Implementation of Neural Networks:

We have created a class **NeuralNetwork** which is initialized with number of hidden nodes and mode of operation (Best Algorithm has set parameters). On initialization, the weights and biases are randomly generated. Numpy has been used for matrix generation, matrix manipulation (multiplication, simple division, simple multiplication, computing transpose) and computing the activation functions. The method **train_nerualnet** splits input into individual images, each of which is fed to the 3-Layer Neural Network through the method **back_propagation**. This method executes the Forward Propagation and the Backward Propagation for each image. The changes in the weights and biases for each layer are then added for each mini-batch (Stochastic Gradient Descent is used). After each mini-batch, the weights and biases for the entire Neural Network are updated. In the end, the classification accuracy and confusion matrix are computed.

1. Problem 3: Best Algorithm - Neural Networks

---

a. Implementation Overview:

For the best algorithm, we have selected the Neural Network algorithm with a specific set of parameters and a slightly different implementation. For this version of the Neural Network we have used the following:

- ➔ Stochastic Gradient Descent Used
- ➔ Cost Function: Cross Entropy Function. Implemented in order to avoid the neuron's learning slowdown which occurs in case of the Root Mean Squared Error Cost Function.
- ➔ Weight Initialization using normalized Gaussians: Implemented to ensure learning slowdown for the neurons
- ➔ Weight Decay using L2 Regularization: Implemented to avoid Overfitting

Parameters Set To:

Epoch: 10 | ETA:0.25 | SGD Batch Size: 25 | Regularization Parameter: 0.0005
Hidden Layer Nodes Required: 125
Accuracy Achieved: 70-72.32% | Time Required: 145.12 sec

b. Selection of Parameters, Accuracy & Time Required for execution:

Epochs	Hidden Layer Nodes	Learning Rate (eta)	SGD Batch Size	Regularization Parameter	Accuracy (%)	Time (sec)
10	125	0.25	25	0.0005	72.32	145.12

10	175	0.25	25	0.0005	71.58	188.5
30	130	0.30	30	0.0003	71.04	403.74
3	100	0.75	75	0.0007	71.15	36.02
1	125	0.25	25	0.0005	65.53	15.56

c. Size of Training Data Set:

Size of Training Set	**Accuracy (%)**	**Time Required (sec)**
25% of Training Data	70.09	37.30
50% of Training Data	70.51	67.17
75% of Training Data	70.73	88.73

d. Misclassified Examples/Patterns Observed:

➔ GrayScale Images were often misclassified. This may be due to the inability of the neural network to learn patterns through color distribution

➔ Images which comprised of similar shades of a dominant color were misclassified

➔ Images having complex patterns like animal faces, complex landscape features seemed to be misclassified

e. Assumptions/Challenges Faced:

Despite various iterations of parameter tuning, we were not able to observe any significant jump in accuracy. Also, we employed cross entropy, regularization, Gaussian Distributed weight initialization and were still not able to increase the accuracy significantly. It would be very interesting to understand the issues in this regard with our implementation and the maximum achievable accuracy.

2. Problem 2: Neural Networks

---

a. Implementation Overview:

For Neural Networks implementation, we have used the following elements:

- ➔ Stochastic Gradient Descent Used
- ➔ Cost Function: Root Mean Squared Error Used
- ➔ Activation Function: tanh(z). Other parameters being constant, we observed better results for tanh(z) activation function than the sigmoid activation function
- ➔ No special weight initialization or regularization used

Parameters Set To:

Epoch: 5 | ETA: 0.2 | SGD Batch Size: 30

Hidden Layer Nodes Required: 90

Accuracy Achieved: 70.41% - 71.47% | Time: 51-55 secs

b. Selection of Parameters, Accuracy & Time Required for execution:

Epochs	Hidden Layer Nodes	Learning Rate (eta)	SGD Batch Size	Accuracy (%)	Time (sec)
30	175	0.3	30	70.83	534.4
10	100	0.1	30	70.62	117.4
1	75	0.7	50	70.41	11.57
5	125	0.2	30	69.98	72.42
1	100	0.01	30	62.88	25.35

c. Size of Training Data Set:

Parameters Set To: Epoch: 5 | ETA: 0.2 | SGD Batch Size: 30 | Hidden Layer Nodes: 90

Size of Training Set	Accuracy(%)	Time Required (sec)
25% of Training Data	62.35	15.90
50% of Training Data	68.50	29.95
75% of Training Data	70.30	35.06

d. Misclassified Examples:

The patterns of misclassification observed for Neural Networks are similar to those observed in Problem 3 (mentioned above)

e. Assumptions/Challenges Faced

Here too, we weren't able to observe any significant increase in accuracy by tweaking the parameters. While the drop in the accuracy after tweaking the parameters was quite significant, we got felt that the error minima for the our implementation occurs when the accuracy is in the range 70-73%..

3. **K - Nearest Neighbor Classification:**

---

a. Implementation Overview:
   - ➔ Implemented using three functions (knn(), traindata(), findKnn())
   - ➔ For computing similarity, we have used simple distance function. Here, we consider all 192 features and compute the difference between individual pixels of corresponding colors. We sum over this difference with all training examples for each test example. Thus, each test example will produce a distance array of dimension (len(number of training images)*1)
   - ➔ We then select the 'k' most similar neighbors to the test image.
   - ➔ The maximum occurring label among these k similar training images is taken as the final label for the test image

b. Selection of Parameters, Accuracy & Time Required for execution:

Value of k	Accuracy (%)	Time Required (sec)
5	69.45	300.38

9	70.41	306.54
21	70.51	358.99
29	70.94	363.819
45	70.62	350.97

c. Size of Training Data Set:

Size of Training Set	**Accuracy (%)**	**Time Required (sec)**
25% of Training Data	70.09	78.26
50% of Training Data	70.20	154.93
75% of Training Data	70.20	229.81

d. Misclassified Examples:

➔ GrayScale Images were often misclassified. This may be due to the inability of the neural network to learn patterns through color distribution

➔ Images which comprised of similar shades of a dominant color were misclassified

➔ Images having complex patterns like animal faces, complex landscape features seemed to be misclassified

e. Assumptions/Challenges Faced

In earlier iterations, we used Weighted Distance Formula to provide preference to most similar neighbors for a test image. However, we were not able to improve accuracy significantly.

Conclusion & Recommendation to Client (with Parameters):

We will recommend the Best Algorithm implemented for this Assignment to the client. Following are the parameters and accuracy/time obtained on the same:

Epoch: 10 | ETA:0.25 | SGD Batch Size: 25 | Regularization Parameter: 0.0005
Hidden Layer Nodes: 125 | Accuracy Achieved: 70-72.32% | Time Required: 145.12 sec