CSCI 5922 – Neural Nets and Deep Learning
Shreyas Kapoor | Lab Assignment 1

Question 1 – Neural Network Hyperparameters

The dataset that was used for this question was the **Olivetti faces** dataset from the **sklearn** library. It contains set of images taken between 1992-1994 at AT&T Labs Cambridge. The dataset contains 10 different images from 40 different subjects taken on various time, with different facial expressions and under different lightings. The common thing is that all the images were taken in dark homogenous background with everyone in upright frontal position. The dataset contains of 400 samples and can be classified into 40 classes. The target of the dataset is to label integer 0-39 to each image indicating the identity of the person.
To train the dataset, Neural Network was used. In that, **MLPClassifier** was chosen to train the neural network. **Number of iterations was kept being 100 and number of neurons per hidden layer was 10**. The combination of 9 neural networks trained were as follows

1. No of hidden layer = 2; Activation Function = relu
2. No of hidden layer = 2; Activation Function = logistic
3. No of hidden layer = 2; Activation Function = tanh
4. No of hidden layer = 10; Activation Function = relu
5. No of hidden layer = 10; Activation Function = logistic
6. No of hidden layer = 10; Activation Function = tanh
7. No of hidden layer = 25; Activation Function = relu
8. No of hidden layer = 25; Activation Function = logistic
9. No of hidden layer = 25; Activation Function = tanh

Results for each model

TABLE 1: Accuracy

	Activation Function		
No of Hidden Layers	relu	logistic	tanh
2	0.083	0.0083	0.075
10	0.608	0.05	0.567
25	0.858	0.166	0.808

