Assignment 4:

Sudarshan Devkota

Problem 1:

1. Fisher Linear Discriminant Function:

For the classification of CIFAR-10 dataset, we implement the Fisher method for finding a linear discriminant function that maximally separates the class means while minimizing the in-class variance. The separation between the classes is increased by maximizing the distance of each class center from the overall mean of the entire data. The classification process that was implemented for the given dataset is described below.

I. Since, fisher LDF can be applied directly to images, instead of extracting features first, we treat each image x_k from class i as feature vectors x_k^i . For 32*32 images, the size of the feature vectors is 784 x 1.

II. Compute mean vectors for each class and overall mean of data.

$$\boldsymbol{\mu}_i = \frac{1}{N}\sum_{k=1}^{N} \boldsymbol{x}_k^i$$

III. Compute between class scatter matrix ***B***

$$\boldsymbol{B} = \sum_{i=1}^{M}(\boldsymbol{\mu}_i - \boldsymbol{\mu})(\boldsymbol{\mu}_i - \boldsymbol{\mu})^T$$

IV. Compute covariance matrix for each class, and their sum ***A***.

$$\boldsymbol{\Sigma}_i = \frac{1}{N}\sum_{k=1}^{N}(\boldsymbol{x}_k^i - \boldsymbol{\mu}_i)(\boldsymbol{x}_k^i - \boldsymbol{\mu}_i)^T$$

$$\boldsymbol{A} = \sum_{i=1}^{M} \boldsymbol{\Sigma}_i$$

V. Compute $A^{-1}B$ and obtain its eigenvectors and eigenvalues.

VI. Select the dominant eigenvectors corresponding to the largest eigenvalues as the LDF vectors. Eigenvector h_1 corresponds to λ_1 , eigenvector h_2 corresponds to λ_2 and so on.

VII. Using the fisher LDFs, the 10 classes are easy to separate with the defined matrix

$$H = [h_1,\ h_2,\ \dots h_9]$$

VIII. Transform the HCD feature vectors x_k^i into the Fisher LDF space

$$f_k^i = H^T x_k^i$$

IX. Transform the class means and covariance matrices to the Fisher LDF space as well

$$m_i = H^T \mu_i \ and\ S_i = H^T \Sigma_i H$$

X. Now the data is classified by assigning it to the class with the smallest Mahalanobis distance. For each test vector *k,* compute

$$d_i = (f_k^i - m_i)^T S_i^{-1} (f_k^i - m_i)\ for\ all\ i\ classes$$

Then, assign the test vector the label of the class for which d_i is the smallest.

2. Results:

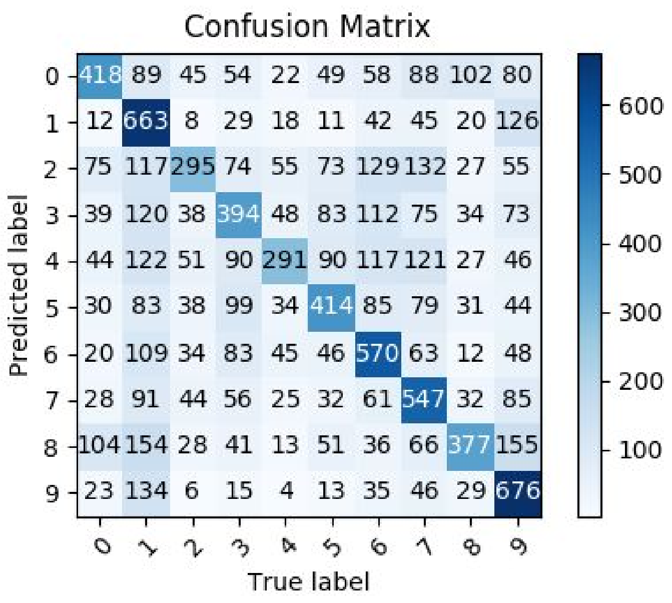
We obtain the following results for the classification of objects in the CIFAR-10 dataset using Fisher LDFs. CIFAR-10 contains 50,000 training and 10,000 test images, but this implementation only uses a subset of the entire dataset: 10000 training images and 1000 testing images. Table 2 and 3 show the final statistics of the classifier in terms of accuracy and error rate for each class in training set and test set.

