Word Report: Programming Assignment 2

Subject: Machine Learning (CSE 6363 – 004)

UTA ID: 1002073011

Student Name: Prayag Axaykumar Parikh

NOTE: Once you run all the cells from ML_PA2.ipynb, please wait for up to 5-7 minutes, because 54000 training images over 15 epochs is going to take some time for LeNet-5 model to get trained.

Results on MNIST dataset

1. MNIST dataset has (28,28,1) size grayscale images. 60000 Train images and 10000 Test images are downloaded from Kaggle and kept in the '**archive**' folder of the uploaded zip folder.
2. Since LeNet-5 is using 3 convolutional layers of (5x5) size kernel, I am padding the input images and the input dimensions = (32,32) before passing to the first convolutional layer.
3. For Validation purpose I am taking 10% of the train images, i.e. 60000 x 0.10 = 6000 images as validation train images, and remaining 54000 images altogether are going to be used for this model.
4. Also, 10000 test images are there in the dataset.
5. Results of Training accuracies for first 10 epochs is as below:

```
Training Accuracy for epoch 1: 0.4669
Training Loss for epoch 1: 1693.6963
Validation Accuracy for epoch 1: 0.7183
Training Accuracy for epoch 2: 0.7568
Training Loss for epoch 2: 821.8541
Validation Accuracy for epoch 2: 0.8293
Training Accuracy for epoch 3: 0.8271
Training Loss for epoch 3: 529.6815
Validation Accuracy for epoch 3: 0.8695
Training Accuracy for epoch 4: 0.8617
Training Loss for epoch 4: 387.8273
Validation Accuracy for epoch 4: 0.8905
Training Accuracy for epoch 5: 0.8839
Training Loss for epoch 5: 292.1582
Validation Accuracy for epoch 5: 0.9030
Training Accuracy for epoch 6: 0.8981
Training Loss for epoch 6: 213.9399
Validation Accuracy for epoch 6: 0.9120
Training Accuracy for epoch 7: 0.9094
Training Loss for epoch 7: 156.3288
Validation Accuracy for epoch 7: 0.9228
Training Accuracy for epoch 8: 0.9173
Training Loss for epoch 8: 123.3446
Validation Accuracy for epoch 8: 0.9280
Training Accuracy for epoch 9: 0.9239
Training Loss for epoch 9: 100.6871
Validation Accuracy for epoch 9: 0.9333
Training Accuracy for epoch 10: 0.9308
Training Loss for epoch 10: 78.6964
Validation Accuracy for epoch 10: 0.9383
```

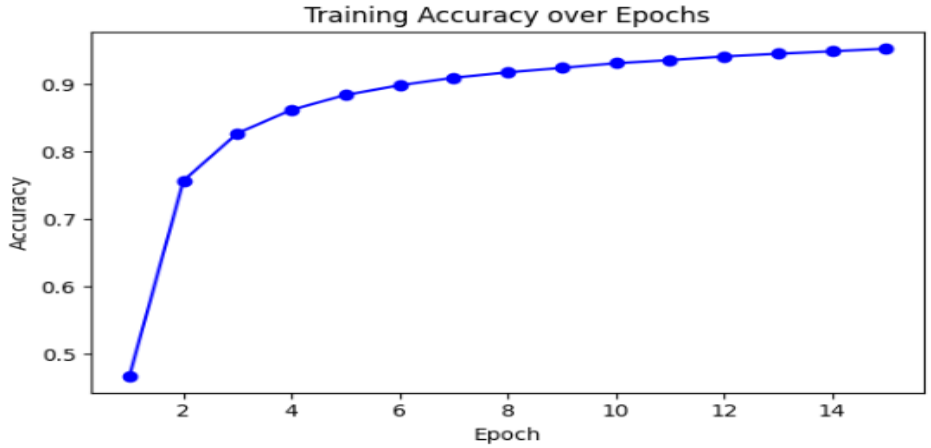
6. As we can see that for the 1st epoch training accuracy was just 46.69% which increased over first 10 epochs and became 93.08% for 10th epoch and 95.24% for 15th epoch.
7. During the first epoch Validation accuracy is 0.71 but Training accuracy is too low comparatively just 0.47 that means Training Time is Inadequate and it needs more training for 54000 images. Therefore, by the 10th epoch Validation accuracy and Training accuracy are almost same 0.93, which means that now, the model has been trained for adequate amount of time.
8. Here, the usage of Validation accuracy is to identify that how many epochs are needed for model to get trained and another reason is to check, **Generalization**: If the accuracies are high and similar, it suggests that the model is learning patterns from the training data that are applicable to new, unseen data. This is a positive outcome.

