Name: Arfanul Kabir Apurbo

ID: 18-37135-1

Section: [B]

Course: Computer Vision and Pattern Recognition

Title: CNN model to classify the MNIST handwritten dataset

Abstract:

Because handwritten information differs from person to person, automatic recognition of handwritten digits from visual data can be challenging. My major goal for this project is to propose a simple convolutional neural network (CNN) model to classify the MNIST handwriting dataset with a test accuracy of above 98 percent, as well as to compare and contrast different optimizers.

Introduction:

Despite the fact that the dataset is effectively solved, it may be used to learn and practice how to build, analyze, and apply convolutional deep learning neural networks for image classification from the ground up. This includes how to create a robust test harness for estimating the model's performance, how to investigate model enhancements, and how to save and load the model to make predictions on new data. Because handwritten information differs from person to person, automatic recognition of handwritten digits from visual data can be challenging. My major goal for this project is to propose a simple convolutional neural network (CNN) model to classify the MNIST handwriting dataset with a test accuracy of above 98 percent, as well as to compare and contrast different optimizers.

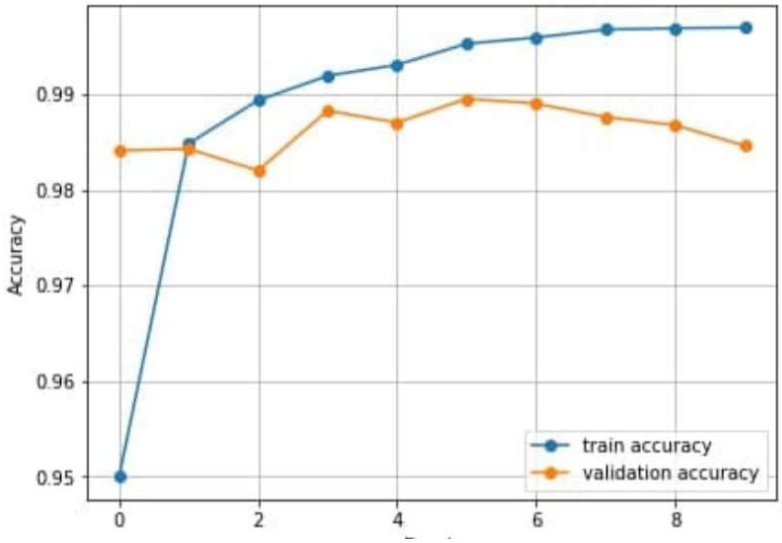
```
Model: "sequential_3"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_3 (Conv2D)            (None, 26, 26, 64)        640
_________________________________________________________________
max_pooling2d_3 (MaxPooling2 (None, 13, 13, 64)        0
_________________________________________________________________
conv2d_4 (Conv2D)            (None, 11, 11, 56)        32312
_________________________________________________________________
max_pooling2d_4 (MaxPooling2 (None, 5, 5, 56)          0
_________________________________________________________________
flatten_3 (Flatten)          (None, 1400)              0
_________________________________________________________________
dense_6 (Dense)              (None, 64)                89664
_________________________________________________________________
dense_7 (Dense)              (None, 10)                650
=================================================================
Total params: 123,266
Trainable params: 123,266
Non-trainable params: 0
_________________________________________________________________
```

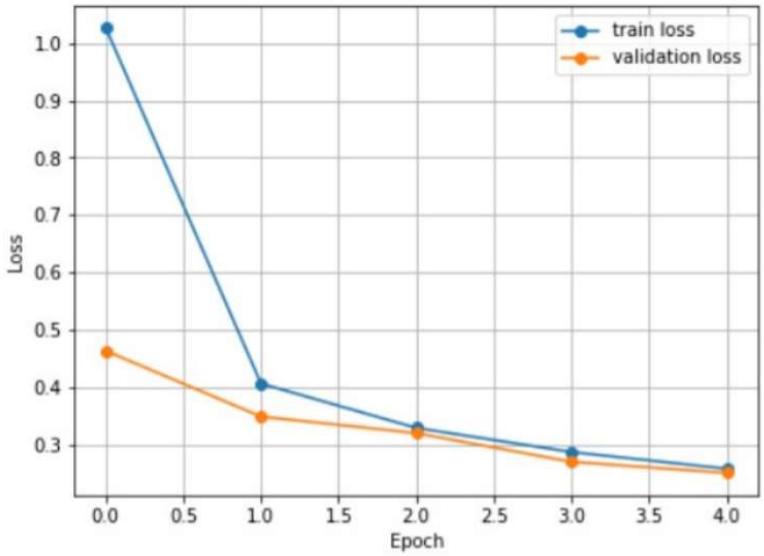
```
Model: "sequential_2"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv1d_2 (Conv1D)            (None, 24, 32)            4512
_________________________________________________________________
max_pooling1d_2 (MaxPooling1 (None, 12, 32)            0
_________________________________________________________________
conv1d_3 (Conv1D)            (None, 10, 64)            6208
_________________________________________________________________
max_pooling1d_3 (MaxPooling1 (None, 5, 64)             0
_________________________________________________________________
flatten_2 (Flatten)          (None, 320)               0
_________________________________________________________________
dense_4 (Dense)              (None, 128)               41088
_________________________________________________________________
dense_5 (Dense)              (None, 10)                1290
=================================================================
Total params: 53,098
Trainable params: 53,098
Non-trainable params: 0
_________________________________________________________________
```