Table 2: Accuracy and error rate for each class in training set:

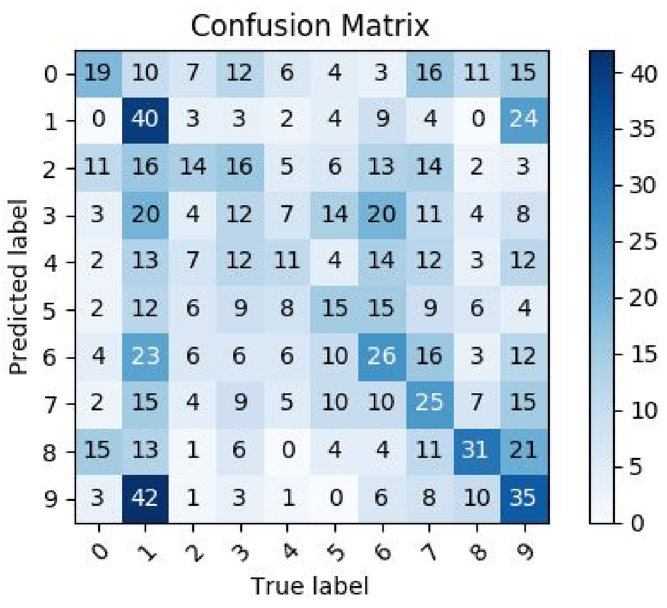
Class labels	0	1	2	3	4	5	6	7	8	9
Accuracy	0.527	0.394	0.503	0.421	0.524	0.48	0.458	0.433	0.546	0.487
Error rate	0.473	0.606	0.497	0.579	0.476	0.52	0.542	0.567	0.454	0.513
Overall accuracy: 0.464	Overall error rate: 0.536									

Table 3: Accuracy and error rate for each class in test set:

Class labels	0	1	2	3	4	5	6	7	8	9
Accuracy	0.311	0.196	0.264	0.136	0.216	0.211	0.217	0.198	0.403	0.235
Error rate	0.689	0.804	0.736	0.864	0.784	0.789	0.783	0.802	0.597	0.765
Overall accuracy: 0.228	Overall error rate: 0.772									



a) Training set



b) Testing set

Fig 1: Confusion matrix for a) training set and b) test set.

The figures in 1)a and 1)b shows the confusion matrix for training set and test set. Using Fisher LDFs for classification, overall classification accuracy for the training set is obtained to be 46.4% while the classification accuracy for the test set is 22.8%. For simplicity, the 10 object names in the CIFAR-10 dataset have been encoded as integers from 0 through 10. The set of objects {airplane, automobile, bird, cat, deer, dog, frog, horse, ship, truck} are denoted with the set of integers {0,1,2,3,4,5,6,7,8,9}.

Problem 2

1. Convolutional Neural Networks (CNN):

Here, we implement a Convolutional Neural Network for the classification of objects in CIFAR 10 dataset. Similar to ordinary neural networks, CNNs are made up of neurons that have learnable weights and biases. Each neuron receives some inputs, performs a dot product and optionally follows it with a non-linearity. The whole network expresses a single differentiable score function: from the raw image pixels on one end to class scores at the other. And they have a loss function (e.g. /Softmax) on the last (fully-connected) layer.

Architecture Overview

The architecture of the implemented CNN network is described below. For the implementation of the given architecture, Keras was used with TensorFlow backend.

