Hacettepe University

Artificial Intelligence Engineering Department

AIN433: Introduction to Computer Vision Lab. - 2022 Spring

Programming Assignment 1

March 13, 2022

Student name:
Alperen OZCELIK

Student Number:
21992995

1 Problem Definition

In this assignment, we have 300 images from 10 different categories with 30 images in each category.

What is required of us is to combine the features we will obtain using the PCA algorithm, which is a dimensional reduction technique, and the color histogram, to represent the images with these combined features.

Then, by finding the distances of 20 query images to these images, to find out which category the images belong to and calculate MAP metric for each experiment.

2 Dimensionality Reduction

Dimensionality reduction refers to techniques that reduce the number of input features in a dataset. More input features often make a predictive modeling task more challenging to model, more generally referred to as the curse of dimensionality.

Dimensionality reduction techniques are often used to simplify a classification or regression dataset in order to better fit a predictive model so we can achieve faster results by reducing problem complexity.

2.1 Principal Component Analysis

Principal Component Analysis(PCA) is one of the most popular linear dimension reduction. Sometimes, it is used alone and sometimes as a starting solution for other dimension reduction methods. PCA is a projection based method which transforms the data by projecting it onto a set of orthogonal axes.

In PCA algorithm we normalize our dataset using means then we get the covariance matrix of normalized dataset. After that I calculate the eigenvectors and eigenvalues of the covariance matrix with built-in function then plot the results.

PCA brings together:

- A measure of how each variable is associated with one another. (covariance matrix)
- The directions in which our data are dispersed. (eigenvectors)
- The relative importance of these different directions. (eigenvalues)

The purpose of eigenvectors is to extract features according to their important features. Since eigenvalues show the importance of different directions why we sorted the eigenvalues and choose eigenvectors accordingly.

2.2 Principal Component Analysis Implementation

Example how I add Python code:

```
1
2  class PCA:
3
4      def __init__(self, dataset_matrix, n_components):
5
6          self.M = dataset_matrix
7          self.n = n_components
8          print("Dataset matrix shape:", self.M.shape)
9
10     def mean_vector(self):
11
12         mean_vector = []
13         for i in self.M.T:
14             mean_vector.append(i.mean())
15
16         mean_vector = np.array(mean_vector)
17         mean_vector.resize(1, mean_vector.shape[0])
18         print("Mean vector shape:", mean_vector.shape)
19         return mean_vector
20
21     def normalized_matrix(self):
22
23         D = self.M - self.mean_vector()  # D = normalized matrix
24         return D
25
26     def covariance_matrix(self):
27
28         D = self.normalized_matrix()
29         covariance_matrix = np.dot(D.T, D)
30         print("Covariance matrix shape:", covariance_matrix.shape)
31         return covariance_matrix
32
33     def eigen_vector(self):
34
35         eigenValues, eigenVectors = np.linalg.eig(self.covariance_matrix())
36         idx = eigenValues.argsort()[::-1]
37         V = eigenVectors[idx]
38         V = V[:self.n]  # choose first n eigenvectors
39         return V
40
41     def feature_matrix(self):
42
43         D = self.normalized_matrix()
```

```
        temp_matrix = np.dot(self.eigen_vector(), D.T)
        feature_matrix = np.dot(temp_matrix, D)
        print("PCA feature matrix shape:", feature_matrix.shape)
        return feature_matrix

    def plot(self):

        x_axis, y_axis, z_axis = [], [], []
        feature_matrix = self.feature_matrix()

        for i in range(feature_matrix.shape[1]):
            x_axis.append(feature_matrix[0][i])
            if self.n > 1:
                y_axis.append(feature_matrix[1][i])
            if self.n > 2:
                z_axis.append(feature_matrix[2][i])

        fig = plt.figure()

        if self.n == 1:
            ax = fig.add_subplot()
            ax.set_title('1 Dimension')
            ax.scatter(x_axis, np.zeros_like(x_axis), c='red', marker='o')

        elif self.n == 2:
            ax = fig.add_subplot()
            ax.set_title('2 Dimension')
            ax.scatter(x_axis, y_axis, c='orange', marker='o')

        elif self.n == 3:
            ax = fig.add_subplot(111, projection='3d')
            ax.set_title('3 Dimension')
            ax.scatter(x_axis, y_axis, z_axis, c='purple', marker='o')

        plt.show()
```

2.3 Results of PCA

By making n = 3, we have become able to express each picture (256,256) with a vector (3,1), so we have an algorithm that is much faster and can be more useful in a larger data set.

