Assignment 2
Sudip Prasai

I. **Classification of feature points using Harris Corner Detection**

Harris Corner Detector is a corner detection operator used to extract corners and infer features of an image. A corner can be interpreted as the locations in images where a slight shift in the location will lead to a large change in intensity in both horizontal (X) and vertical (Y) axes. We compute the feature points using Harris Corner Detection algorithm and then use classifier to classify these points assuming that the feature vector have gaussian distribution. The steps involved in the algorithm are explained below:

1.The Harris Corner Detector algorithm is used to get the feature vector and class as follows:

- Given an Image *I(x, y)*, apply gaussian filter *g*(*x, y*) to smooth the image *I*.

$$g(x,y) = \frac{1}{2\pi\sigma^2} \cdot e^{-\frac{x^2+y^2}{2\sigma^2}}$$

  where x is the distance from the origin in the horizontal axis, *y* is the distance from the origin in the vertical axis, and σ is the standard deviation of the Gaussian distribution.

- And then, convolve the Image *I with g*(*x, y*) to get gaussian image I_g.
  - $I_g = g(x, y) * I$

- Compute x and y derivative of image using sobel operator g_x , g_y
  - $I_x = g_x * I_g$
  - $I_y = g_y * I_g$

- Find product of derivatives at every pixel
  - $I_{xx} = I_x . I_x$
  - $I_{yy} = I_y . I_y$
  - $I_{xy} = I_x . I_y$

- Convolve these with the window function *w*(*x*, y). *w*(*x*, y) is just a smoothing function with all elements as 1.
  - $S_{xx} = w * I_{xx}$
  - $S_{yy} = w * I_{yy}$
  - $S_{xy} = w * I_{xy}$

- Define at each pixel (x, y), the matrix

$$\boldsymbol{H}_{xy} = \begin{bmatrix} S_{xx}(x,y) & S_{xy}(x,y) \\ S_{xy}(x,y) & S_{yy}(x,y) \end{bmatrix}$$

- Compute the response of the detector at each pixel
  - R = *Det* (*H*) - k(trace(*H*))^2, where k is tweakable parameter , generally between 0.04-0.06
  - *Det* (*H*) = ($\lambda_1.\lambda_2$) and trace(*H*) = ($\lambda_1 + \lambda_2$)

  Where, λ_1 and λ_2 are the eigen vlaues of matrix H

- Apply threshold(th) on value of R
  - If R > th, gives the corner points
  - If R < -th, gives the edge points
  - All other points are flat region
- Generate feature vector **x** with λ_1 and λ_2 as the attributes and the target class given by the threshold value.

2.The multivariate Gaussian distribution of the feature vector **x** has the form:

$$p(\boldsymbol{x}|C_i) = \frac{1}{(2\pi)^{\frac{d}{2}}|\boldsymbol{\Sigma}_i|^{\frac{1}{2}}} exp\left[-\frac{1}{2}(\boldsymbol{x} - \boldsymbol{\mu}_i)^T \boldsymbol{\Sigma}_i^{-1}(\boldsymbol{x} - \boldsymbol{\mu}_i)\right]$$

Where, $\boldsymbol{\mu}_i$ is the mean vector of each class and $\sum_i$ is the class covariance matrix given as:

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

3.We choose the class for which Mahalanobis distance $g_i(\boldsymbol{x})$ yields the greatest value.

$$g_i(\boldsymbol{x}) = -\frac{1}{2}log(|\boldsymbol{\Sigma}_i|) - \frac{1}{2}\left[(\boldsymbol{x} - \boldsymbol{\mu}_i)^T \boldsymbol{\Sigma}_i^{-1}(\boldsymbol{x} - \boldsymbol{\mu}_i)\right]$$

4.The next method to classify the feature vector is using the Bayesian rule in which the likelihood of each class is given by P(C_i | $\boldsymbol{x}$).

$$P(C_i|\boldsymbol{x}) = \frac{p(\boldsymbol{x}|C_i)P(C_i)}{p(\boldsymbol{x})}$$

5.Choose class i if P(C_i | **x**) > P(C_i | **x**) for all $j \neq i$

Results

A. Feature vector distribution

We computed the feature vector using eigen values of the Hessian matrix for each of the three classes (flat region, corner, edge). To observe the distribution of these feature vectors, we plotted the distribution for each of the three classes as shown in fig 1.

