Deep Learning

Programming Assignment #3

CNN

Professor:
Mitesh Khapra

Pawandeep Singh (CS17S027)
Sidharth Aggarwal (CS17S012)

1.1 Configration:Best Model

- Batch Size - 4000
- Learning Rate - 0.001
- Fully Connected Layers with 1024 Input
- Softmax Layer with Output 10
- Epochs - 130
- Convolutional Layer Detail
  - $3 * 3$ Kerne
  - Strid - 1
  - Padding - 1
- Pooling Layer Detail
  - 2*2 Kernel
  - Strid - 2
  - Padding - 1

1.2 Plots

In this section we have shown the plots for accuray and the loss during the training and validation. We have observed that the as the loss of the training data decreases the loss for the validation also decreases but at a certain point the validation starts increase though the loss of the training is reducing. So the plots are as below

As you can see in the above graph of the Training and the Validation Loss as the epochs are increasing the loss is decreasing as the model is being trained according to the training data.

Similarly in the Accuracy graph, as the model is getting trained epoch by epoch the accuracy is increasing.

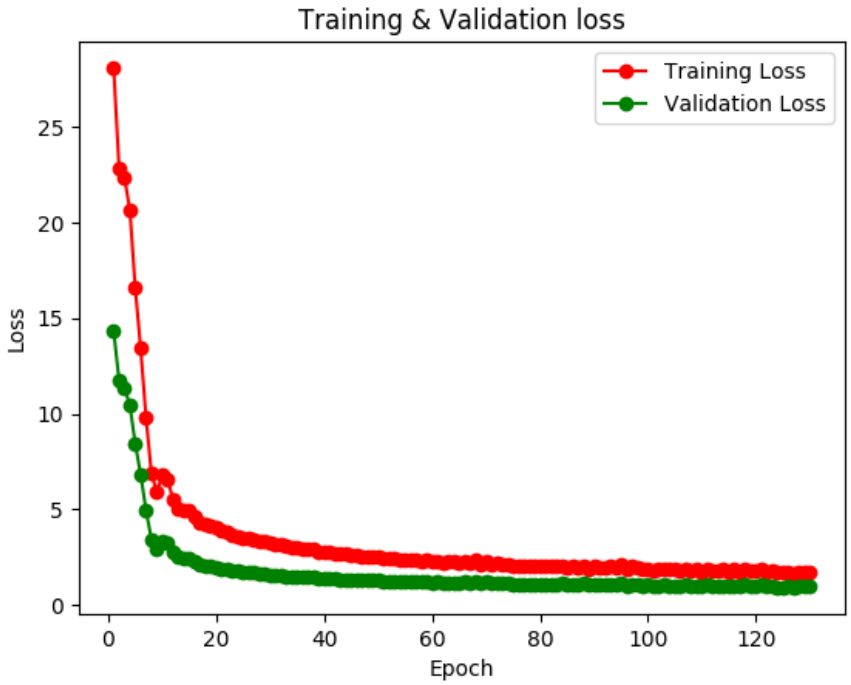

Figure 1.1: Training And Validation Loss

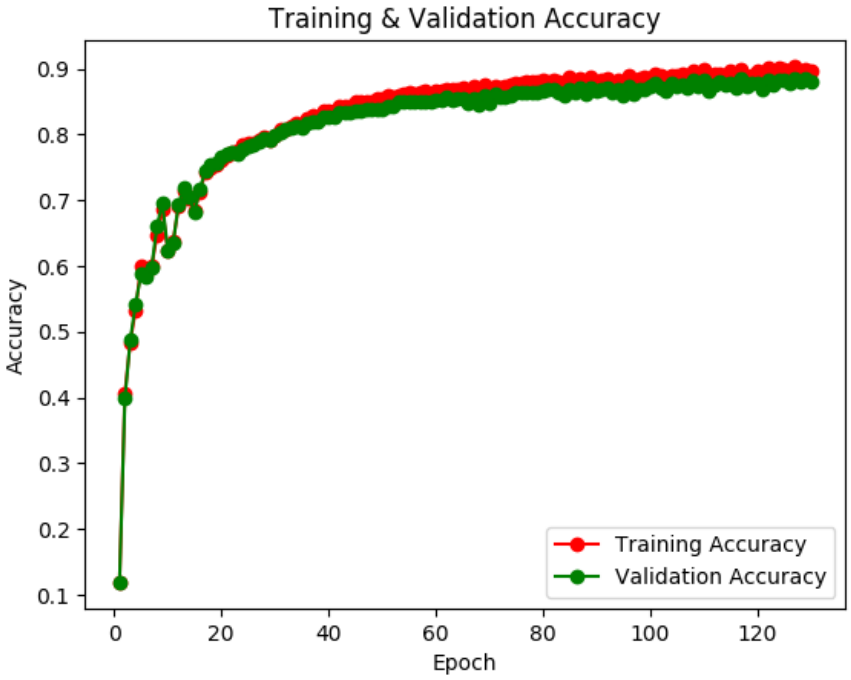

Figure 1.2: Training And Validation Accuracy

1.3 Performance On Test Data

Using our best configration which has learning rate of 0.001, batch size of 4000 and using Xavier initialization with 130 epochs, we are getting an accuracy of 90.5% on Kaggle on the public test data.
We have also tried the Data Augmentation by modifying the data into several forms such as horizontal flip, vertical flip, Blurring and rotating. Using Data Augmentation we we getting 88.8% accuacy on Kaggle.

1.4 Experiments Done On Parameter

Batch Size	**Learning Rate**	**Train Acc.%**	**Val Acc. %**
2000	0.01	88.2	86.4
4000	0.01	88.3	89.1
2000	0.001	91.3	92.7
4000	0.001	92.4	93.2

Table 1.1: Training & Validation Accuracy for varying Batch sizse & Learning Rate.

Stride	**Padding**	**Train Acc.%**	**Val Acc. %**
1	0	90.2	91.3