At the same time, I tried the PCA class with n = 2 and n = 1 parameters then run my algorithm, plot 2-dimensional and 1-dimensional versions as well, so that I also can represent the picture with 2 and 1 features.

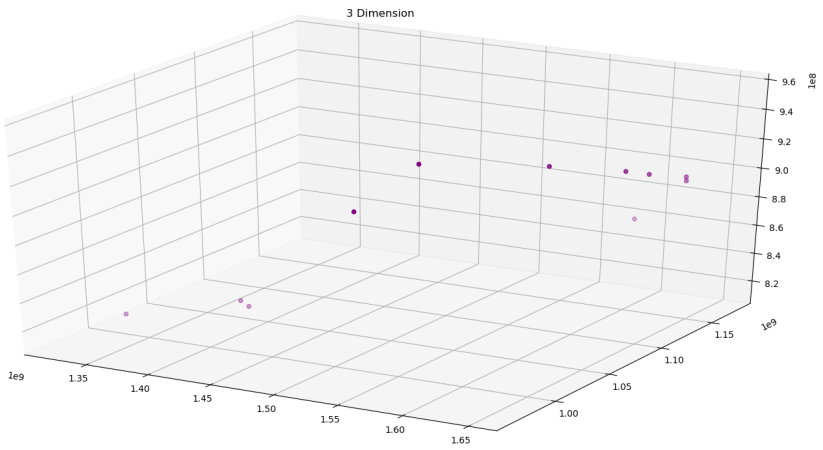

Figure 1: Plot of 3 dimensional data

3 Image Retrieval System

Image retrieval system is an algorithm that is used to find n images with the closest feature to the query image we have given as input.

The key point about content based image retrieval is the feature extraction. The features correspond to the way we represent an image on a high level. While doing image retrieval, I used PCA components and color histogram of images together.

3.1 Color Histogram and PCA Components

The color histogram is a graphical representation showing how frequently various color values occur in the image.

After extracting the histograms of RGB colors in input images, I also obtained PCA components with the PCA class I wrote before. Here, I took the bin parameter of the histograms as '33' then total of 99 values because of 3 colors of RGB, came from the histogram. And in my PCA algorithm, the value of n was 100. so that both the histogram and PCA components were equally effective. I have obtained a feature vector of 199 length for each image.

Then I plot query images from every category with their 10 most similar images according to euclidean distance.

