Layer API - A Simple Neural Network

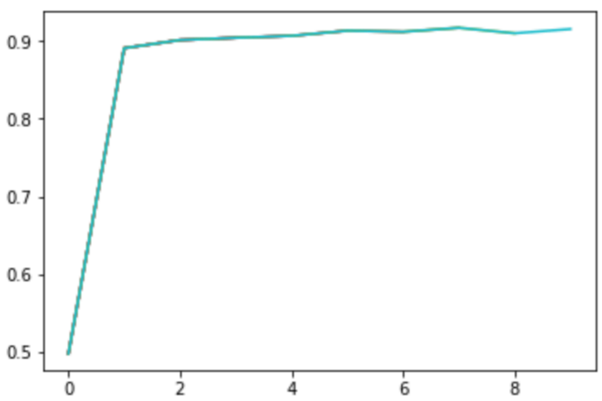
Programming Assignment 1-3 : Report

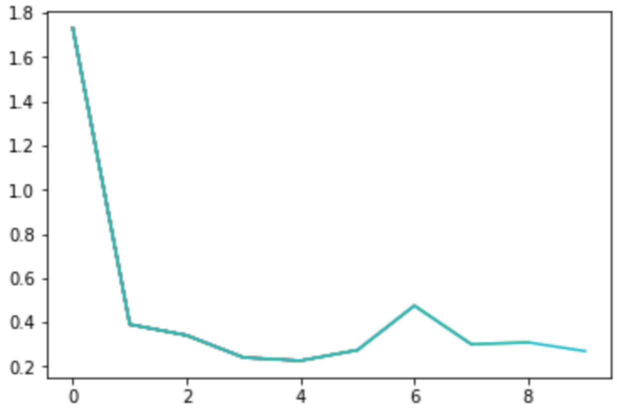
Submitted by : Group 3

1. Softmax is used in final layer, softmax_cross_entropy used for cross entropy, **GradientDescentOptimizer** used for optimizer

Number of layers	Activation in intermediate layer	Epochs	Accuracy	Learning rate	F score
1	NA	1000	91.8	0.5	90.75

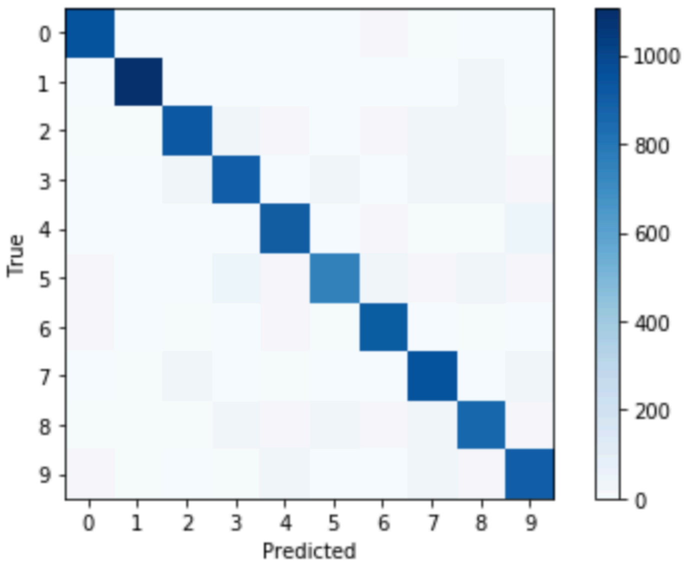
FIGURES :

