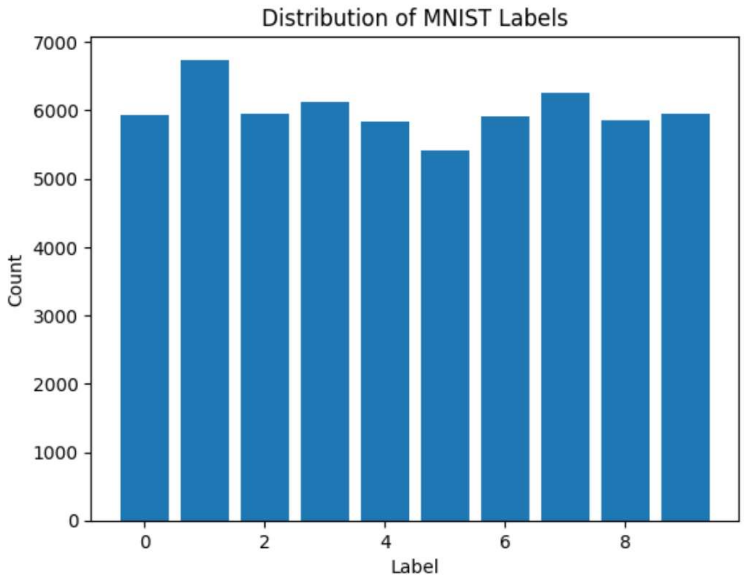
Part 4 - CNN and Autoencoders

Add a description...

Harshit Aggarwal

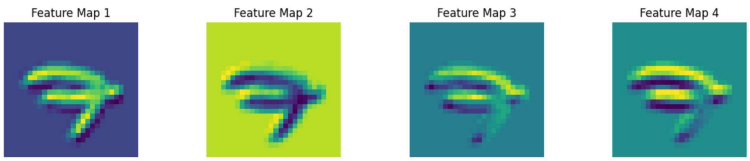
CNN is used on mnist data set and the hyperparameter tuning is done on the following metrics.

The dataset looks like:



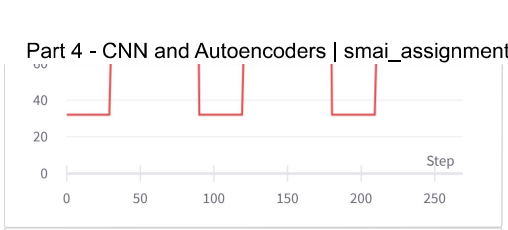
So, there is almost no class imbalance.

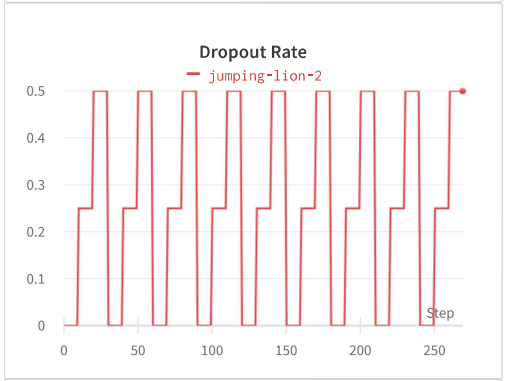
The feature maps look like:

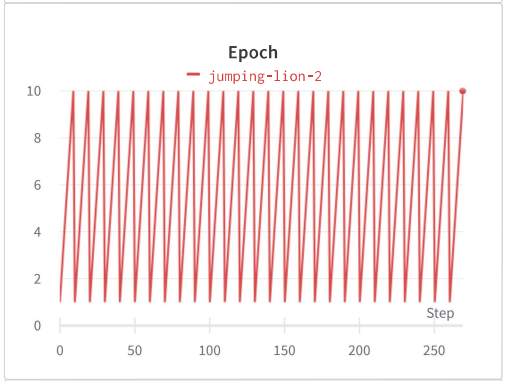


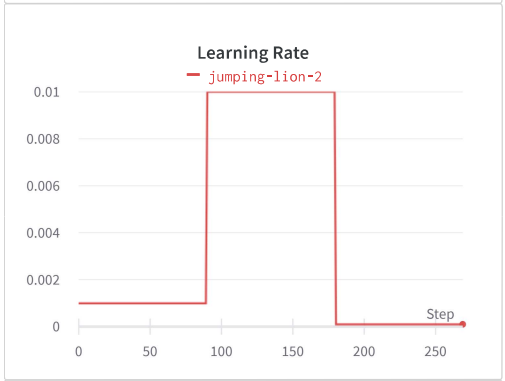

```
config.learning_rate = [0.001, 0.01, 0.0001]
config.batch_size = [32, 64, 128]
config.num_epochs = 10
config.dropout_rate = [0.0, 0.25, 0.5]
```

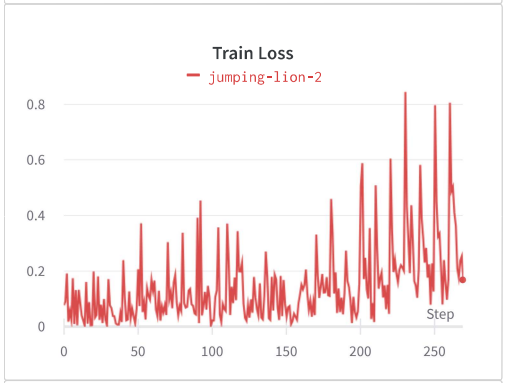
The graphs for this hyperparameter tunning are given below.