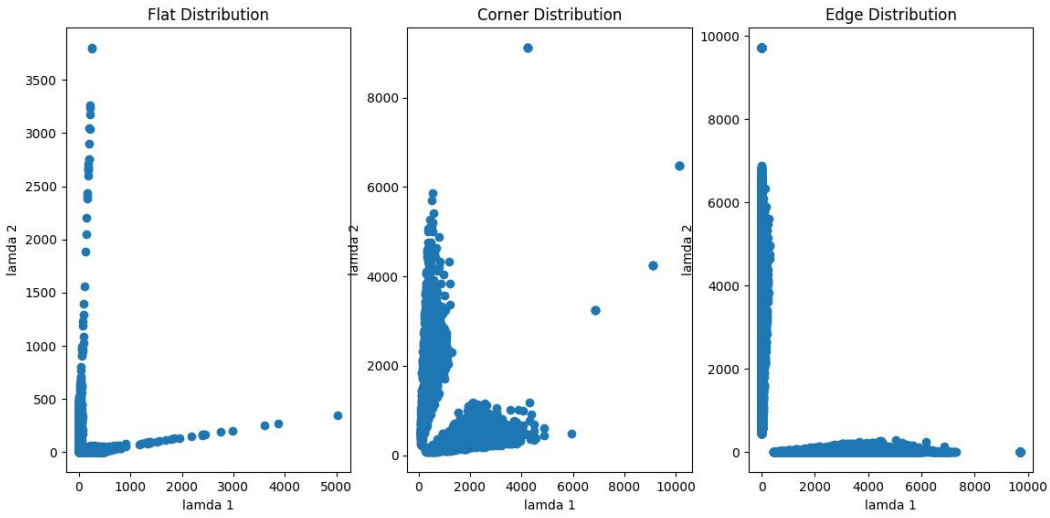

Fig1. Distribution of feature vector (λ_1 and λ_2) for three different classes

B. Classification Accuracy

The classification accuracy of the classifier was obtained to be 93.34%. For the edges, 54.58 % of the data were misclassified whereas 1.67% of misclassification rate was observed in case of flat regions, but the edges 54.58% of misclassification rate.

Error Rate(%)		
Corner	**Edge**	**flat**
8.23	54.58	1.67

Table I. Error rate or misclassification rate for each category of features point

C. Confusion Matrix

The confusion matrix of the classifier is shown in the fig2., where we can see that 4933 feature points that belongs to class 'flat region' were misclassified as "corner" and 132 feature point that belong to class 'corner' were misclassified as 'edge' and 16626 points that belongs 'edge' were misclassified as 'corner'.