1	Input Image	32 x 32 x 3 images with 'zero center' normalization
2	Convolution Layer	32 5 x 5 convolutions with stride = 1 and padding = 2
3	Activation Function	ReLU
4	Pooling	2 x 2 max-pooling with stride = 2
5	Convolution Layer	32 5 x 5 convolutions with stride = 1
6	Activation Function	ReLU
7	Pooling	2 x 2 max pooling with stride = 2
8	Convolution Layer	64 5 x 5 convolutions with stride = 1
9	Activation Function	ReLU
10	Pooling	2 x 2 max pooling with stride = 2
11	Fully Connected Layer	64
12	Activation Function	ReLU
13	Fully Connected Layer	10
14	Activation Function	Softmax
15	Loss Function	Cross-Entropy

2. Results:

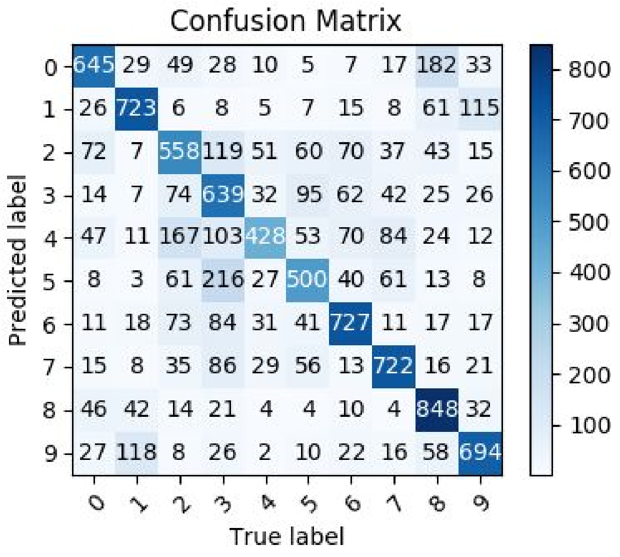
The following results are obtained for the classification of objects in the CIFAR-10 dataset using Convolutional Neural Networks. CIFAR-10 contains 50,000 training and 10,000 test images, but this implementation only uses a subset of the entire dataset: 10000 training images and 1000 testing images. Before feeding the data through the network, the dataset is normalized with zero-center normalization, where the mean of the entire dataset is subtracted from each image. Table 2 and 3 show the final statistics of the classifier in terms of accuracy and error rate for each class in training set and test set.

Table 2: Accuracy and error rate for each class in training set:

Class labels	0	1	2	3	4	5	6	7	8	9
Accuracy	0.708	0.748	0.534	0.48	0.691	0.602	0.702	0.721	0.659	0.713
Error rate	0.292	0.252	0.466	0.52	0.309	0.398	0.298	0.279	0.341	0.287
Overall accuracy: 0.648	Overall error rate: 0.352									

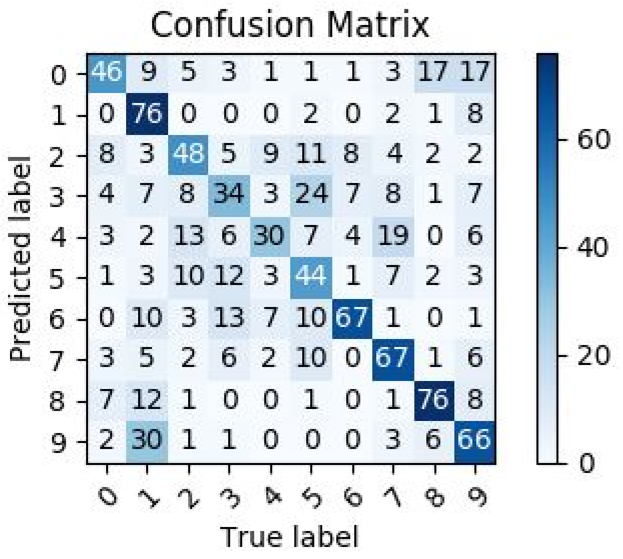
Table 3: Accuracy and error rate for each class in test set:

Class labels	0	1	2	3	4	5	6	7	8	9
Accuracy	0.622	0.484	0.527	0.425	0.545	0.4	0.761	0.583	0.717	0.532
Error rate	0.378	0.516	0.473	0.575	0.455	0.6	0.239	0.417	0.283	0.468
Overall accuracy: 0.554	Overall error rate: 0.446									