Finally, after 15 epochs:
Validation accuracy (10% of train data): 0.9552
Testing accuracy: 0.9454

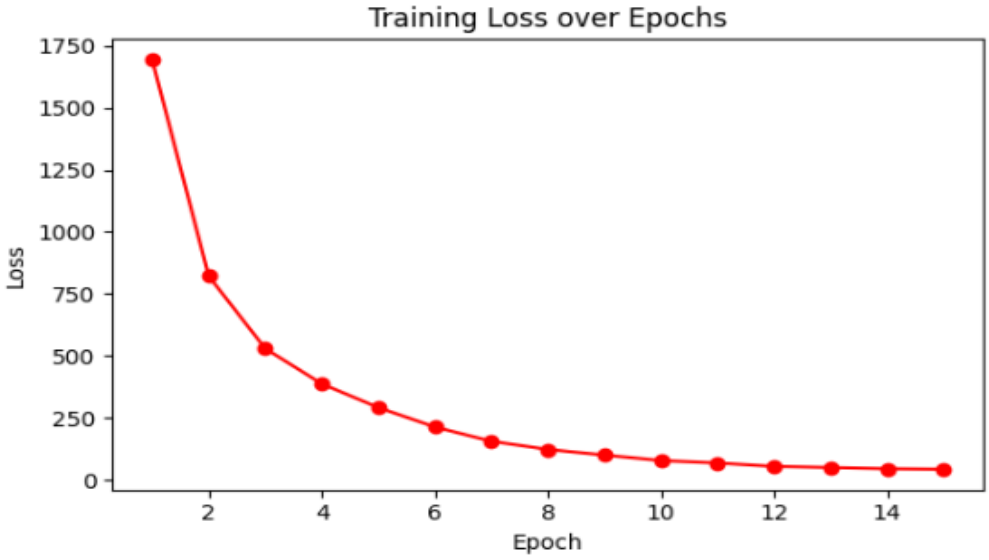
9. Testing accuracy is the final accuracy as the product we can deliver to the customer and can be used now, because test dataset was never used during the training process, so it's completely unseen dataset.

Analysis of Results on MNIST

1. I am using two different graphs to show the performance of the LeNet-5 model on MNIST dataset.
   First: Training Accuracy per epoch (increasing)
   Second: Loss per epoch (decreasing)
2. Training Accuracy per epoch:
   i. It's increasing drastically till 7 epochs and then it's increasing steadily but gradually, the reason is that initially when the training time is not enough for the model, it underfits the training images therefore accuracy is very low, after a certain time it improves the accuracy and after some time, when the weights and biases are trained almost optimally, it doesn't have much scope of learning and accuracy increases with not a huge difference.



3. Loss per epoch:
   i. Over the period of 15 epochs, Loss is decreasing with the same flow as the accuracy is increasing, in a way both accuracy and loss are inversely proportional to each other, therefore the amount of accuracy increases the same amount of loss reduction can be seen from the below graph.



REFERENCES (used during assignment)

1. Reading MNIST dataset: https://www.kaggle.com/code/hojjatk/read-mnist-dataset/notebook

2. To understand architecture in Tensorflow(Professor's recommendation): https://gist.github.com/Moataz-E/6751b1b92fe8f4ff617f10c7f9f9d315

3. Using tf.randon.normal for generating weights: https://www.tensorflow.org/api_docs/python/tf/random/normal

4. Basic training loops: https://www.tensorflow.org/guide/basic_training_loops

5. Understanding of Flatten layer: https://stackoverflow.com/questions/43237124/what-is-the-role-of-flatten-in-keras

6. Adam optimizer: https://www.tensorflow.org/api_docs/python/tf/compat/v1/train/AdamOptimizer