3.2 Results of Image Retrieval System

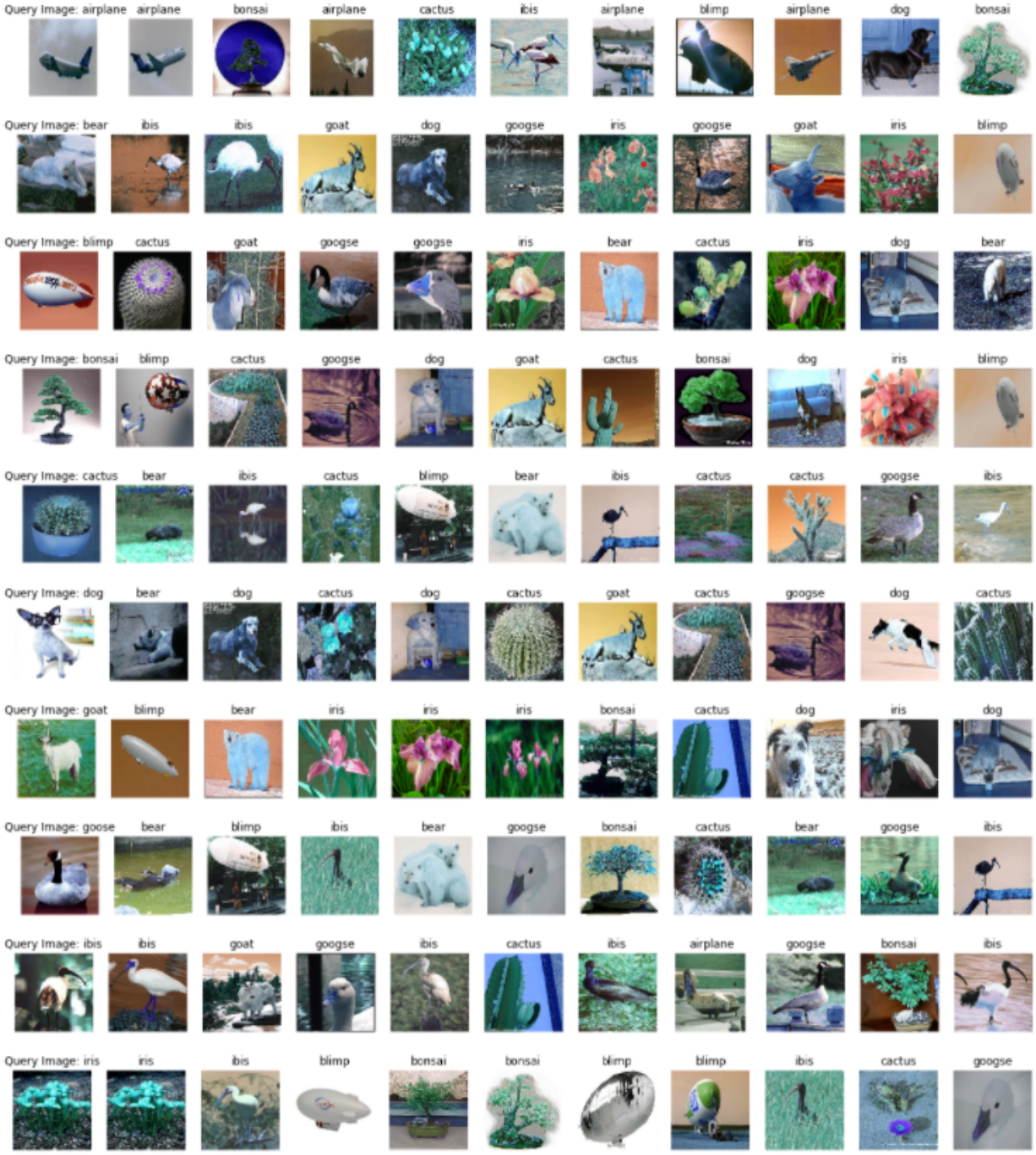

Figure 2: Query Images with 10 most similar images

3.3 Metric Calculation with MAP

I printed the results of all categories using mean average precision.

MAP Results for Each Class	
Class Name	MAP Result
Airplane	0.65
Bear	0.50
Blimp	0.00
Bonsai	0.07
Cactus	0.29
Dog	0.72
Goat	0.00
Goose	0.00
Ibis	0.60
Iris	0.65

Table 1: MAP Results for Each Class

4 Conclusion

If we look at the map results, the model gave successful results in the plane, dog and iris categories, but it did not know at all in some categories. In this project, we worked on a small dataset, but I think we can use this algorithm more efficiently by enlarging the dataset.

References

- https://machinelearningmastery.com/dimensionality-reduction-algorithms-with-python/
- https://builtin.com/data-science/step-step-explanation-principal-component-analysis
- https://datacarpentry.org/image-processing/05-creating-histograms/
- https://www.youtube.com/watch?v=pM6DJ0ZZee0