - 124s 133ms/step - loss: 0.0393 - accuracy: 0.9885 - val_loss: 0.0216 - val_accuracy: 0.9923
Epoch 5/5
937/937 [=====] - 120s 128ms/step - loss: 0.0360 - accuracy: 0.9892 - val_loss: 0.0203 - val_accuracy: 0.9935
313/313 [=====] - 5s 15ms/step - loss: 0.0203 - accuracy: 0.9935
Test accuracy: 0.9934999942779541
In [2]: |
```

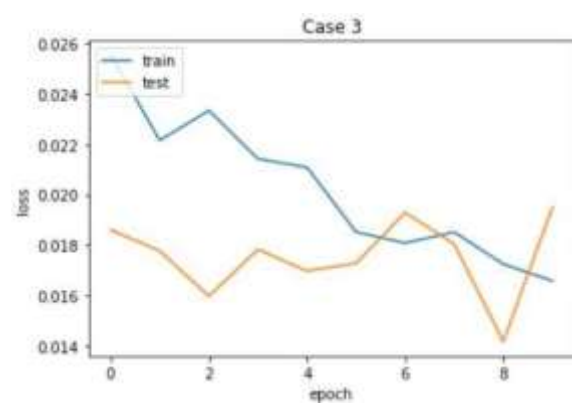
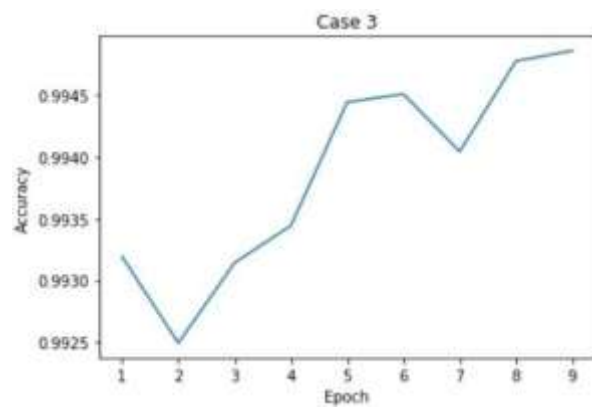
Case 1:



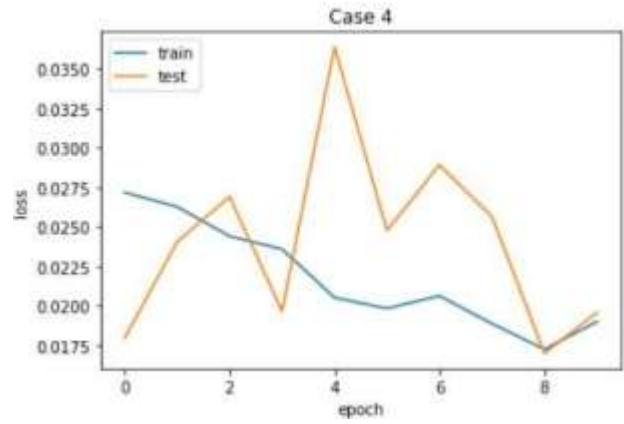
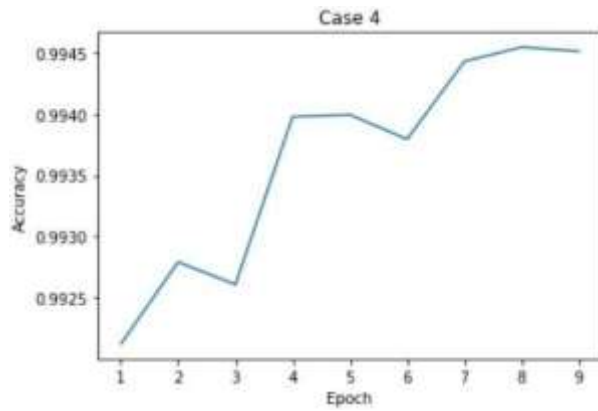
Case 2:



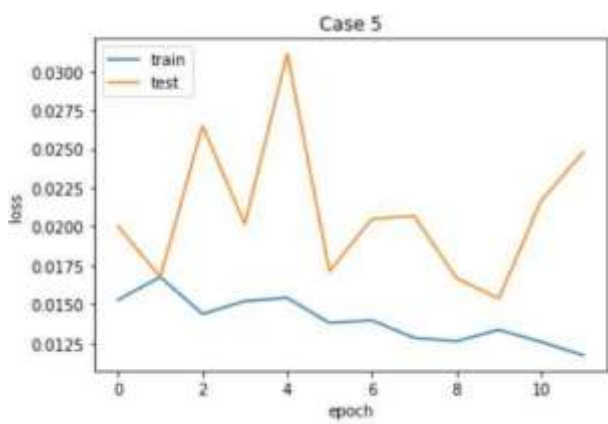
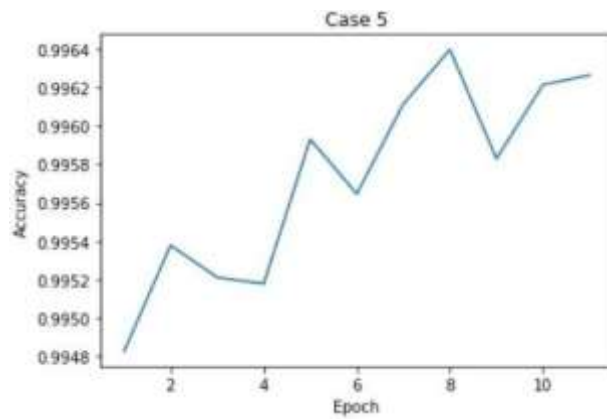
Case 3:



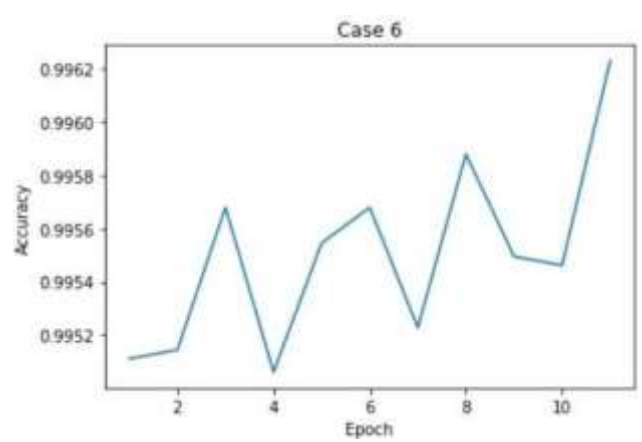
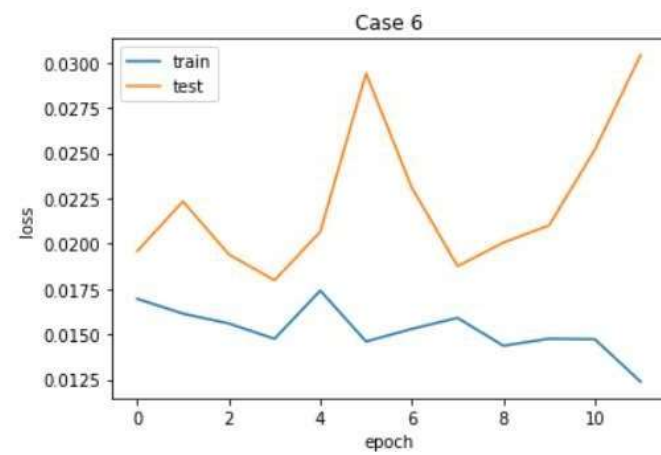
Case 4:



Case 5:



Case 6:



4. Discussions

There are several methods of digit recognition. The handwritten digit recognition can be improved using some widely held methods of the neural network like Deep Neural Network (DNN), Deep Belief Network (DBF) and Convolutional Neural Network (CNN), etc. Tavanaei et al. proposed multi-layered unsupervised learning in a spiking CNN model where they used MNIST dataset to clear the blur images and found the overall accuracy of 98% and the range of performance loss was 0.1% to 8.5%. Rezoana et al. proposed a seven-layered Convolutional Neural Network for the purpose of handwritten digit recognition where they used MNIST dataset to evaluate the impact of the pattern of the hidden layers of CNN over the performance of the overall network. They have plotted the loss curves against the number of epochs and found that the performance loss was below 0.1 for most of the cases and sometimes, in some cases, the loss was less than 0.05. In another paper, Siddique et al. proposed an L-layered feed-forward neural network for the handwritten digit recognition where they have applied neural network with different layers on