	0	1	2	3	4	5	6	7	8	9
0	645	29	49	28	10	5	7	17	182	33
1	26	723	6	8	5	7	15	8	61	115
2	72	7	558	119	51	60	70	37	43	15
3	14	7	74	639	32	95	62	42	25	26
4	47	11	167	103	428	53	70	84	24	12
5	8	3	61	216	27	500	40	61	13	8
6	11	18	73	84	31	41	727	11	17	17
7	15	8	35	86	29	56	13	722	16	21
8	46	42	14	21	4	4	10	4	848	32
9	27	118	8	26	2	10	22	16	58	694

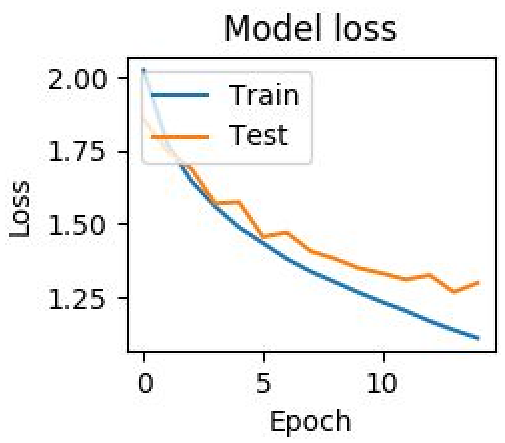
a) Training set



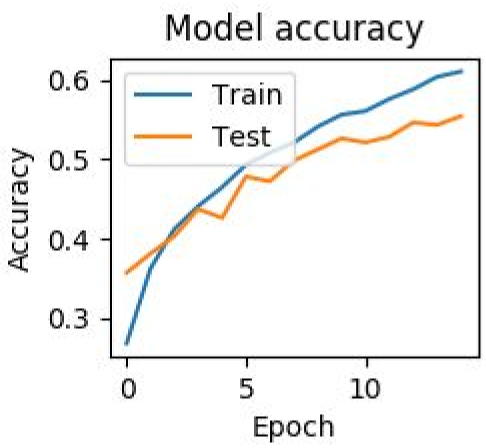
	0	1	2	3	4	5	6	7	8	9
0	46	9	5	3	1	1	1	3	17	17
1	0	76	0	0	0	2	0	2	1	8
2	8	3	48	5	9	11	8	4	2	2
3	4	7	8	34	3	24	7	8	1	7
4	3	2	13	6	30	7	4	19	0	6
5	1	3	10	12	3	44	1	7	2	3
6	0	10	3	13	7	10	67	1	0	1
7	3	5	2	6	2	10	0	67	1	6
8	7	12	1	0	0	1	0	1	76	8
9	2	30	1	1	0	0	0	3	6	66

b) Testing set

Fig 2: Confusion matrix for a) training set and b) test set.



a)



b)

Fig 3: a) Loss vs number of epochs, and b) Accuracy vs number of epochs for both training set and test set.

Figure 3 shows a plot of loss and accuracy against the number of epochs for training and test set. After the 15th epoch, while the training set accuracy was still increasing, it was seen that the testing loss displayed negligible change and accuracy for the test set stayed almost constant. So, to prevent overfitting the model, the training was stopped after the 15th epoch, where the testing accuracy is 55.4% and the training accuracy is 64.8%.

Even with a small subset of the entire CIFAR-10 dataset, CNN was seen to outperform Fisher LDF for the classification task. To make our CNN network more robust and accurate, we can further customize our network by integrating techniques like Batch-Normalization and Dropout.

Instructions on how to run program

First, the following dependencies need to be installed to be able to run the program:

```
$ sudo apt-get install python3-pip
$ pip3 install numpy
$ pip3 install matplotlib
$ pip3 install tensorflow
$ pip3 install keras
```

Unzip the 4808133.zip file and from inside the code/ directory run the following commands

```
$ python3 fisher_ldf.py
$ python3 cnn.py
```

References:

[1] Class Notes - Abhijeet Mahalanobis
[2] https://www.cs.toronto.edu/~kriz/cifar.html
[3] Keras Documentation: https://keras.io/