ECE661: Homework 10

Fall 2022
Due Date: 11:59pm, Dec 09, 2022

Turn in typed solutions via BrightSpace. Additional instructions can be found at BrightSpace. Since this is the last HW of the semester and to ensure timely grading, **no late days are allowed for this assignment**.

1 Introduction

This homework consists of two parts:

1. **PCA, LDA and Autoencoder** for dimensionality reduction and nearest neighbor for face recognition.
2. **Cascaded AdaBoost Classifiers** for object detection.

2 Task 1: Face Recognition using PCA and LDA

The goal of this part of the homework is to classify an unknown face image given a database of labeled face images. Your overall approach will involve the following steps:

1. For this homework, you will use a provided subset of the FacePix database [1]. The images have been divided into a training set and a test set, each with 630 images. The name of each image is a string like `XX-YY.png` where `XX` is the identity of the human subject and `YY` is an integer index for the image for that human subject. Note that while the images in both the training set and the test set share the same names, they are different images.
2. Vectorize the images in the training set and compute the covariance matrix for the image vectors. Do not forget to subtract the mean and normalize each image vector to unit magnitude before you calculate the covariance matrix.
3. Use PCA and LDA to create a low-dimensional representation for the training face images. With each approach, you retain p eigenvectors:

- For PCA, the p eigenvectors are directly the eigenvectors of the covariance matrix. Make sure you use the computational trick for calculating the eigenvectors of the covariance matrix as discussed in Section 2.1 of Prof. Kak's Optimal Subspaces Tutorial [3].
- For LDA, the eigenvectors are of the matrix $S_W^{-1}S_B$ where S_W is the within-class scatter matrix and S_B is the between-class scatter matrix. Should it happen that your S_W is singular, then you must use Yu and Yang's algorithm for finding the LDA eigenvectors (refer to Section 3.5 of [3]).

4. For classification, first project all images of the training set into the p–dimensional subspace. Now project each image in the test set into the same p-dimensional subspace and use *your own implementation* of the Nearest Neighbor (NN) algorithm for classification. For this assignment, show the classification accuracy for both PCA and LDA as a function of the subspace dimensionality p. Subsequently, compare the results for PCA and LDA and how they change with p.

3 Task 2: Face Recognition using an Autoencoder

An autoencoder is a type of neural network that is commonly used for dimensionality reduction. It is comprised of two neural network based components: an encoder and a decoder. The encoder takes in an high-dimensional sample such as an image and outputs a p-dimensional vector representation just like PCA and LDA. The decoder then takes in the p-dimensional vector representation and outputs a sample (*e.g.* an image) in the original high-dimensional sample space. During training, the autoencoder refines itself by attempting to regenerate the input sample from the corresponding p-dimensional vector representation.

In this homework, you will be provided with an autoencoder that is pretrained on the same face dataset [1]. Your task is to implement the rest of the logic to perform the same face recognition task as described in Task 1. Here are the steps:

1. Familiarize yourself with the provided script `autoencoder.py`. It contains the autoencoder class, which is implemented in PyTorch, along with the necessary logic that handles: loading of the pretrained weights, loading of the training and test data, and encoding images into the p-dimensional representation vectors. Additional guidelines can be found in the comments of the script. You can use the same

`conda` environment that you created for previous homeworks for this task.

2. We provide you with three pretrained weights that corresponds to $p = 3, 8, 16$. In the script, you have everything you need to produce the representation vectors of the training set and the test set: `X_train` and `X_test`, respectively. Additionally, their labels are given in `y_train` and `y_test`.

3. Your task for this homework is to utilize your own Nearest Neighbor algorithm to perform the exact same face recognition task as in Task 1. You should repeat the task three times with the three provided model weights and plot the accuracies as a function of p in the same plot along with PCA and LDA.

4 Task 3: Object Detection using Cascaded AdaBoost Classifiers

The goal here is to use the Viola and Jones approach [4] to the design of an object detector with an arbitrarily low false positive rate[1]. The Viola and Jones algorithm carries out a cascaded implementation of AdaBoost classifiers, with each stage of the cascade trying to meet a target false-positive and true-detection rate. You should refer to Section 5 of Prof. Kak's AdaBoost Tutorial [2] for more details on the AdaBoost algorithm and Section 11 for the Viola and Jones approach on designing such a cascade. You will apply your implementation of cascaded AdaBoost classifiers to the proprietary car detection dataset that is provided to you. Please do not distribute the dataset.

5 Submission Instructions

Include a typed report explaining how did you solve the given programming tasks.

1. Turn in a zipped file, it should include (a) a typed self-contained pdf report with source code and results and (b) source code files (only .py files are accepted). Rename your .zip file as hw10_<First Name><Last

[1] In object detection, you have a false positive when you declare a non-object blob of pixels as an object.

Name>.zip and follow the same file naming convention for your pdf report too.

2. **Submit only once on BrightSpace**. Otherwise, we cannot guarantee that your latest submission will be pulled for grading and will not accept related regrade requests.

3. Your pdf must include a description of

   - Task 1
     - A brief description of how you have implemented both PCA and LDA.
     - A plot showing the classification accuracy as a function of the subspace dimensionality p for both PCA and LDA. The classification accuracy is defined as:

$$\text{accuracy} = \frac{\text{\# of test images correctly classified}}{\text{total \# of test images}} \tag{1}$$

     - A discussion on the results comparing PCA and LDA.
   - Task 2
     - In the same plot, plot the classification accuracies as a function of p.
     - A discussion on the results comparing Autoencoder against PCA and LDA.
   - Task 3
     - A brief outline of how you implemented the cascaded classifiers, for both training and inference.
     - A plot showing the false positive (FP) and false negative (FN) rate after the first k stages of the cascade as a function of k:

$$\text{FP} = \frac{\text{\# of misclassified negative test images}}{\text{\# of negative test images}}, \tag{2}$$

$$\text{FN} = \frac{\text{\# of misclassified positive test images}}{\text{\# of positive test images}}. \tag{3}$$

   - Your source code. Make sure that your source code files are adequately commented and cleaned up.

4. To help better provide feedbacks to you, make sure to **number your figures**.

5. The sample solutions from previous years are for reference only. **Your code and final report must be your own work.**

References

[1] FacePix Database. URL https://cubic.asu.edu/content/facepix-database.

[2] The AdaBoost Algorithm for Designing Boosted Classifiers, . URL https://engineering.purdue.edu/kak/Tutorials/AdaBoost.pdf.

[3] PCA (Principal Components Analysis) and LDA (Linear Discriminant Analysis) for Image Recognition, . URL https://engineering.purdue.edu/kak/Tutorials/OptimalSubspaces.pdf.

[4] Viola–Jones object detection framework. URL https://en.wikipedia.org/wiki/Viola%E2%80%93Jones_object_detection_framework.