Implementation of a CNN architecture to classify the MNIST handwritten dataset

Md Abdus Sobhan
Computer Science of Engineering
American International University
Bangladesh
Dhaka, Bangladesh
masobhan.07@gmail.com

Abstract_ In this report, I have shown the procedural things and results of implementations of a CNN architecture to classify the MNIST handwritten dataset. As the accuracy of this architecture is a concern so that the accuracy has been increased by modifying the model architecture. More than 98 percentages of the accuracy of this CNN architecture is the target to get. Additionally, the testing has been executed by using different kinds of optimizers such as Adam, SGD, and RSMprop. These optimizers show the accuracy level of this architecture and give us the plotted graph which can define the architecture. Finally, after showing all the simulations and their results, all the steps and results have been discussed.

Introduction: In terms of solving image processing problems, the convolutional neural network (CNN) is mostly used. On the other hand, the MNIST is a handwritten dataset and it contains 60 thousand training sets of examples and 10 thousand test sets. The best accuracy can be achieved of CNN by using the MNIST dataset. Convolutional Neural Networks are used for image classifications as well as to object detections. The convolutional layers can ensure the strength of CNN. Finally, some algorithms called optimizers such as adam, SGD, and mrsprop are used to change the attributes of neural networks such as learning rate and weight to reduce the losses.

Results: This section contains the results of the sequential model which consists of the total and the trainable parameters. This model has the one-dimensional convolutional layer, max-pooling layer, flatten layer, dense layers, and finally the output layer.