1	1	90.1	90.8
2	0	89.6	88.3
2	1	89.2	89.4

Table 1.2: Training & Validation Accuracy for varying Stride & Padding for best model. & Learning Rate.

FC. Layers	**Neurons/Layer**	**Train Acc.%**	**Val Acc. %**
1	512	85.6	87.3
1	1024	87.1	87.8
2	512	89.5	88.3
2	1024	91.4	90.6
3	512	91.6	90.2
3	1024	92.4	93.2

Table 1.3: Training & Validation Accuracy for varying number of FC layers and neurons per FC layer.

1.5 Parameter Setting For Best Results

We have experimented a lot by changing the different parameters like learning rate, batch size, initialization, epoch, kernel sizes of the convolutional and pooling layer.
We have got the Best Accuracy on the test data on following parameter values:-

- Batch Size - 4000
- Initialization - Xavier
- Epochs - 130
- Learning Rate - 0.001

1.6 Dimension of Input And Output

1.6.1 Input

Input :- Batch*28 *28 *1

1.6.2 CONV1

Input :- 64*28*28
Output :- 64*14*14

1.6.3 POOL1

Input :- 64*14*14
Output :- 128*14*14

1.6.4 CONV2

Input :- 128*14*14
Output :- 128*7*7

1.6.5 POOL2

Input :- 128*7*7
Output :- 256*7*7

1.6.6 CONV3

Input :- 256*7*7
Output :- 256*7*7

1.6.7 CONV4

Input :- 256*7*7
Output :- 256*4*4

1.6.8 POOL3

Input :- 256*4*4
Output :-256*2*2

1.6.9 FC1

Input :- 256*2*2
Output :- 256*2*2

1.6.10 FC2

Input :- 256*2*2
Output :- 256*2*2

1.6.11 SOFTMAX

Input :-256*2*2
Output :- 10

1.7 Number Of Parameter

- Total Parameters - 4222154
- Number Of Fully Connected Layers :- 4215818
  (1024*10 + 10) + (4 * 4 * 256*1024)+(1024*10 + 1024)
- Number Of Convolutional Layers :- 6336
  (6433)+(12833)+(25633)+(25633)

1.8 Number Of Neurons

- Total Neurons - 103194
- Number Of Fully Connected Neurons :- 2058
- Number Of Convolutional Neurons :- 101136

1.9 Effect Of Batch Normalisation

As we have studied about the Batch Normalisation technique in the neural network that it reduces the internal covariate shift in neural networks. And also we observed that after applying the batch normalisation the learning was happening faster that without it.

