MID PROJECT REPORT

1 Abstract

Handwritten digit recognition refers to a computer's capacity to recognize human handwritten digits from various sources such as photographs, papers, touch displays, and so on. Because everyone's handwriting is different. The purpose of this research is to develop a CNN (Convolutional Neural Network) model for the MNIST dataset that has a 98 percent accuracy. To achieve the best possible outcome, three distinct optimizers (Adam, SGD, and RMSprop) will be applied. [1]

2 Introduction

A convolutional neural network (CNN) is a neural network with one or more convolutional layers and is primarily used for image processing, classification, segmentation, and other autocorrelation data. CNNs have been used to understand natural language processing (NLP) and speech recognition, while recurrent neural networks (RNNs) are often used for NLP. Each convolution layer contains several filters called the convolution kernel. A filter is a matrix of integers used for a subset of input pixel values that are the same size as the kernel. Each pixel is multiplied by the corresponding value in the kernel and the result is summed to a single value to represent a grid cell such as a pixel in the output channel / feature map for convenience. The MNIST (Modified National Institute of Standards and Technology) database of handwritten numbers contains a training set of 60,000 cases and a test set of 10,000 cases. Each image contains 28 * 28 pixels, and each pixel has a value between 0 and 255. The model used four different planes, including the Conv2D plane, MaxPooling2D plane, flattening plane, and high density plane, and used different optimizers to achieve maximum accuracy.I have used two activation functions which includes *ReLU* and *Softmax* My model Summary is given below-