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv1d (Conv1D)              (None, 24, 32)            4512
_________________________________________________________________
max_pooling1d (MaxPooling1D) (None, 12, 32)            0
_________________________________________________________________
conv1d_1 (Conv1D)            (None, 10, 64)            6208
_________________________________________________________________
max_pooling1d_1 (MaxPooling1 (None, 5, 64)             0
_________________________________________________________________
flatten (Flatten)            (None, 320)               0
_________________________________________________________________
dense (Dense)                (None, 128)               41088
_________________________________________________________________
dense_1 (Dense)              (None, 10)                1290
=================================================================
Total params: 53,098
Trainable params: 53,098
Non-trainable params: 0
_________________________________________________________________
```

Fig 1: Sequential model

The conventional neural network that has been implemented, has some figures based on different optimizers and other parameters such as train accuracy, validation accuracy, and test accuracy. All are given below_

Fig 2 below shows us the model has the estimated train accuracy is 99 percent, the validation accuracy is almost 98 percent and finally, the test accuracy is 98.7 percent. The optimizer which is used here is called RMSprop.

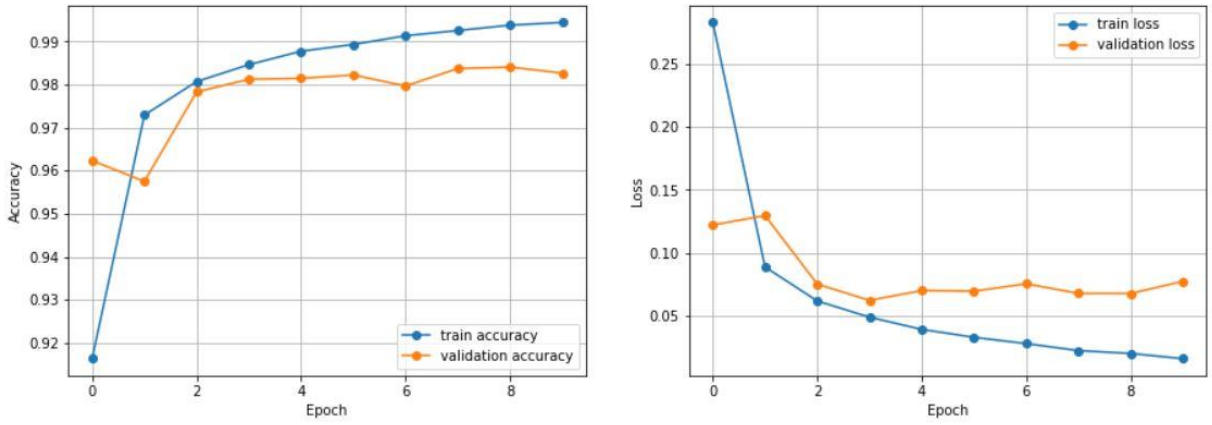

Fig 2: loss for rmsprop

Fig 3 below shows us the model has the estimated train accuracy is 92 percent, the validation accuracy is almost 92 percent and finally, the test accuracy is 92.6 percent. The optimizer which is used here is called SGD.

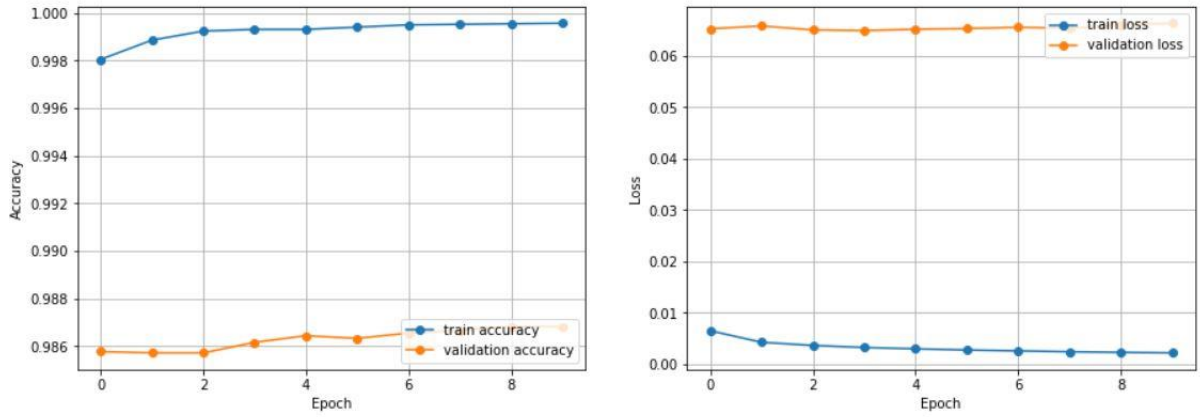

Fig 3: loss for SGD

Fig 4 below shows us the model has the estimated train accuracy is 98 percent, the validation accuracy is almost 98 percent and finally, the test accuracy is 98.3 percent. The optimizer which is used here is called Adam.

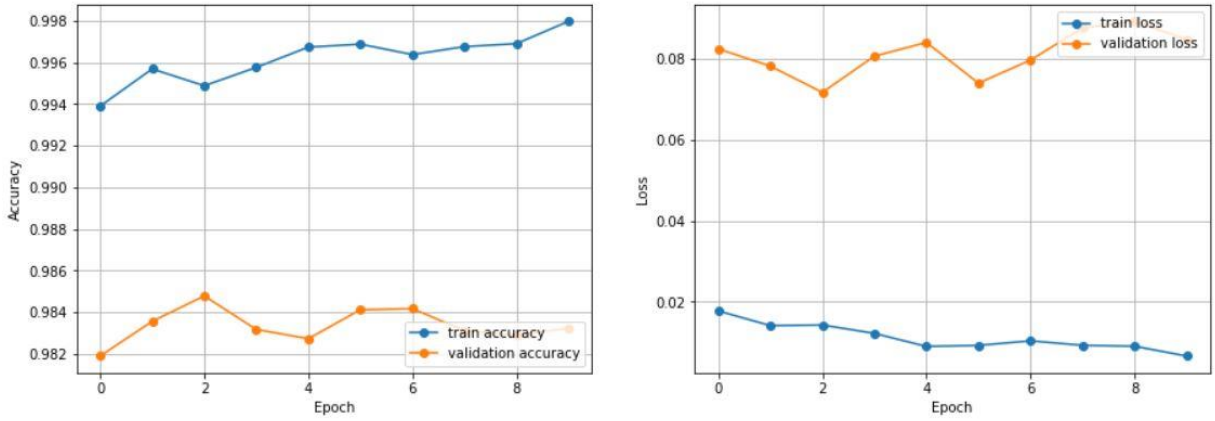

Fig 4: loss for Adam

Discussion: The above model has been implemented for classifying the handwritten MNIST dataset. After implementing, the accuracy of the dataset as well as the loss of dataset I have got in a various number of percentages. The accuracy can be increased by applying such types of procedures. Applying three different optimizers I got the highest percentages of accuracy in RMSprop optimizers which is 98.7% and the lowest test accuracy has been given by the SGD optimizer which is 93.6%. Besides, the adam optimizer gave us the 98.3% of test accuracy. A little

bit of different value of validation accuracy and train accuracy has been noticed. So the optimizer which is used firstly states that the accuracy can be increased step by step which is may increase the authenticity of the architecture model.

----------@----------