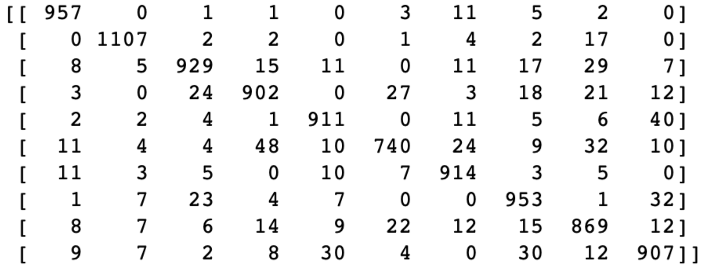




Accuracy Curve

Loss curve





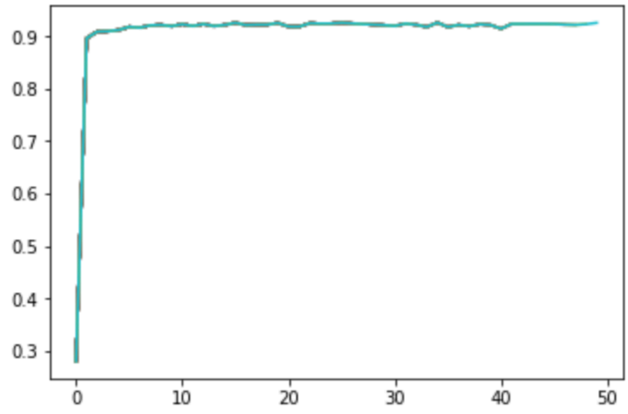
```
[[ 957    0    1    1    0    3   11    5    2    0]
 [   0 1107    2    2    0    1    4    2   17    0]
 [   8    5  929   15   11    0   11   17   29    7]
 [   3    0   24  902    0   27    3   18   21   12]
 [   2    2    4    1  911    0   11    5    6   40]
 [  11    4    4   48   10  740   24    9   32   10]
 [  11    3    5    0   10    7  914    3    5    0]
 [   1    7   23    4    7    0    0  953    1   32]
 [   8    7    6   14    9   22   12   15  869   12]
 [   9    7    2    8   30    4    0   30   12  907]]
```

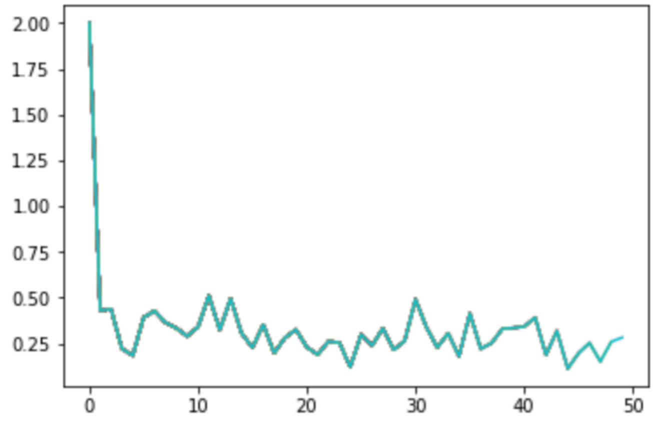
Confusion Matrix

2. Softmax is used in final layer, softmax_cross_entropy used for cross entropy, **AdamOptimizer** used for optimizer

Number of layers	Activation in intermediate layer	Epochs	Accuracy	Learning rate	F score
1	NA	5000	92.7	0.005	92.55