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 24, 24, 32)        832
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 12, 12, 32)        0
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 10, 10, 64)        18496
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 5, 5, 64)          0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 3, 3, 64)          36928
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 1, 1, 64)          0
_________________________________________________________________
flatten (Flatten)            (None, 64)                0
_________________________________________________________________
dense (Dense)                (None, 64)                4160
_________________________________________________________________
dense_1 (Dense)              (None, 10)                650
=================================================================
Total params: 61,066
Trainable params: 61,066
Non-trainable params: 0
_________________________________________________________________
```

Figure 1: Model Summary

Form this model summary we can see, I have used two conv2D layers and two MaxPooling2D layers. I have also used a flatten layer and two dense layer where one of them is the output layer

3 Results

The result of the model using different optimizers is given below –

Optimizer	Train Accuracy	Train Loss	Validation Accuracy	Validation Loss	Test Accuracy
Adam	99.69%	0.89%	98.46%	6.95%	98.67 %
SGD	100%	0.059%	99.07%	4.64%	99.22%
RMSprop	99.99%	0.026%	99.14%	13.57%	99.17%

The training and validation accuracy, training validation loss after using Adam optimizer is shown in the graph below-

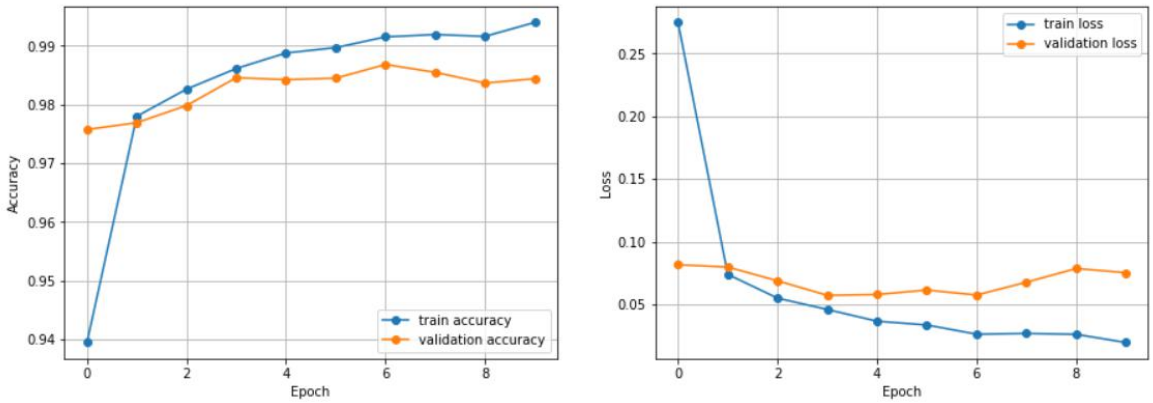

Figure 2: Graph data of Adam Optimizer

The training and validation accuracy, training validation loss after using SGD optimizer is shown in the graph below-

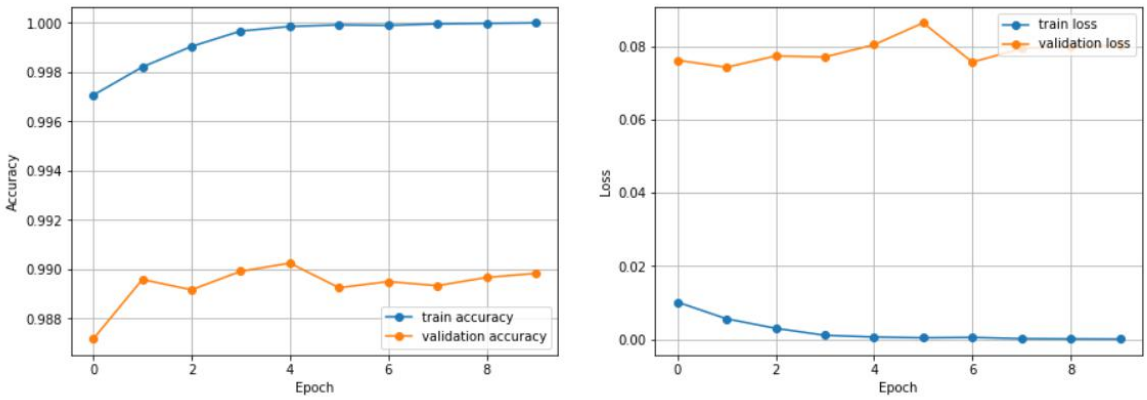

Figure 3: Graph data of SGD Optimizer

The training and validation accuracy, training validation loss after using RMSprop optimizer is shown in the graph below-

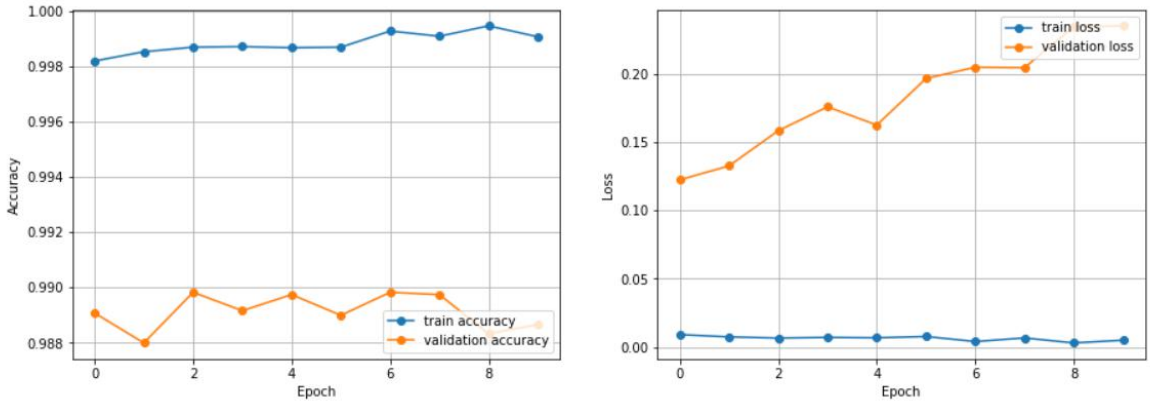

Figure 4: Graph data of RMSprop Optimizer

4 Discussion

From the table data we can see that, on all three optimizer the proposed model accuracy is over 98%. While using Adam optimizer the training accuracy is 99.69%, testing accuracy is 98.67% and the validation accuracy is 98.46%. In case of train loss and validation loss the data is 0.89% and 6.95% respectively. While using SGD optimizer the training accuracy is 100%, testing accuracy is 99.22% and the validation accuracy is 99.07%. In case of train loss and validation loss the data is 0.059% and 4.64% respectively. While using RMSprop optimizer the training accuracy is 99.99%, testing accuracy is 99.17% and the validation accuracy is 99.14%. In case of train loss and validation loss the data is 0.026% and 13.57% respectively. Although, RMSprop closest value with validation accuracy and testing accuracy and had the lowest training loss It is not the best option as the validation loss is the highest. Considering all the data, it can be concluded that we can get the best output from this model by using SGD optimizer. It gives the highest training and testing accuracy with the lowest validation loss.