1.10 Filter Plot

In this section we have plotted the all the filters of first layer.

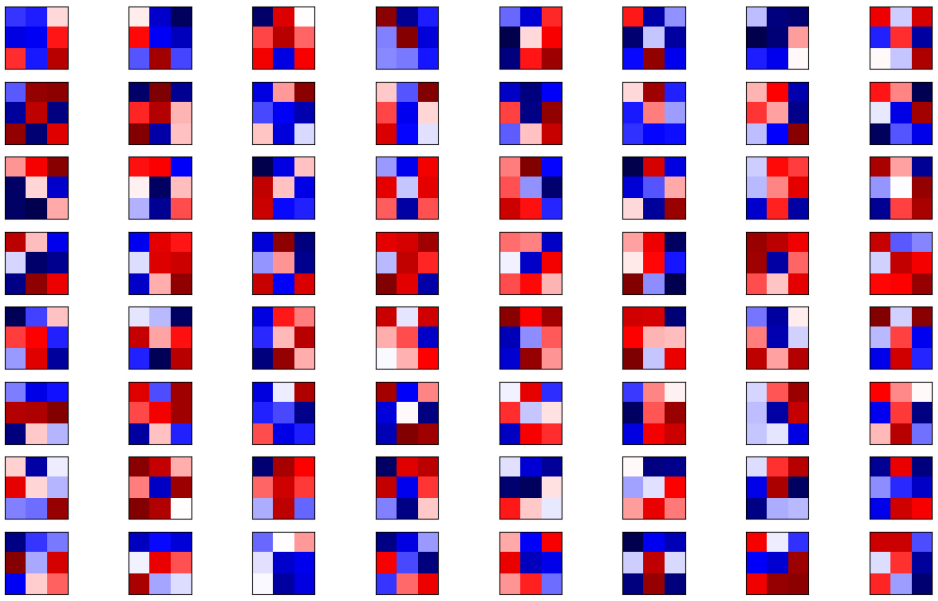

Figure 1.3: 64 Filter Plot For Layer1

1.11 Extras

1.11.1 Data Augmentation

We have applied the data augmentation for increasing the data. In this we have tried to augment in five ways as below:-