the MNIST dataset to observe the variation of accuracies of ANN for different combinations of hidden layers and epochs. Their maximum accuracy in the performance was found 97.32% for 4 hidden layers at 50 epochs. Comparing with their above performances based on MNIST dataset for the purpose of digit recognition we have achieved better performance for the CNN. In our experiment, we have found the maximum training accuracy 100% and maximum validation accuracy 99.51% both at epoch 10. The overall performance of the network is found 99.51%. Moreover, the overall loss ranged from 0.0307 to 0.01956. Hence, this proposed method of CNN is more efficient than the other existing method for digit recognition.

5. Conclusion

In this project, the variations of accuracies for handwritten digit were observed by varying the hidden layers and epochs. The accuracy curves were generated for the six cases for the different parameter using CNN MNIST digit dataset.

The six cases perform differently because of the various combinations of hidden layers. The layers were taken randomly in a periodic sequence so that each case behaves differently during the experiment. Among all the observation, the maximum accuracy in the performance was found 99.51% and minimum loss at 0.0196 for 10 epochs in case 4. In digit recognition, this type of higher accuracy and lower loss will cooperate to speed up the performance of the machine more adequately. However, the minimum accuracy among all observation in the performance 99.32% was found and maximum loss at 0.0307 in case 6. In the future, we plan to observe the variation in the overall classification accuracy by varying the number of hidden layers and batch size.

REFERENCES

- [1]Y. LeCun et al., "Backpropagation applied to handwritten zip code recognition," Neural computation, vol. 1, no. 4, pp. 541-551, 1989.