Confusion Matrix

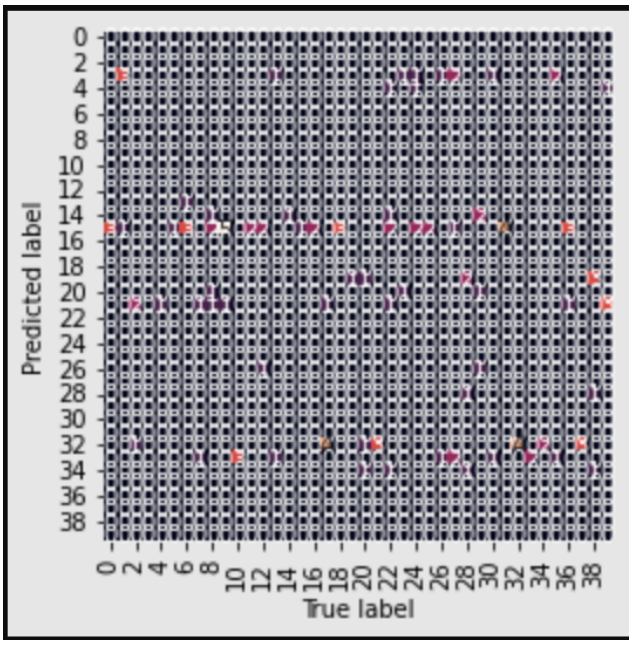

Fig-1 : Model 1

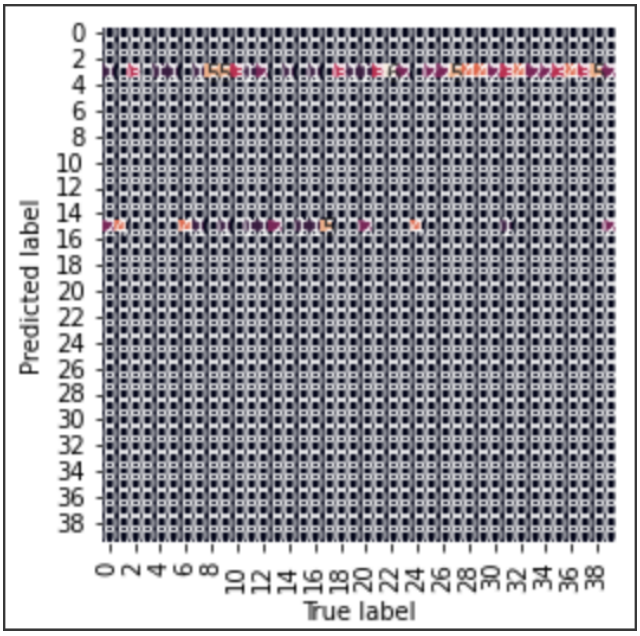

Fig-2: Model 2

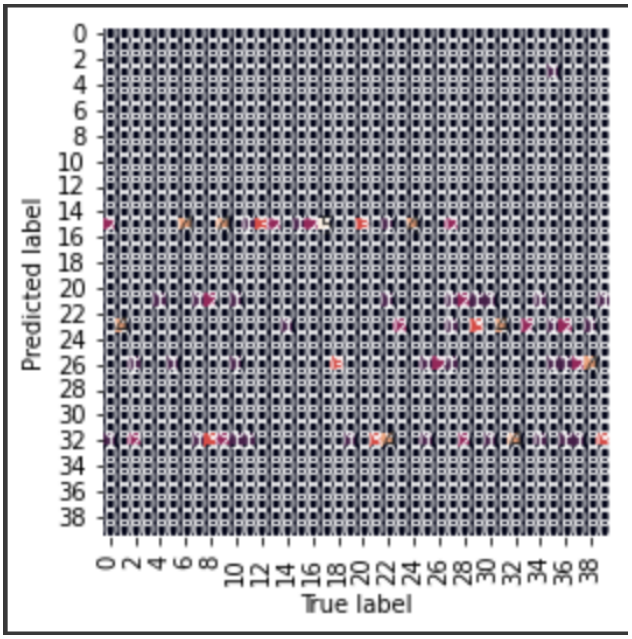

Fig 3: Model 3

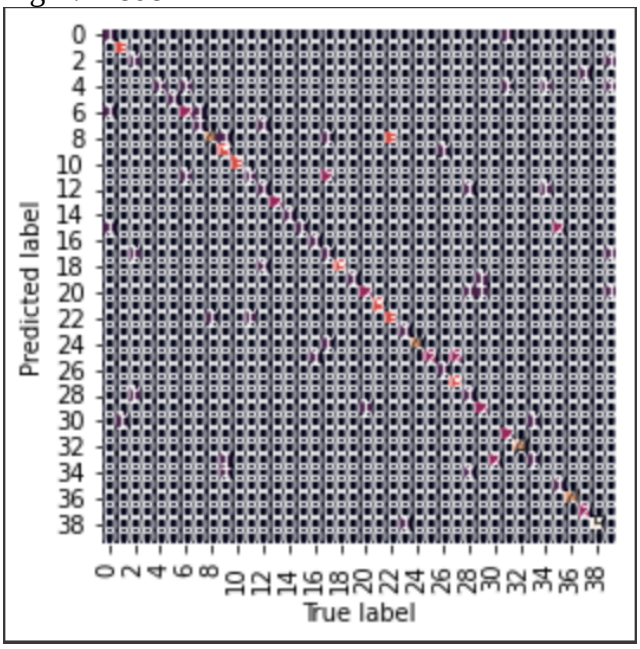

Fig 4: Model 4

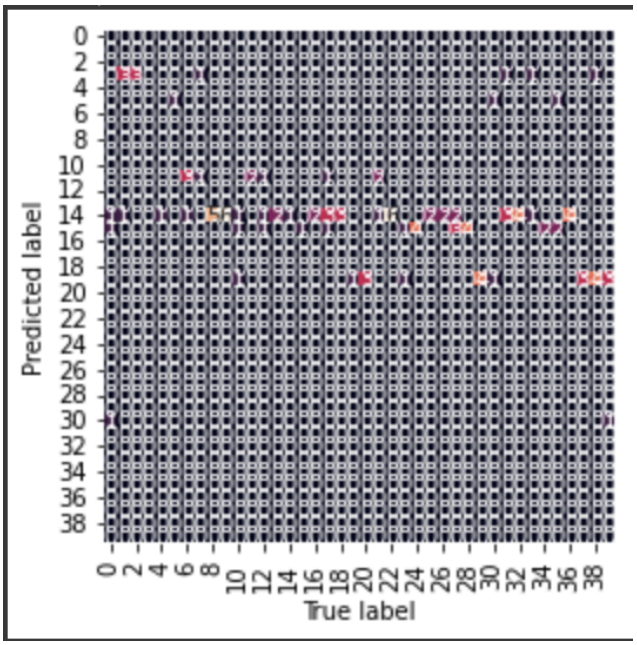

Fig 5: Model 5

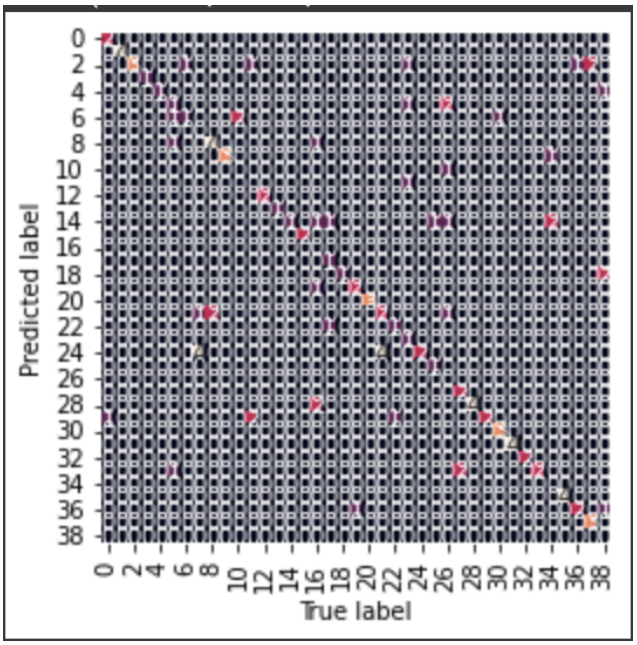

Fig 6: Model 6

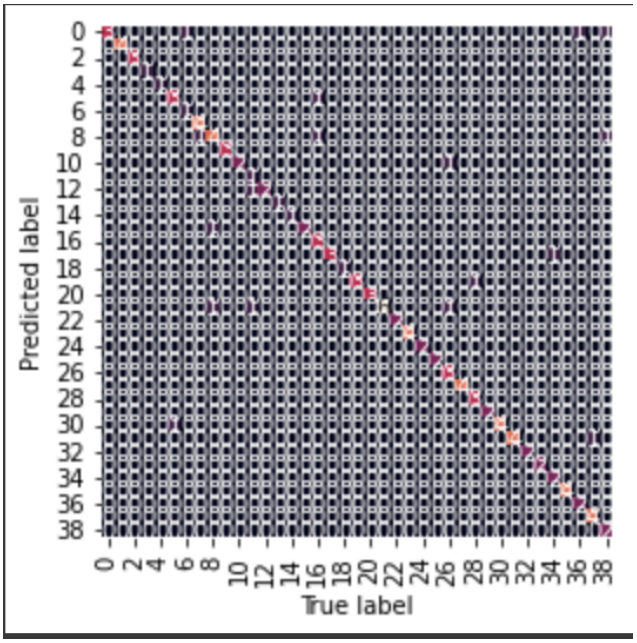

Fig 7: Model 7

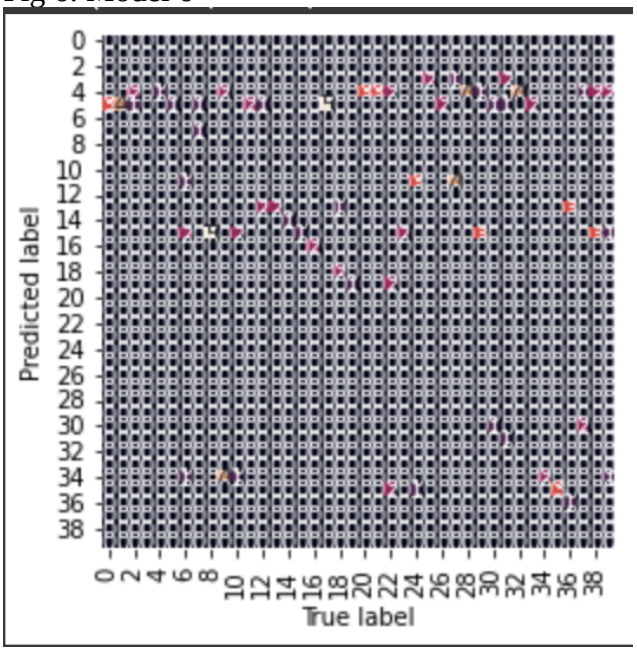

Fig 8: Model 8

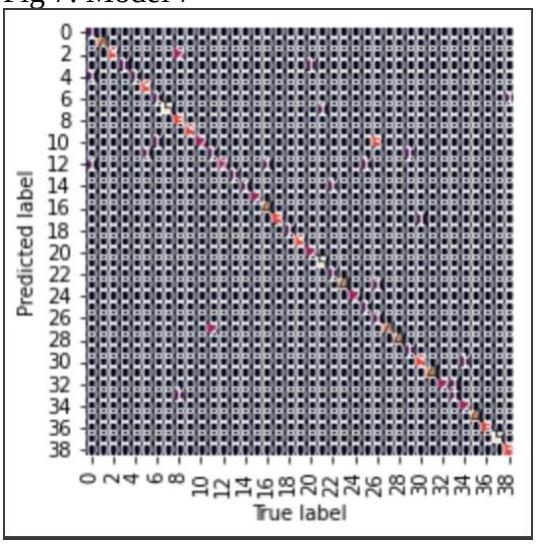

Fig 9: Model 9

Analysis on trends

The first trend is the **number of hidden layers used** in the neural network. From the accuracy table, it's obvious that **as we increase the number of hidden layers, the accuracy increases**. This happens because when we increase the hidden layers, we are giving more time and data to the neural network to train on the data. Every layer generates a loss of its own and this loss helps the model to learn and change the parameters. The more data we give to the model, higher are its chances to perform better.

The second trend is the **activation function** used in the model. Of all the activation functions used, **RELU performed the best** irrespective of the number of hidden layers used. This was expected as RELU has an advantage of non-saturation of its gradient and hence accelerates the convergence of stochastic gradient descent. Also, RELU has been proven to be a default choice to train neural networks as they both work best.

As we **increased the number of layers, the precision increased for models with RELU and tanh activation function**. But for **logistic** activation function, **the precision did not increase** even if we increased the number of hidden layers. Also, the accuracy with logistic activation function is very low compared to other function. This suggests that logistic is not a good choice to train this dataset.

Question 2 – Impact of training duration and training data

The dataset used for this problem is same as the previous dataset. Here also, similar to previous part, I used MLPClassifier from sklearn library to train the models. Number of iterations was set to 150. RELU was used as the activation function. The number of hidden layers were 300 and each hidden layer had 40 neurons.

