Name : Doaa Samir

ID : 2103114

Name : Aya Abdelmoniem

ID : 20221462478

NN Project's Report

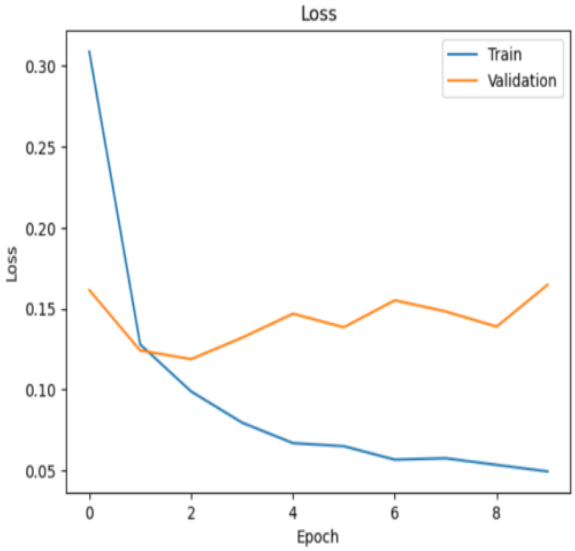
step 1

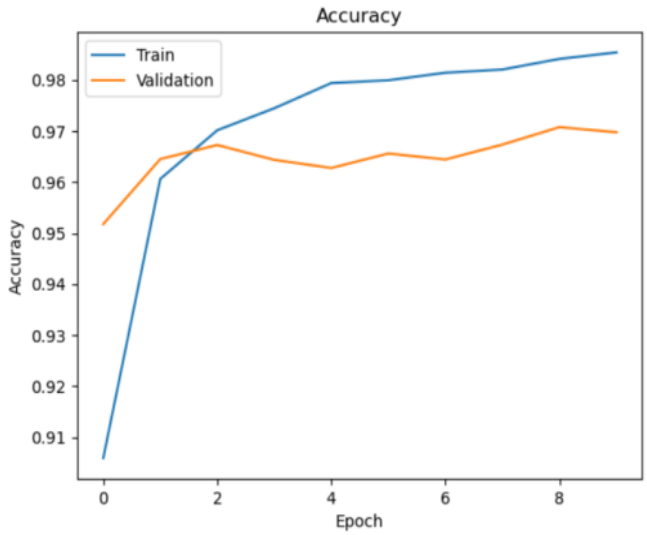
First I am going to download the MNIST dataset from kaggel and then apply any preprocessing needed on the data. Then I am going to split the data randomly into being 80% training set and 20% validation set. We are going to convert its type to numpy to convert it to tensor. CTDataset is normalizing the features (pixels values) between 0 and 1 and the labels converts it to units vector where the 1 is our location of what class it is the label determines the element with the highest value and represents its location as a one-hot vector/tensor the known probability that an image corresponds to a certain class.

Step 2

Dropout is a technique used in neural networks to reduce overfitting. It is a technique where randomly selected neurons are dropped out during training. Therefore, the neural network is forced to rely on the remaining neurons to make predictions, which helps in preventing overfitting. Layer normalization is a normalization technique used in neural networks to improve the stability and performance of a model. Layer normalization normalizes over the features of each individual

sample. This will reduce the dependency on the specific batch examples and will allow for more stable and strong training.



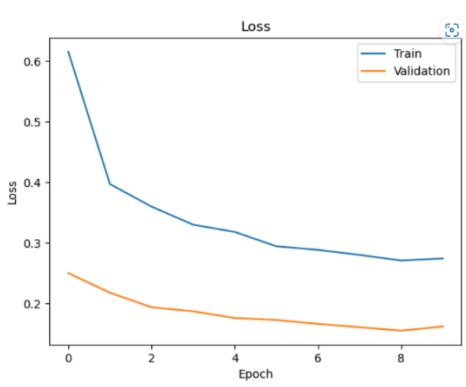


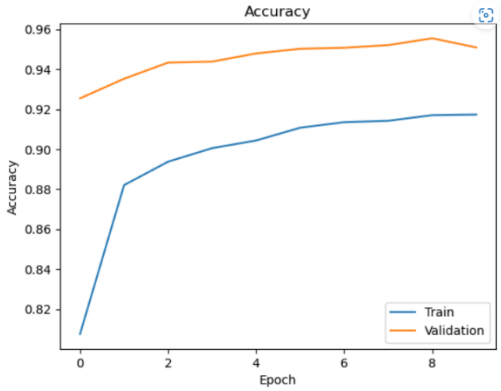
Here I have the loss and accuracy plot without any dropout or layer normalization. In the loss plot the validation starts with decreasing and then it keeps on increasing and decreasing continuously while the epoch is increasing. In the training loss it keeps on decreasing while epoch is increasing. In the accuracy plot the train is increasing while the epoch is increasing and the validation accuracy keeps on decreasing and increasing continuously. From the plots there is overfitting.