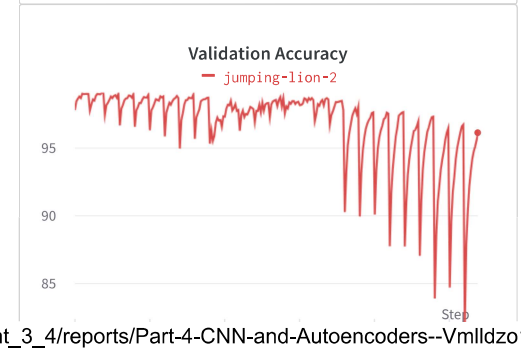

40
20
0
Step
0
50
100
150
200
250

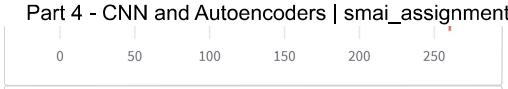

Dropout Rate
jumping-lion-2
0.5
0.4
0.3
0.2
0.1
0
Step
0
50
100
150
200
250

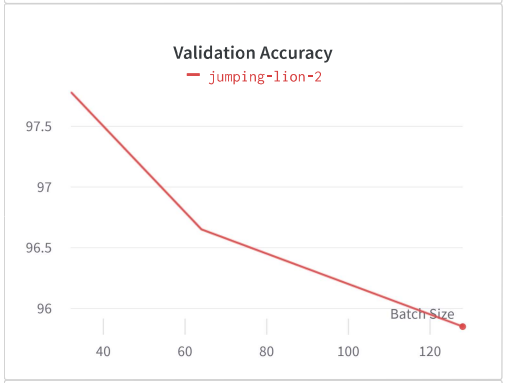

Epoch
jumping-lion-2
10
8
6
4
2
0
Step
0
50
100
150
200
250

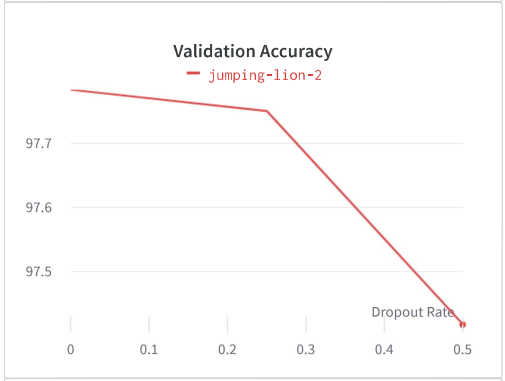

Learning Rate
jumping-lion-2
0.01
0.008
0.006
0.004
0.002
0
Step
0
50
100
150
200
250