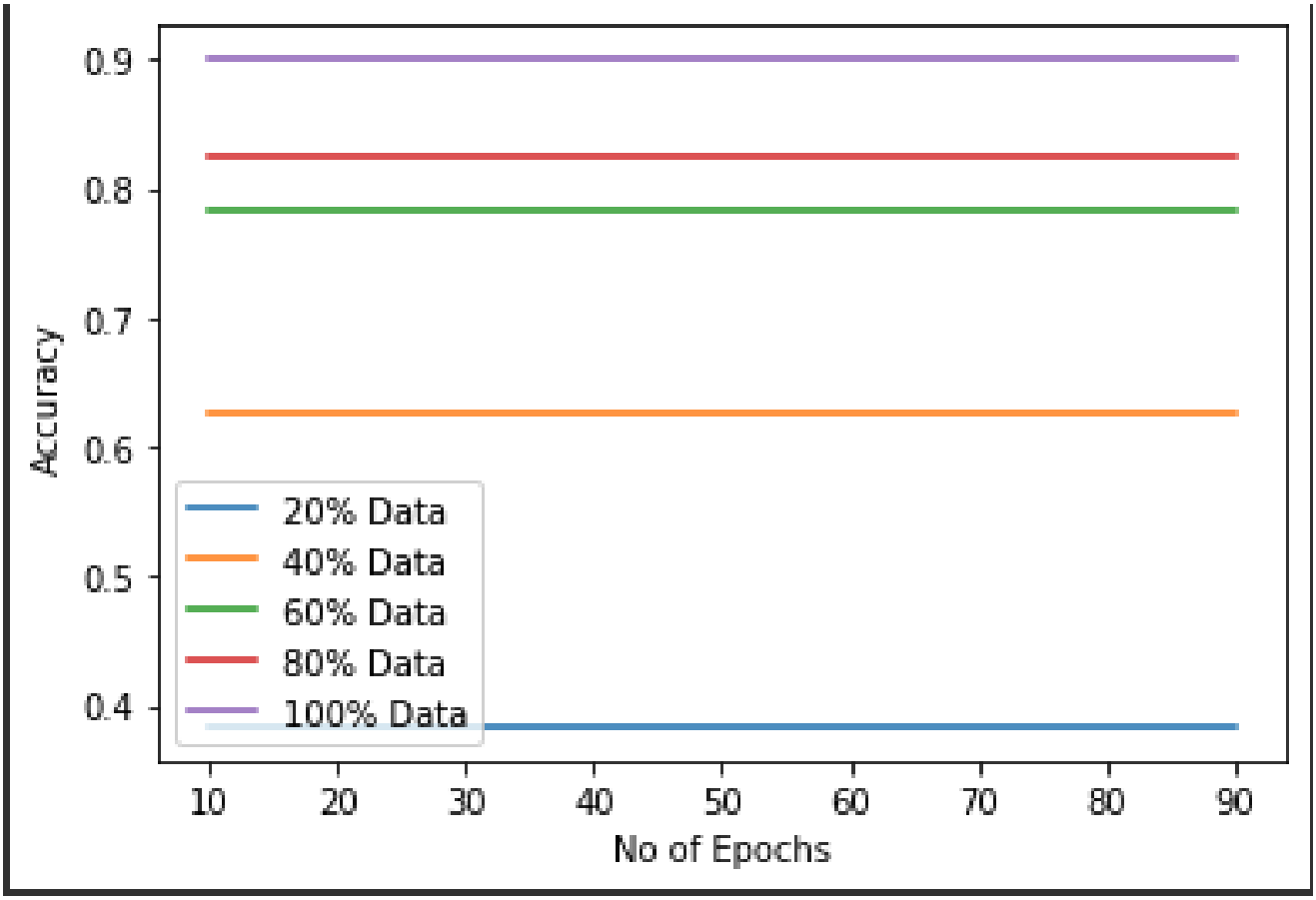

Fig 10 : Plot of 5 models' accuracy compared to number of epochs.

Analysis of trends

As we increase the size of the training data, the accuracy increases. With 20% of the training data, the accuracy was just about 0.48. But as the size went to 100%, the accuracy increased to 0.94. This was expected as we are giving more data to the model to get trained on.

For the dataset I chose and with the parameters I had set, there was **no effect of the number of epochs on the accuracy of the model**. The model attained its best accuracy on its first epoch. Although surprising, I believe this happened because there were not enough samples in the training dataset and with the combination of layers, neurons and iterations, the model got trained in its first iteration.