Step 3 :

After Add Dropout and Layer Normalization layers the overfitting reduces but with different probability of dropping out there is different results

1- Here with 70 percentage dropping out of nodes :
   Gives better result and reduce overfitting , in plots the accuracy of validation set Is better than the accuracy in training and it seems that both will not intersect and When training 70 percentage of the features are set to zero , When validation all features are used So the model at validation can lead to higher accuracies and this percentage thats forces the layers to take less responsibility for the input this ensures that the model is getting generalised and hence reducing the overfitting problem but too high a dropout rate can slow the convergence rate of the model as shown in the loss plot as the final loss is 0.2 and that's hurt final performance and the model will not being able to fit perfectly

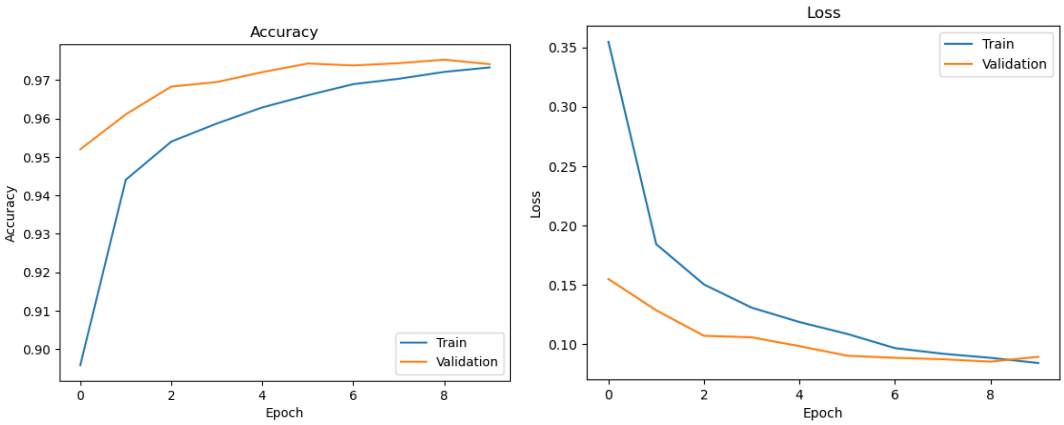




```
Epoch 0
Epoch [1/10], Train Loss: 0.6159, Train Acc: 0.8075, Val Loss: 0.2498, Val Acc: 0.9254
Epoch 1
Epoch [2/10], Train Loss: 0.3971, Train Acc: 0.8820, Val Loss: 0.2174, Val Acc: 0.9352
Epoch 2
Epoch [3/10], Train Loss: 0.3598, Train Acc: 0.8937, Val Loss: 0.1934, Val Acc: 0.9433
Epoch 3
Epoch [4/10], Train Loss: 0.3299, Train Acc: 0.9005, Val Loss: 0.1866, Val Acc: 0.9438
Epoch 4
Epoch [5/10], Train Loss: 0.3180, Train Acc: 0.9043, Val Loss: 0.1756, Val Acc: 0.9479
Epoch 5
Epoch [6/10], Train Loss: 0.2942, Train Acc: 0.9107, Val Loss: 0.1723, Val Acc: 0.9503
Epoch 6
Epoch [7/10], Train Loss: 0.2882, Train Acc: 0.9135, Val Loss: 0.1658, Val Acc: 0.9507
Epoch 7
Epoch [8/10], Train Loss: 0.2799, Train Acc: 0.9142, Val Loss: 0.1604, Val Acc: 0.9521
Epoch 8
Epoch [9/10], Train Loss: 0.2707, Train Acc: 0.9170, Val Loss: 0.1546, Val Acc: 0.9555
Epoch 9
Epoch [10/10], Train Loss: 0.2740, Train Acc: 0.9173, Val Loss: 0.1616, Val Acc: 0.9509
```