- Horizontal Flip

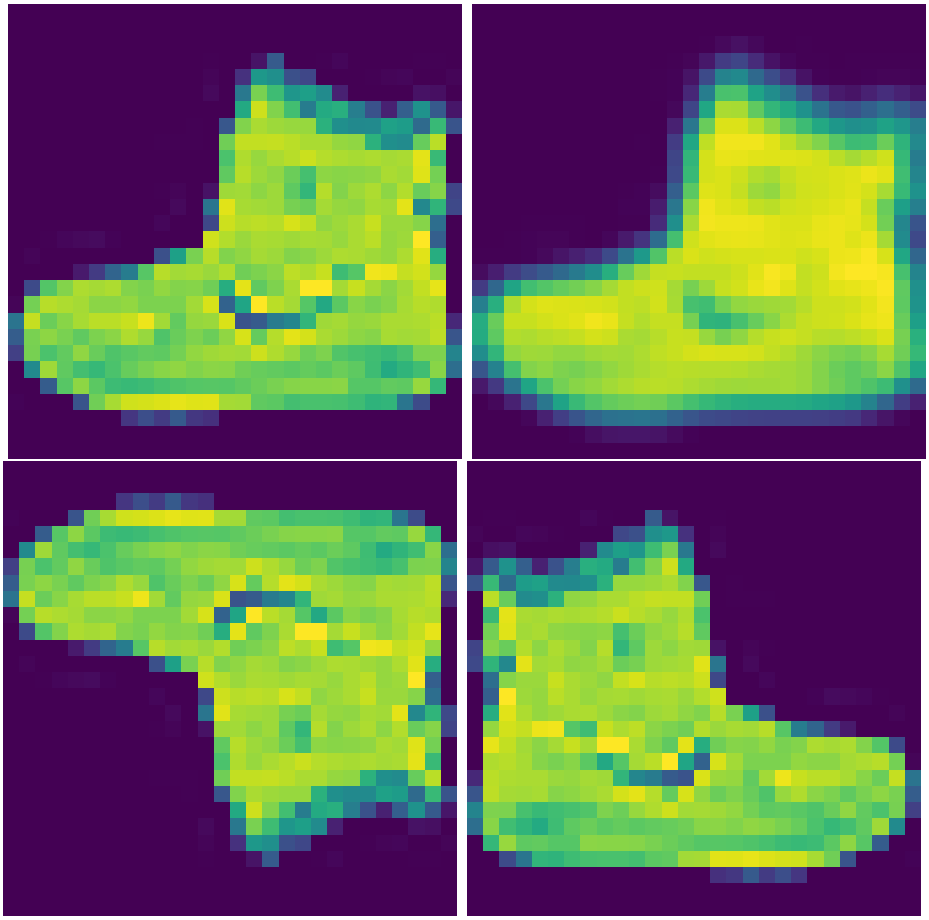

Figure 1.4: Images For Data Augmentation.a)Original b) Blur c)HorizontalFlip d)Vertical Flip

- Vertical Flip
- Gaussian Noise
- Left Rotation 45
- Right Rotation 45

For the Data Augmentation we have plotted the graph for Accuracy and Loss between the Training and Validation. As we told earlier that we got 88.8% accuracy. So the plots are as given below:-

1.11.2 DeepDream

We have tried to implement the DeepDream. We were almost there to implement it. We have taken the refernce from web to implement the DeepDream. We have attached the code for the DeepDream using tensorflow

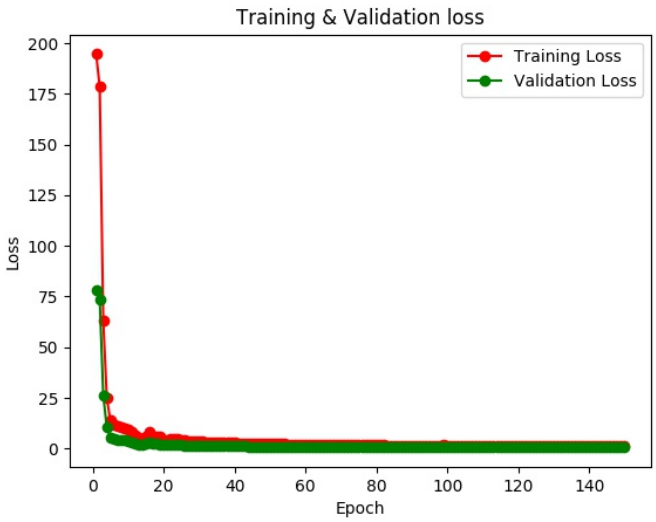

Figure 1.5: Training And Validation Loss using Data Augmentation

1.12 Implementation Support

- Batch Initialization
  - Xavier
  - He
- Batch Size
- Learning Rate
- Data Augmentation
- Partial DeepDream

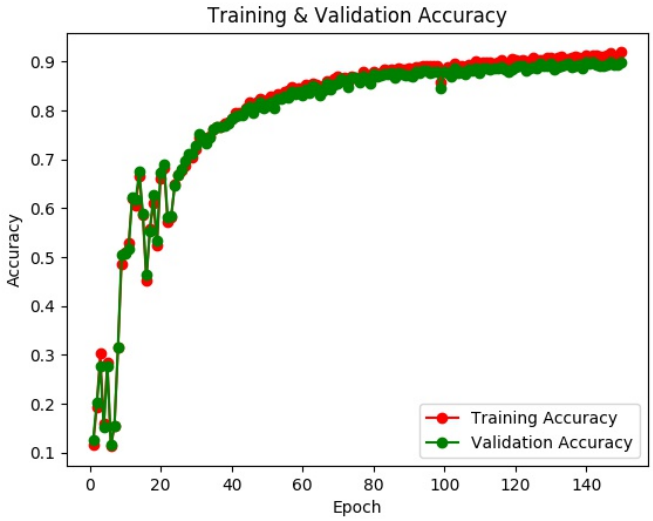

Figure 1.6: Training And Validation Accuracy using Data Augmentation