- [2]A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Image net classification with deep convolutional neural networks," in Advances in neural information processing systems, 2012, pp. 1097-1105.
- [3]D. Hubel and T. Wiesel, "Aberrant visual projections in the Siamese cat," The Journal of physiology, vol. 218, no. 1, pp. 33-62, 1971.
- [4]Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," nature, vol. 521, no. 7553, p. 436, 2015.
- [5]D. Cireşan, U. Meier, and J. Schmidhuber, "Multi-column deep neural networks for image classification," arXiv preprint arXiv:1202.2745, 2012.
- [6]K. Fukushima and S. Miyake, "Neocognitron: A self-organizing neural network model for a mechanism of visual pattern recognition," in Competition and cooperation in neural nets: Springer, 1982, pp. 267-285.
- [7]Y. LeCun et al., "Handwritten digit recognition with a back-propagation network," in Advances in neural information processing systems, 1990, pp. 396-404.
- [8]Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," Proceedings of the IEEE, vol. 86, no. 11, pp. 2278-2324, 1998

Neural Network and Fuzzy Control

4th Group - Project Summary

Classifying handwritten digits with CNN

Name: Nikhil Subramanian

RegNo: 18BEC0711

We use CNN to do handwritten digit recognition because it is best for image recognition. CNN has a similar architecture to ANN.

The goal is to observe the variation of accuracies of CNN to classify handwritten digits. This model is trained using MNIST dataset. Using Python and libraries like pandas, tensorflow a CNN model is created. Our model consists of 7 layers. One input layer, one output layer and 5 hidden layers. Using activation functions like ReLU, softmax and maxpooling, our model is tuned perfectly. Upon training the model over ~~two~~ five epochs, the accuracy finally is at the range of 99.25% to 99.51%.