2- Here with 30 percentage of dropping out the nodes it reduces overfitting but it still there is overfitting as in accuracy plot the accuracy of validation intersect the accuracy of training and . In the loss plot the validation starts with decreasing and then it keeps on increasing while the epoch is increasing. In the training loss it keeps on decreasing while epoch is increasing and that's because of the percentage of drop out is not big enough few nodes will be dropped out and that reduces overfitting and make accuracy in validation decrease a little bit every epoch but still there is overfitting and that's mean the layers still take responsibility for the input

- layer normalization here provides faster training, better generalization, and also makes the network more stable and less sensitive to the scale of the inputs

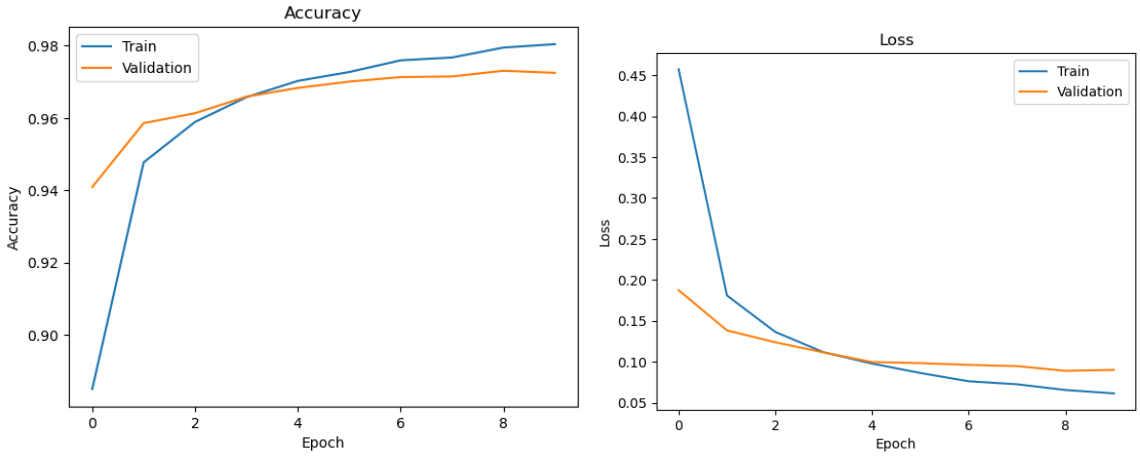



```
Epoch 0
Epoch [1/10], Train Loss: 0.3547, Train Acc: 0.8958, Val Loss: 0.1547, Val Acc: 0.9520
Epoch 1
Epoch [2/10], Train Loss: 0.1841, Train Acc: 0.9441, Val Loss: 0.1284, Val Acc: 0.9611
Epoch 2
Epoch [3/10], Train Loss: 0.1501, Train Acc: 0.9540, Val Loss: 0.1068, Val Acc: 0.9683
Epoch 3
Epoch [4/10], Train Loss: 0.1306, Train Acc: 0.9587, Val Loss: 0.1056, Val Acc: 0.9695
Epoch 4
Epoch [5/10], Train Loss: 0.1184, Train Acc: 0.9629, Val Loss: 0.0981, Val Acc: 0.9721
Epoch 5
Epoch [6/10], Train Loss: 0.1084, Train Acc: 0.9661, Val Loss: 0.0901, Val Acc: 0.9743
Epoch 6
Epoch [7/10], Train Loss: 0.0965, Train Acc: 0.9690, Val Loss: 0.0883, Val Acc: 0.9738
Epoch 7
Epoch [8/10], Train Loss: 0.0917, Train Acc: 0.9704, Val Loss: 0.0871, Val Acc: 0.9744
Epoch 8
Epoch [9/10], Train Loss: 0.0883, Train Acc: 0.9721, Val Loss: 0.0850, Val Acc: 0.9753
Epoch 9
Epoch [10/10], Train Loss: 0.0839, Train Acc: 0.9733, Val Loss: 0.0892, Val Acc: 0.9742
```

Step 4:

1. with learning rate 0.001 and the value of dropout is 0.2

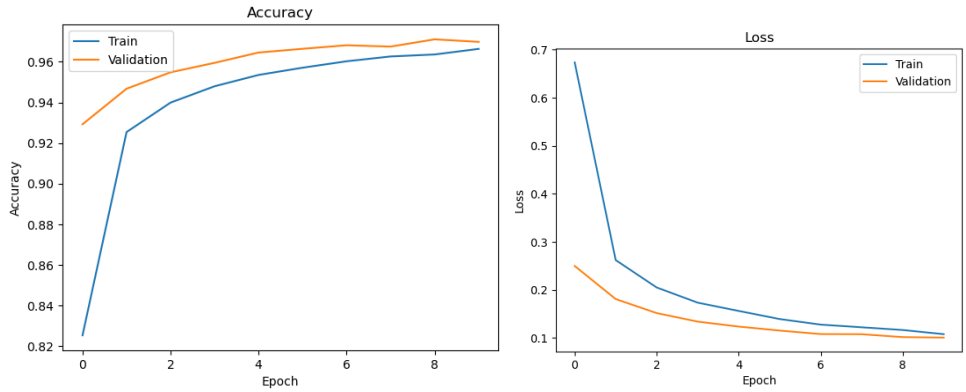
```
Epoch 0
Epoch [1/10], Train Loss: 0.4574, Train Acc: 0.8850, Val Loss: 0.1876, Val Acc: 0.9409
Epoch 1
Epoch [2/10], Train Loss: 0.1810, Train Acc: 0.9477, Val Loss: 0.1383, Val Acc: 0.9586
Epoch 2
Epoch [3/10], Train Loss: 0.1365, Train Acc: 0.9589, Val Loss: 0.1238, Val Acc: 0.9613
Epoch 3
Epoch [4/10], Train Loss: 0.1116, Train Acc: 0.9658, Val Loss: 0.1114, Val Acc: 0.9659
Epoch 4
Epoch [5/10], Train Loss: 0.0978, Train Acc: 0.9703, Val Loss: 0.0997, Val Acc: 0.9683
Epoch 5
Epoch [6/10], Train Loss: 0.0863, Train Acc: 0.9727, Val Loss: 0.0984, Val Acc: 0.9701
Epoch 6
Epoch [7/10], Train Loss: 0.0762, Train Acc: 0.9760, Val Loss: 0.0963, Val Acc: 0.9713
Epoch 7
Epoch [8/10], Train Loss: 0.0725, Train Acc: 0.9768, Val Loss: 0.0947, Val Acc: 0.9715
Epoch 8
Epoch [9/10], Train Loss: 0.0655, Train Acc: 0.9795, Val Loss: 0.0890, Val Acc: 0.9731
Epoch 9
Epoch [10/10], Train Loss: 0.0614, Train Acc: 0.9804, Val Loss: 0.0901, Val Acc: 0.9725
```



- Here with dropout with probability 0.2 and that's mean 20% of the nodes would be dropped it seems a few nodes will be dropped out and that reduces overfitting and make accuracy in validation decrease a little bit every epoch but still there is overfitting and that's mean the layers still take responsibility for the input
- The learning rate is good as its converge and reach the minima by has 0.0 is constant at least 5 epochs

2. with learning rate 0.0005 and the value of dropout is 0.4

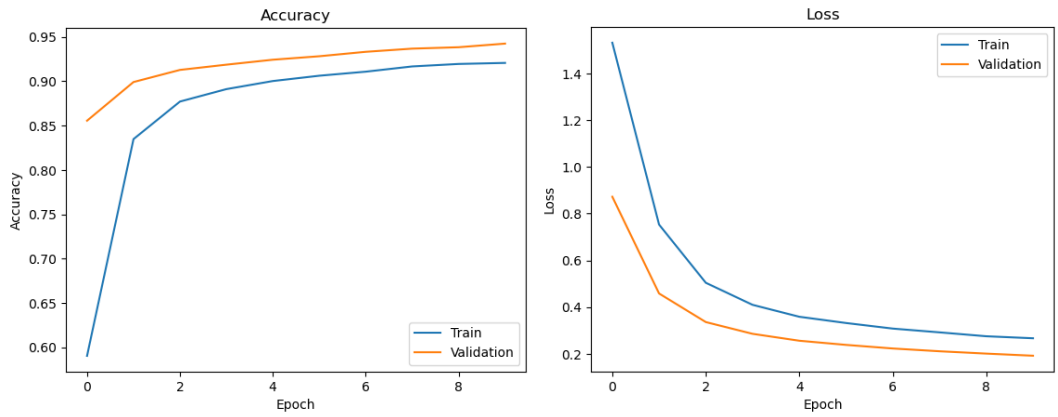
```
Epoch 0
Epoch [1/10], Train Loss: 0.6737, Train Acc: 0.8253, Val Loss: 0.2497, Val Acc: 0.9293
Epoch 1
Epoch [2/10], Train Loss: 0.2619, Train Acc: 0.9254, Val Loss: 0.1808, Val Acc: 0.9467
Epoch 2
Epoch [3/10], Train Loss: 0.2049, Train Acc: 0.9399, Val Loss: 0.1515, Val Acc: 0.9548
Epoch 3
Epoch [4/10], Train Loss: 0.1733, Train Acc: 0.9480, Val Loss: 0.1338, Val Acc: 0.9595
Epoch 4
Epoch [5/10], Train Loss: 0.1560, Train Acc: 0.9535, Val Loss: 0.1234, Val Acc: 0.9646
Epoch 5
Epoch [6/10], Train Loss: 0.1391, Train Acc: 0.9571, Val Loss: 0.1151, Val Acc: 0.9664
Epoch 6
Epoch [7/10], Train Loss: 0.1275, Train Acc: 0.9603, Val Loss: 0.1079, Val Acc: 0.9682
Epoch 7
Epoch [8/10], Train Loss: 0.1220, Train Acc: 0.9626, Val Loss: 0.1075, Val Acc: 0.9675
Epoch 8
Epoch [9/10], Train Loss: 0.1163, Train Acc: 0.9636, Val Loss: 0.1015, Val Acc: 0.9711
Epoch 9
Epoch [10/10], Train Loss: 0.1076, Train Acc: 0.9664, Val Loss: 0.1005, Val Acc: 0.9698
```