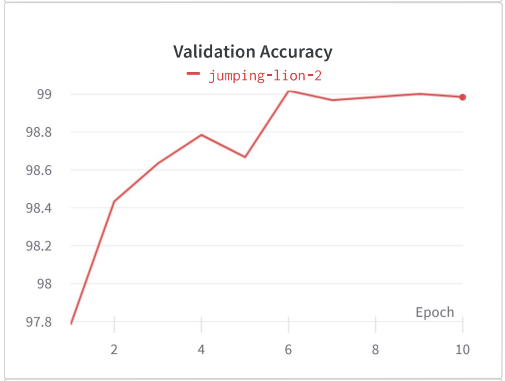

Train Loss
jumping-lion-2
0.8
0.6
0.4
0.2
0
Step
0
50
100
150
200
250

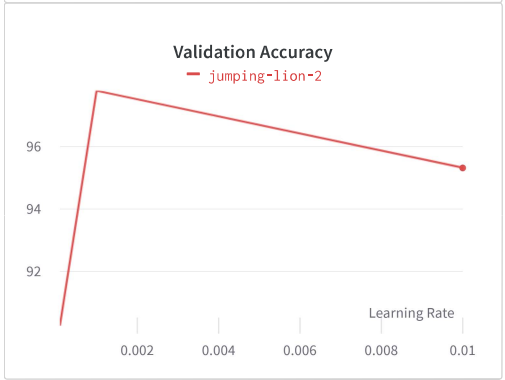

Validation Accuracy
jumping-lion-2
95
90
85
Step



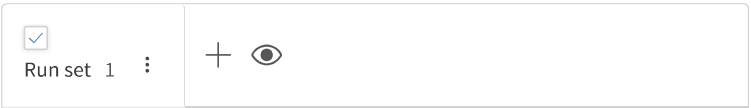








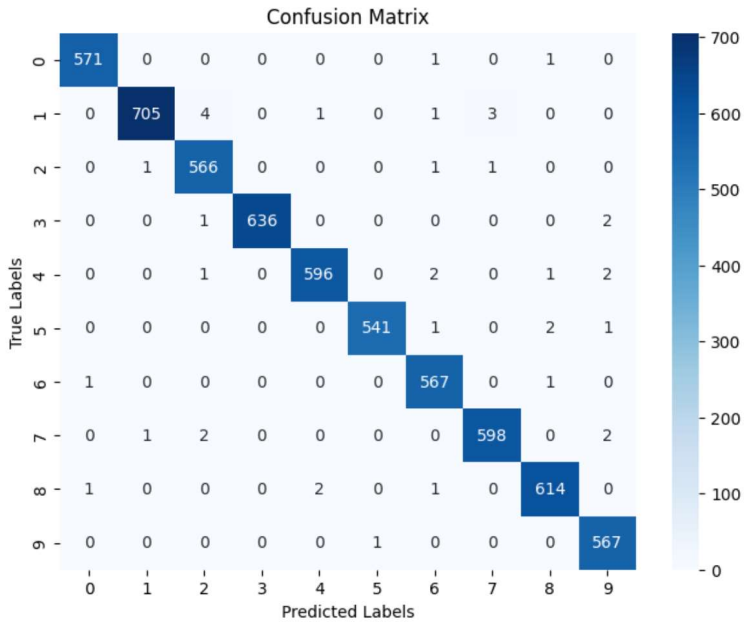
Import panel Add panel



So, the best model is:

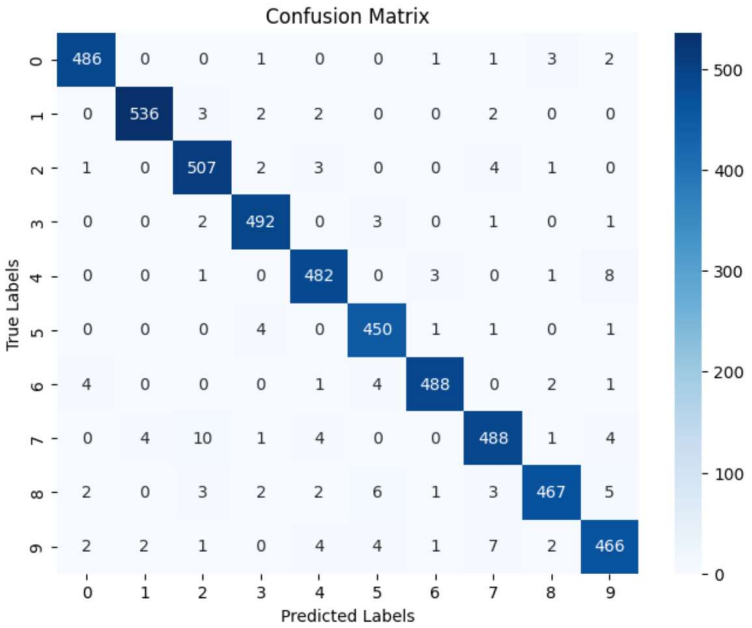
```
Batch Size      128
Dropout Rate    0.5
Epoch   10
Learning Rate   0.0001
Train Loss      0.16834
Validation Accuracy     96.13333
```

The confusion matrix looks like:

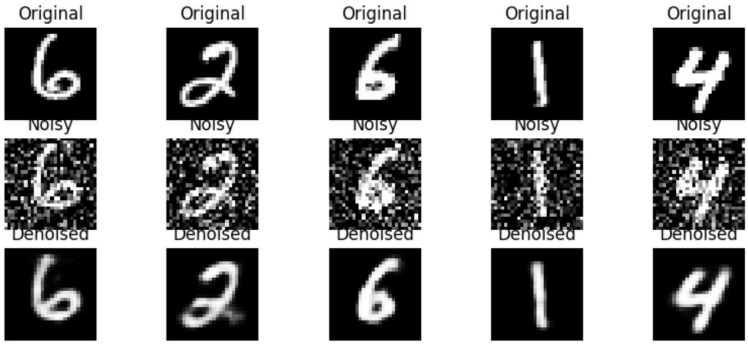


Then we use Autoencoders by adding noise to the mnist data and then training autoencoders on that and then testing it with the dataset provided.

The confusion matrix on the noisy data is:



The noisy data we generated and the data after using autoencoder:



The Accuracy for the same comes out to be 92.21666666666667.