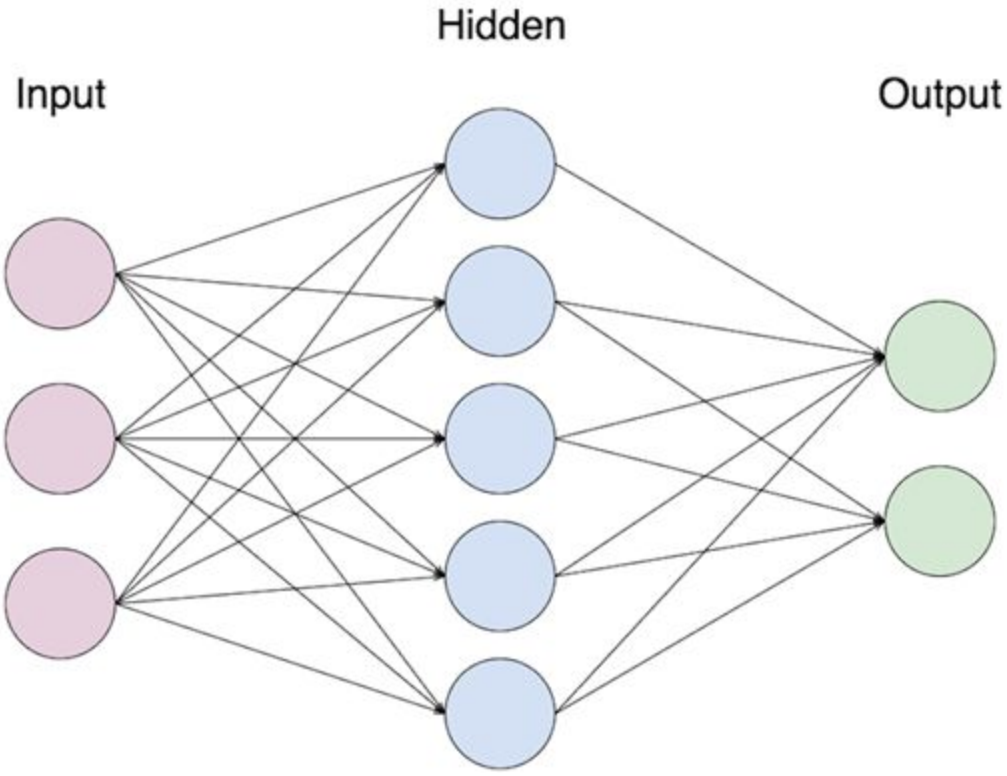
Deep Learning - Project 3

Submitted by - Paritosh Goel, Muhammad Tahir



Set of Experiments

No of Layers	**No of nodes in Each layer**	**Activation Function used**	**Batch Size and Epochs**	**Accuracy**	**Other parameters**
1	125	Relu, Softmax	Batch Size - 2000 , Epochs - 50	Test accuracy: 0.81538	kernel_initializer='he_normal') optimizer='sgd' loss='categorical_crossentropy'
2	**125, 125**	Relu, Relu, Softmax	Batch Size - 2000 , Epochs - 50	Test accuracy: 0.3846	kernel_initializer='he_normal') optimizer='sgd' loss='categorical_crossentropy'
1	125	Relu, Softmax	Batch Size - 2000 , Epochs - 50	Test accuracy: 0.871	**kernel_initializer='random_uniform''**)
1	125	Relu, Softmax	**Batch Size - 500** , Epochs - 50	Test accuracy: 0.88	**kernel_initializer='random_uniform'')**
1	125	Relu,Softmax	**Batch Size - 500** , **Epochs - 100**	Test accuracy: 0.886	kernel_initializer='random_uniform''
1	125	Tanh, Softmax	Batch Size - 500 , Epochs - 100	Test accuracy: 0.840	kernel_initializer='random_uniform''
	125	Tanh,	Batch Size -	**Test**	optimizer='R

1		Softmax	500 , Epochs - 10	**accuracy: 0.931**	MSprop', loss='categorical_crossentropy'
1	125	Tanh, Softmax	Batch Size - 500 , Epochs - 100	Test accuracy: 0.925	**Loss = 'mean_squared_error'**

Procedure Description:
Train on 3900 samples, validate on 975 samples, testing on 1625 samples

Most accurate Accuracy on Test Data - 93%

Epoch 100/100
3900/3900 [==============================] - 0s 22us/step - loss: 0.0024 - acc: 0.9887 - val_loss: 0.0115 - val_acc: 0.9251