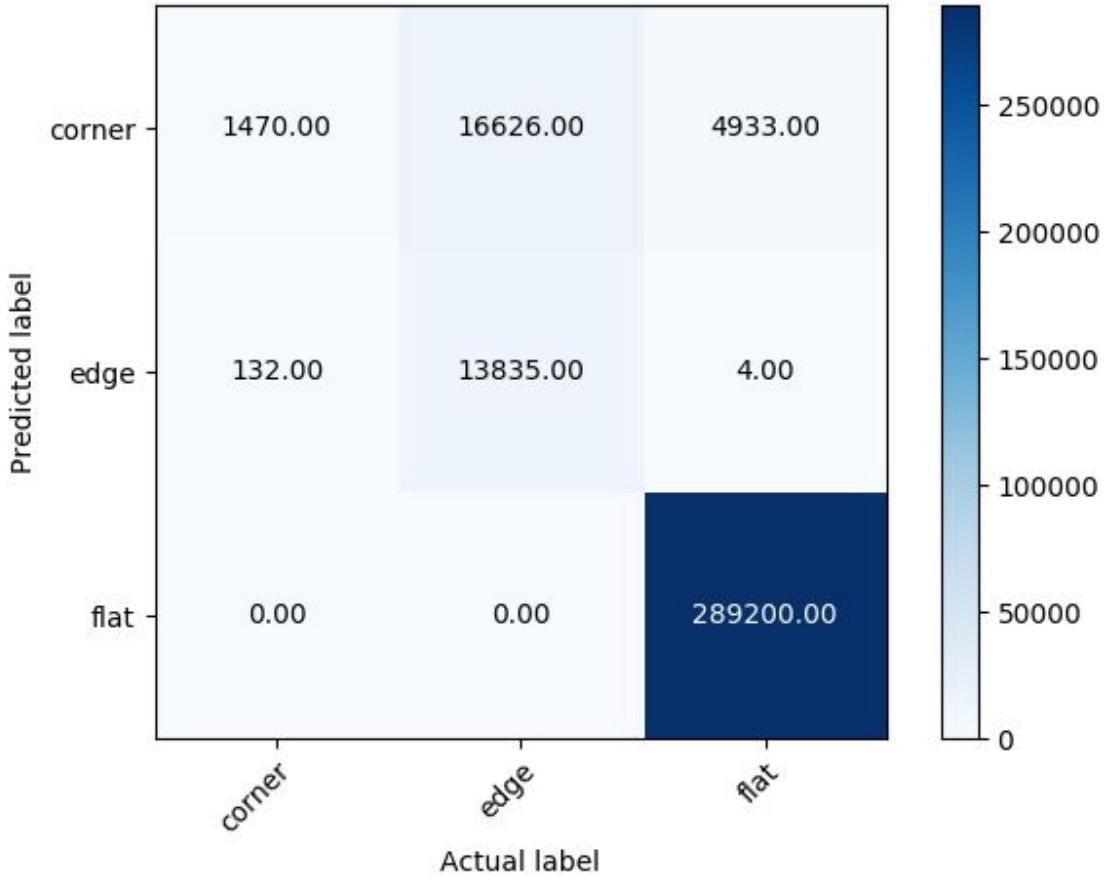

Fig2. Confusion matrix for the classifier

II.**Linear Discriminant Analysis**

Linear discriminant analysis (LDA) or discriminant function analysis is a method used in statistics, pattern recognition, and machine learning to find a linear combination of features which separates two or more classes of objects or events.

Consider a set of observations with *k* classes and each class has n training images. The classification problem is then to find a good predictor for the class of any sample of the same distribution given only an observation.

1. Design a feature vector using Harris Corner Detection algorithm.
   - Compute the R score for each training image
   - For each image, we choose 10 largest positive scores, and 10 largest negative scores to form a 20 x 1 feature vector

$$\boldsymbol{x}_k = [x_{k,1} \quad x_{k,2} \quad \cdots \quad x_{k,20}]^T$$

   - Since there are 50 training images per class, the feature data is 20 x 50 matrix for each class

2.Compute covariance matrix of the entire dataset and mean vectors for each class

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

where , ∑ is the overall covariance matrix of the feature vectors for all the classes and $\boldsymbol{\mu}$ is the overall mean vector for all the classes.

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

Where, $\boldsymbol{\mu}_i$ is the mean vector of each class.

3. Feature vectors extracted from the image are processed by a bank of LDFs $g_i(\boldsymbol{x})$ **,** the LDF for i^{th} class is given as:

$$g_i(\boldsymbol{x}) = \boldsymbol{x}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_i - \frac{1}{2}\boldsymbol{\mu}_i^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_i = \boldsymbol{x}^T\boldsymbol{h}_i + b_i$$

$$\text{Where } \boldsymbol{h}_i = \boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_i \text{ and } b_i = -\frac{1}{2}\boldsymbol{\mu}_i^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_i$$

4. Therefore, for each class, compute $g_i(\boldsymbol{x}) = \boldsymbol{x}^T\boldsymbol{h}_i + b_i$ for $i = 1...k$, and assign **x** the label of the class for which $g_i(\boldsymbol{x})$ *i*s the largest.

$$g_i(\boldsymbol{x}) > g_j(\boldsymbol{x}) \text{ for all } j, j \neq i$$

Results

For the given input dataset of digits with 10 classes (0 – 9), we took 50 images for each digit as training sample and equally 50 images for each digit as test sample. For evaluating the performance of the classifier, we observed the class prediction for both the training and test dataset and calculated the accuracy in both cases.