- learning rate here is small and it takes small steps to reach the minima and in specified epochs it stops at the point that is not the minimum because of the small steps it take and it require more epochs and updates to reach the minima here stops in loss=0.1 while with the learning rate 0.001 it reaches the minima with loss 0.06
- dropout with probability of dropping out the nodes is 0.4 and that's forces the layers to take less responsibility for the input but here with the plot said 0.4 is not enough for erase overfitting as it seems that the graph of accuracy in validation will make intersection with the graph of accuracy in training and seems with more epochs maybe there will be that the accuracy of validation will be less than training accuracy and the model won't perfectly generalised

3-with learning rate 0.0001 and the value of dropout is 0.6

```
Epoch 0
Epoch [1/10], Train Loss: 1.5319, Train Acc: 0.5903, Val Loss: 0.8724, Val Acc: 0.8555
Epoch 1
Epoch [2/10], Train Loss: 0.7532, Train Acc: 0.8347, Val Loss: 0.4582, Val Acc: 0.8991
Epoch 2
Epoch [3/10], Train Loss: 0.5040, Train Acc: 0.8771, Val Loss: 0.3357, Val Acc: 0.9128
Epoch 3
Epoch [4/10], Train Loss: 0.4096, Train Acc: 0.8911, Val Loss: 0.2855, Val Acc: 0.9187
Epoch 4
Epoch [5/10], Train Loss: 0.3585, Train Acc: 0.9002, Val Loss: 0.2559, Val Acc: 0.9243
Epoch 5
Epoch [6/10], Train Loss: 0.3317, Train Acc: 0.9062, Val Loss: 0.2380, Val Acc: 0.9282
Epoch 6
Epoch [7/10], Train Loss: 0.3079, Train Acc: 0.9108, Val Loss: 0.2226, Val Acc: 0.9332
Epoch 7
Epoch [8/10], Train Loss: 0.2917, Train Acc: 0.9167, Val Loss: 0.2108, Val Acc: 0.9367
Epoch 8
Epoch [9/10], Train Loss: 0.2750, Train Acc: 0.9195, Val Loss: 0.2008, Val Acc: 0.9383
Epoch 9
Epoch [10/10], Train Loss: 0.2664, Train Acc: 0.9206, Val Loss: 0.1917, Val Acc: 0.9424
```



- learning rate here is to small and that's mean it takes very small steps to reach the minima
  and in our specified epochs it will reach to loss = 0.2 ; with this learning it require more epochs to reach to the minima or to have learning rate bigger to train the model faster

- dropout with probability of dropping out the nodes is 0.6 in plots the accuracy of validation set Is better than the accuracy in training and it seems that both will not intersect and that's because of the behaviour when training and validation are different , When training 60 percentage of the features are set to zero , When validation all features are used So the model at validation can lead to higher accuracies and this percentage thats forces the layers to take less responsibility for the input this ensures that the model is getting generalised and hence reducing the overfitting problem.