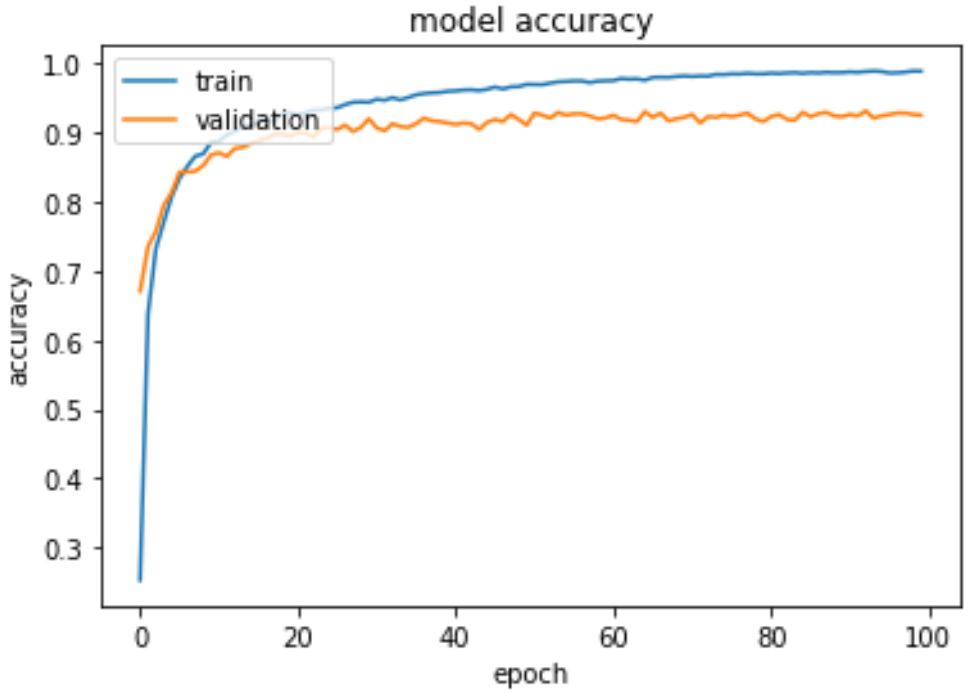
No of Layers - 1
Size of layer - 125 nodes
Activation Function used - Tanh for the first layer and Softmax for the last layer
Batch Size - 500
Epochs - 10
Optimizer - RMSprop
Loss - categorical_crossentropy
Weight Initialization Scheme - random_uniform
Activation functions - Tanh for the first layer, Softmax for the output layer

It was seen that changing the optimizer to RMSprop increased the accuracy to this value.
Epoch size - 10
Images wrongly classified - 6 (instead of 6, classified to 4)

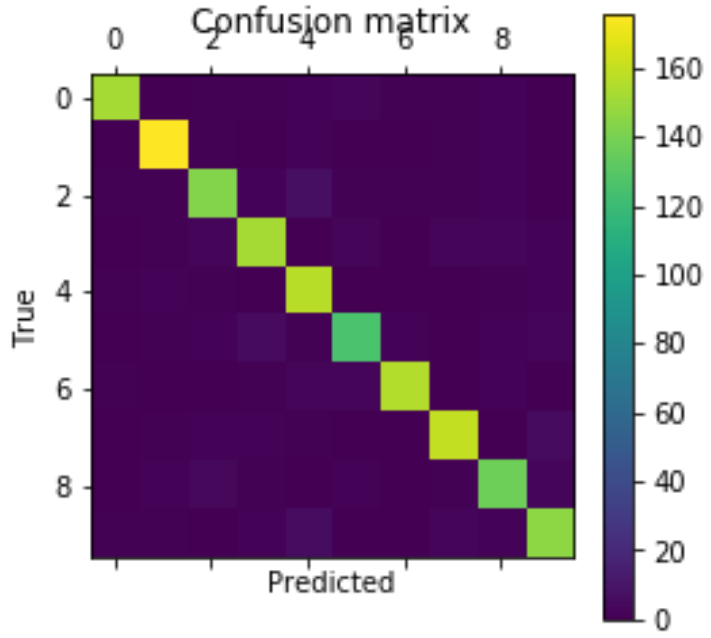
```
model = Sequential() # declare model
model.add(Dense(125, input_shape=(28*28,), kernel_initializer='random_uniform')) # first layer
model.add(Activation('tanh'))
model.compile(optimizer='RMSprop',
```

```
        loss='mean_squared_error',
        metrics=['accuracy'])
history = model.fit(X_train, Y_train,validation_data = (X_val, Y_val),epochs=10,batch_size=500)
```

Plot

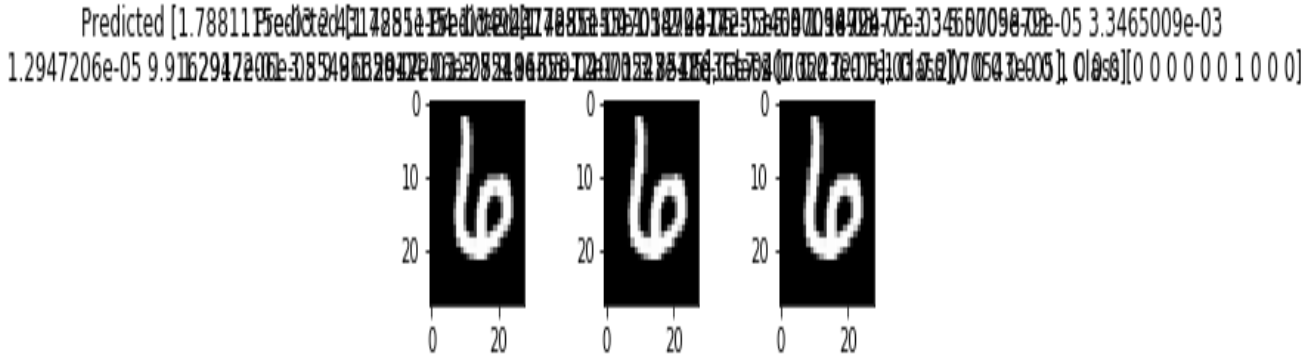


Confusion Matrix -



Model Performance - **Accuracy - 93%**

Visualization



6 is wrongly classified as 4