A.**Confusion matrix**

The confusion matrix of the classifier when evaluated on the test and training sample is shown in fig3 and fig4. We can see that most of the data points were misclassified with this algorithm.

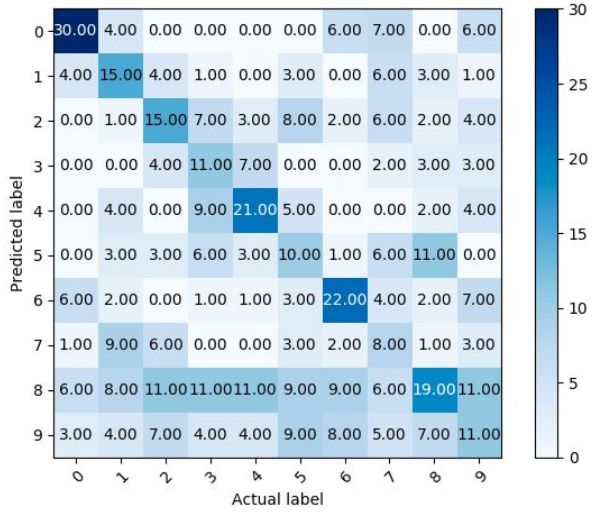

Fig3. Confusion matrix when evaluated on test data

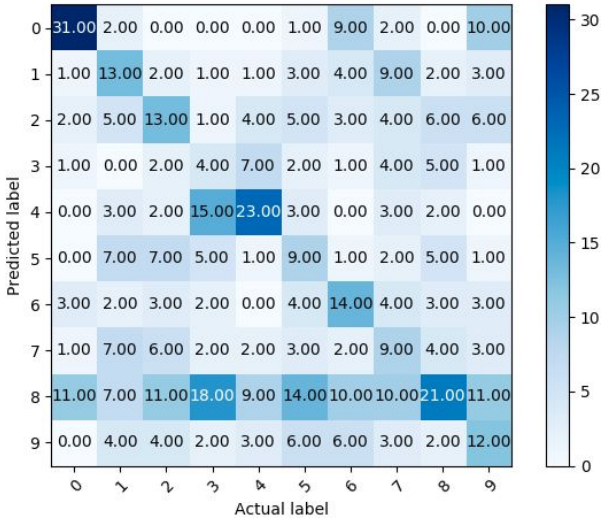

Fig4. Confusion matrix when evaluated on train data

B. **Classification Accuracy**

Also we calculated the error rate for classification of each 10 digits on both test and training data as shown in Table II.The accuracy of the classifier when tested on training sample is 30% and on test data is 32%. In case of training data, digit '2' has the highest misclassification error and digit '0' has the least misclassification error. Similarly, in the case of test data, digit '1' has the highest misclassification error and digit '0' has the least misclassification error.

Data	Error Rate(%)										Accuracy(%)
	0	1	2	3	4	5	6	7	8	9	
Train	44	84	86	57	78	82	72	66	46	80	30
Test	46	84	57.9	68	72	76	68	74	57.9	76	32

TableII. Error rate for each class of digits and overall accuracy on test and train data

Instructions to run the code

1. Install the following required dependencies
   $ sudo apt-get install python3-pip
   $ pip3 install numpy
   $ pip3 install opencv-python
   $ pip3 install matplotlib
2. Unzip the file “4805322.zip”
   $ sudo apt-get install unzip
   $ unzip 4805322.zip

3. From inside the code directory, run the python files
   $ python3 classification.py
   $ python3 LDA.py

References

[1] Lecture notes
[2] https://en.wikipedia.org/wiki/Linear_discriminant_analysis
[3] https://en.wikipedia.org/wiki/Bayes_classifier