Result:

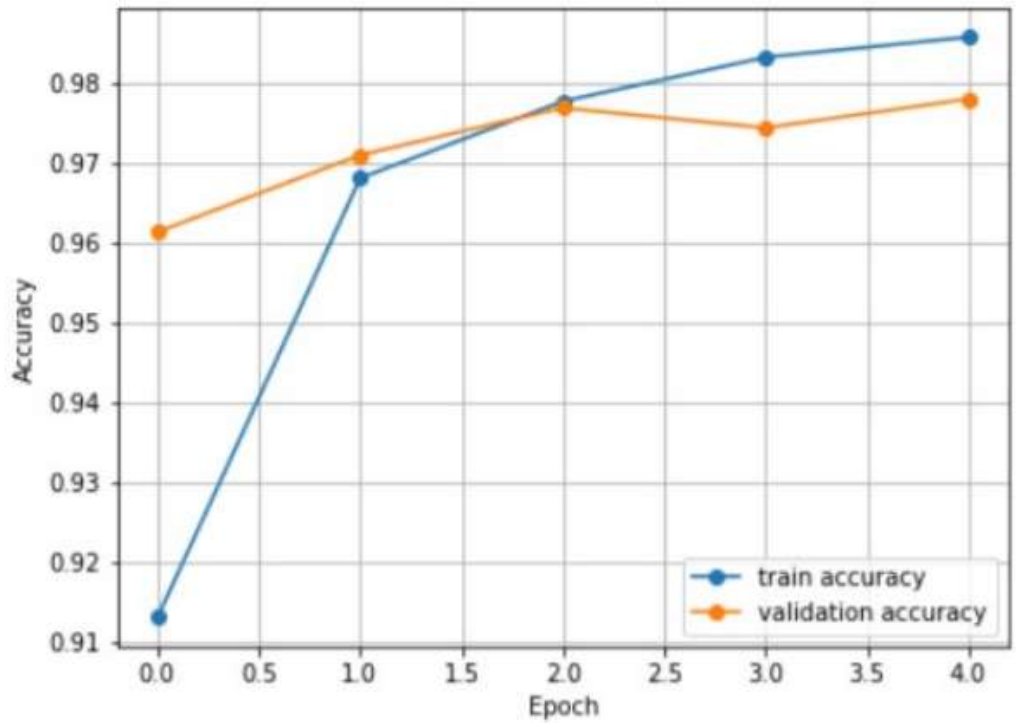
CNN model 1 Loss For SGD

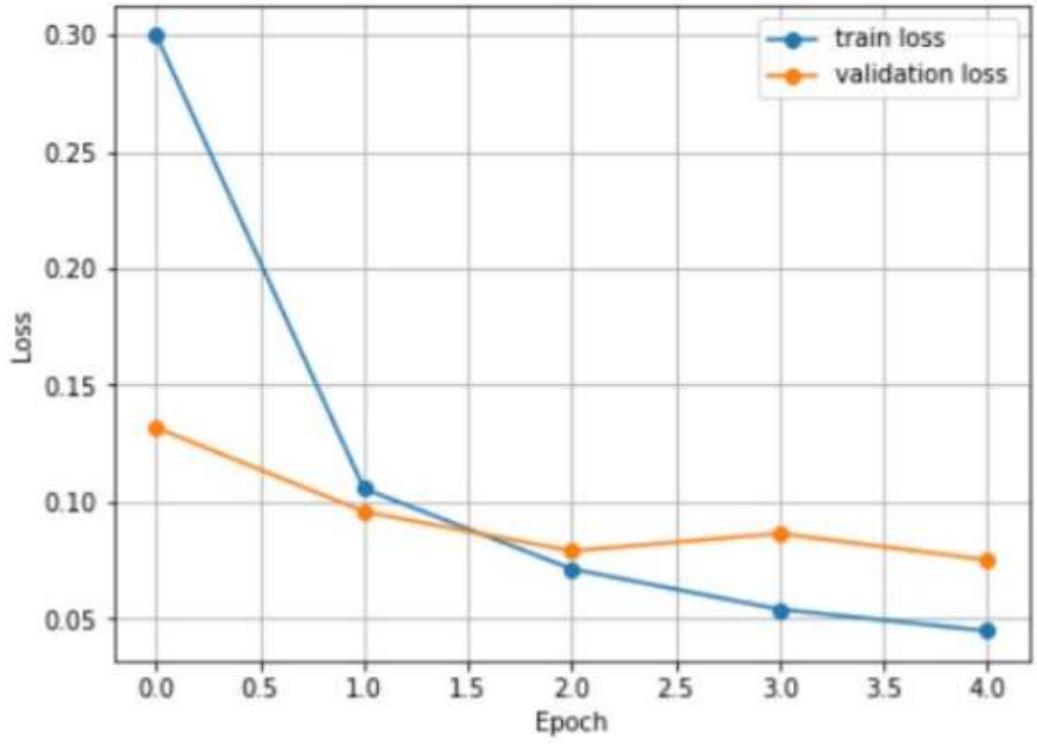
Optimizer	Train Accuracy(percent)	Validation Accuracy(percent)	Test Accuracy(percent)
RMSprop	99.53%	98.45%	98.72%
SGD	92.08%	92.27%	93.66%
Adam	98.66%	97.96%	98.33%