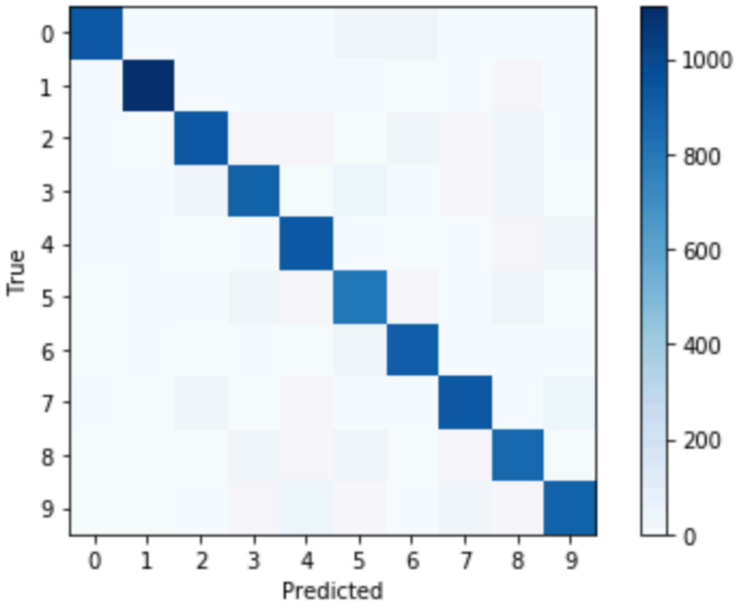
FIGURES





Accuracy Curve

Loss Curve



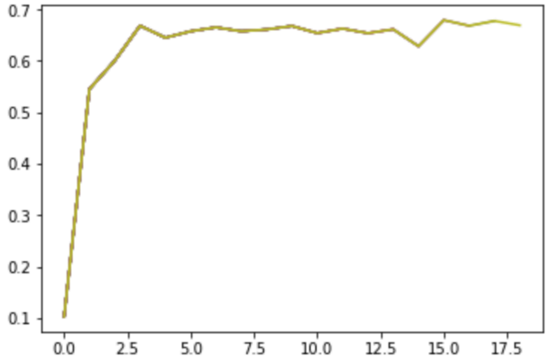
```
[[ 935    0    2    2    4   14   17    2    2    2]
 [   0 1112    2    2    0    2    5    2   10    0]
 [   2    7  929   13   11    7   15   11   33    4]
 [   2    0   19  899    5   44    2   10   22    7]
 [   1    1    5    2  934    0    8    2    9   20]
 [   7    2    3   16   11  806   13    4   25    5]
 [   5    3    7    0    8   22  909    1    3    0]
 [   1    5   22    5   13    1    0  932    4   45]
 [   6    8    6   17   10   37    8    9  865    8]
 [   6    5    1   11   49    9    0   21   10  897]]
```

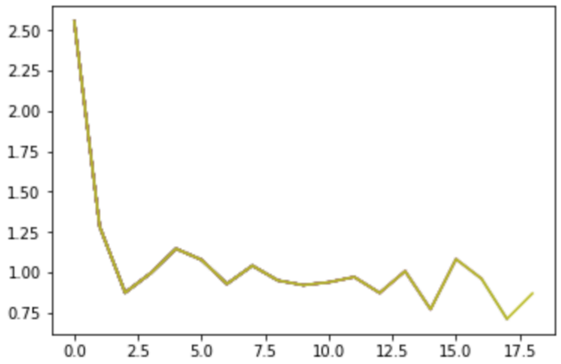
Confusion Matrix

3. Softmax is used in final layer, softmax_cross_entropy used for cross entropy, **AdamOptimizer** used for optimizer

Number of layers	Activation in intermediate layer	Epochs	Accuracy	Learning rate	F score
2	Sigmoid	15000	67.8	0.005	65.05

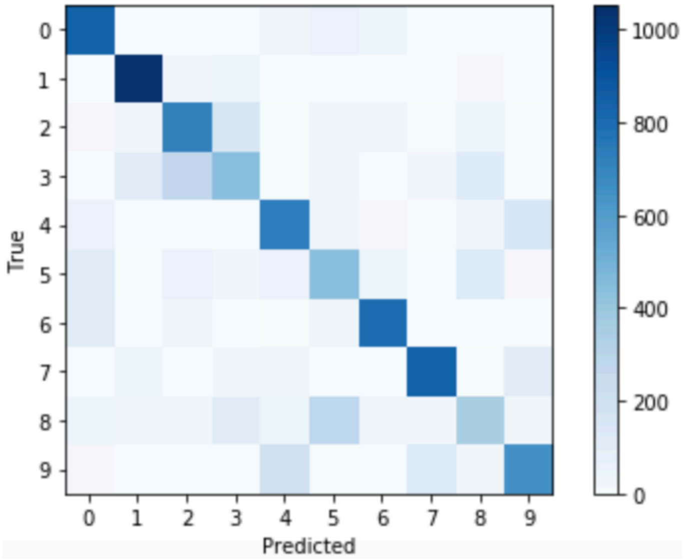
FIGURES





Accuracy curve

Loss Curve



```
[[ 844    0    1    1   34   54   44    0    1    1]
 [    0 1054   16   47    0    1    1    2   12    2]
 [    9   25  709  171    7   25   31    7   43    5]
 [    2  108  270  437    5   35    4   13  130    6]
 [   53    1    1    2  727   17    9    6   20  146]
 [   98    7   60   34   60  441   46    4  130   12]
 [  100    1   27    2    6   31  787    0    4    0]
 [    1   44    2   13   18    1    0  838    7  104]
 [   44   37   27  102   44  283   36   16  358   27]
 [    9    3    0    1  194    8    1  125   22  646]]
```

Confusion Matrix

Variations Tried:

1. Layers : 1/2/3/4
2. Number of neurons in hidden layer : 1000/512/500/300
3. Intermediate Layer activations : relu/sigmoid
4. Learning rate variations : 0.5/0.05/0.005/0.0005/0.8 etc
5. Batch size : 30/100/200
6. Epochs : 1000-20000

Inferences:

Loss decreases with training, but fluctuations exist (possibly due to taking small batches for training). Changes in number of neurons in hidden layer causes changes in accuracy, 512 neurons gave 40% while 500 neurons gave 68% accuracy. For certain models with larger number of layers accuracy drops to 11%, could be due to overfitting or lack of proper hyper parameter tuning. Best accuracy is seen for 1 layer models reaching 92% accuracy within 1000 iterations at a learning rate of 0.5. Loss and accuracy seem to be inversely related, which is intuitive.