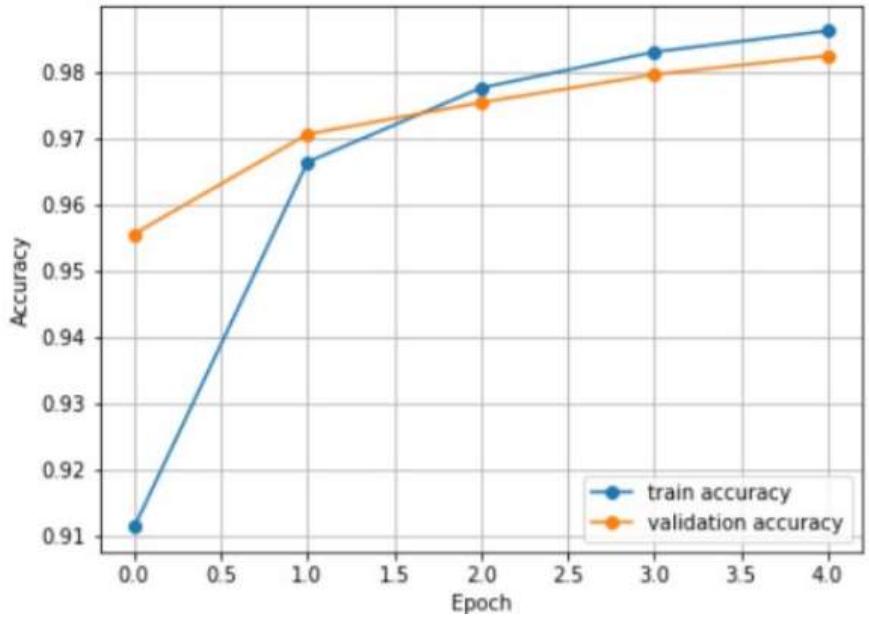


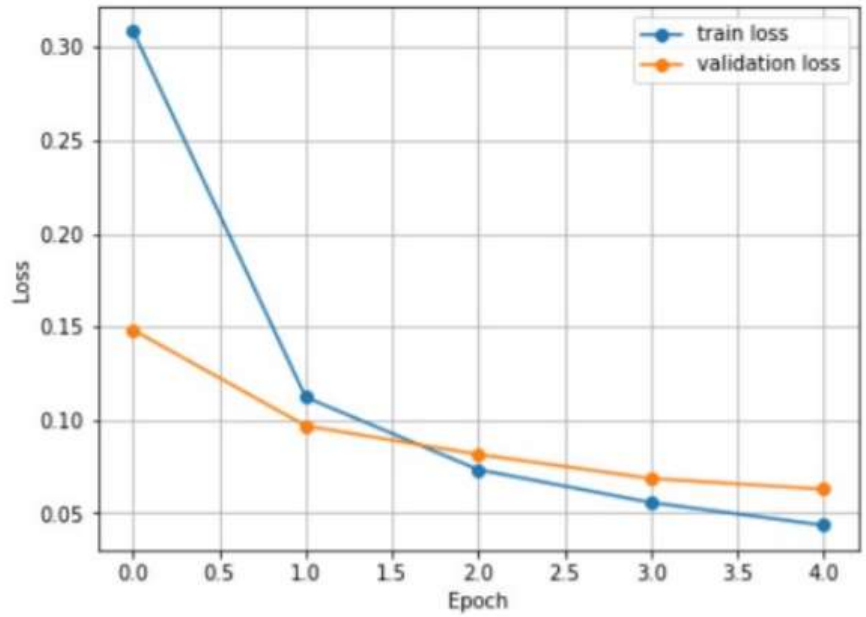
CNN model 1 Loss For ADAM





CNN model 1 Loss For RMSprop





Optimizer	Train Accuracy(percent)	Validation Accuracy(percent)	Test Accuracy(percent)
RMSprop	99.39%	98.50%	98.79%
SGD	93.79%	94.05%	94.49%
Adam	99.39%	98.10%	98.36%

Discussion:

However, the graph analysis reveals a disparity in Train and Validation accuracy, implying that the model would not perform consistently in real-world data. The RMSprop optimizer of the 'Model 1' graph shows a relatively equal rate of Train and Validation accuracy, indicating that the model will perform better in real-world data.

RMSprop exhibited the best validation and testing accuracy, as well as the least training loss. It is not the ideal solution because it has the largest validation loss. Taking into account all of the information, we can infer that employing the SGD optimizer will get the best results from this model. It has the best training and testing accuracy as well as the least validation loss.