HW4 – submission 30.6.19 23:55

Guide lines

1. Include all your personal details including name, id, and **e-mail address.**
2. You should submit all functions and script files written in **MATLAB or Python**. Your code should be well documented and clear. The code should run from **any** computer and include all path definitions (You should take care of this in the code).
3. Please divide the code by questions.
4. Final report – should include explanations on the implementation and the execution, answers to the questions, results, conclusions and visual results. Do elaborate on all parts of the algorithms/solution. The grades are highly depended upon the analysis depth of the report. **Please submit a PDF file and not a DOC file.**
5. Please post question regarding this HW on the QA forum in Moodle.
6. Eventually submit one zip file including the code + images PDF. HW can be submitted in pairs. In such case only one of the student should submit the solution. The zip file should be named HW#_id0_id1 (for example: HW1_123456789_012345678).
7. Please follow all the submission instruction as describe here.

Good luck!

Question 1 – Eigen-Faces and PCA for face recognition:

In this exercise, you will implement the Eigen-faces algorithm for face recognition. Use the provided stub code (`eigenface.m`) and implement the algorithm according to following steps:

1. Prepare the training set:
   a. The training set is loaded using the function `readYaleFaces.m` (line 3 on the given code). Read the documentation for more details.
   b. Calculate the mean face of the trainset and subtract it from all the training images.

2. Calculate the eigen-faces:
   a. Implement a function that finds the r largest eigen-vectors of the trainset covariance matrix:
   $\Sigma = E\left[(X-\mu)(X-\mu)^T \right]$

   To avoid memory issues, use SVD as describes in tutorial 7 sections 1.1+1.2, or read the relevant section in [1]. Implement your algorithm accordingly and explain in detail.
   b. Show the five biggest eigen-faces as images.
3. Reconstruction:
   a. For each image in the train set $\{x_j\}$, subtract the average face (calculated in 1.b) and calculate its low dimension representation vector $y_j = W^T\left(x_j - \overline{x}\right)$ according to the $r = 25$ largest eigen-faces (W is a matrix build from the top r eigen-faces).
   b. Reconstruct each of the images in the train set $\{x_j\}$ according to its representation y_j. Calculate the representation error.

      Note: calculate two types of errors between the original image and the reconstructed image:
      i. **Dynamic range RMSE** – normalize each image by its dynamic range [max(graylevel)-min(graylevel)] and calculate RMSE.
      ii. **RMSE error** – calculate the RMSE between the images and normalize by 255.
   c. Describe and explain your results. What is the average representation error?
4. Recognition:
   a. For each image in the **Test set** $\{x_i\}$, find the representation vector y_i according to the $r = 25$ top eigen-faces (don't forget to subtract the train set average face). What is the mean representation error? Was there any change in comparison to the train set? Explain.
   b. Classify each image from the **test** set **(use only the images of people who appear also on the train set).** This can be done by using nearest neighbor classifier between each y_i to the train set represent vectors $\{y_j\}$. For nearest neighbor classification, use the matlab function `fitcknn.m`. What are the success ratios (for face_id)?

Question 2 –FLD for face recognition:

In this part you will implement the Fisherfaces technique for face recognition. This technique is similar to the Eigenfaces technique, with a few important differences that improve its recognition performance.

1. Complete the stub code `fishertrain.m` that computes the Fisher basis for a given training set "`face_data`", with a specific labeling "`label_train`". Follow these steps:
   a. Use the algorithm you developed in the previous question to compute a basis with r= N-C components. Where N is the number of train images and C is a given parameter. This will be W_{pca}.

   As we saw in tutorial 11, for the FLD is done by constructing a scatter matrices S_B and S_W. As these matrices are extremely large, we would like to reduce their dimension. This can be done by projecting them onto the PCA basis found on the previous section:

   $$\bar{S} = W_{pca}^T S W_{pca}$$

   In practice, this projection can be done on the data itself and not on the scatter matrices (the two are equivalent!):

   b. **Project all the faces in the train set onto the W_{pca} basis found on (a) to find their low dimension representations $\left\{y_j\right\}$. Do not forget to subtract the mean face.**
   c. **Calculate $\bar{S}_B$ and by $\bar{S}_W$ W_{pca} in the following way:**

   $$\bar{S}_W = \sum_{k=1}^{K} \sum_{y_i \in c_k} \left(y_i - \mu_k^y\right)\left(y_i - \mu_k^y\right)^T$$

   $$\bar{S}_B = \sum_{k=1}^{K} N_k \left(\mu_k^y - \mu^y\right)\left(\mu_k^y - \mu^y\right)^T$$

   Where μ_k^y is the average __representation__ of each class: $\mu_k^y = \frac{1}{N_{c_k}} \sum_{y_i \in c_k} y_i$, μ^y is the average representation over all the data: $\mu^y = \frac{1}{N}\sum_{i=1}^{N} y_i$ and N is the total number of images.

   d. Compute W_{fld}: This matrix contains the C –1 largest eigen vectors found using the generalized eigenvalues decomposition: $\bar{S}_B w_i = \lambda_i \bar{S}_W w_i$ (you may use the function `eigs` **which is capable with solving the generalized eigenvalues problem**).
   e. The resulting Fisher basis is $W = W_{pca} W_{fld}$. Explain why.

You will now use this function to classify the identity of the faces in the test set, using "`train_face_id`" as the labeling for training the Fisher basis.

2. Calculate the representation errors on the train set and test set, as you did in the previous question. **Use the parameter C = 15.**
3. Classify the test set **(use only the images of people who appear also on the train set)**. What is the success ratio? Compare the results obtained by the Eigenfaces method to those obtained by the Fisherfaces method. Which is better? Why?

References:

[1] Turk, Matthew, and Alex Pentland. "Eigenfaces for recognition." *Journal of cognitive neuroscience* 3.1

[2] A Tutorial on Principal Component Analysis - Jonathon Shlens

[3] P.N. Belhumeur, J.P. Hespanha and D.J. Kriegman, Eigenfaces vs. Fisherfaces: Recognition using class specific linear projection, IEEE trans. on PAMI, Vol. 19, pp. 711-720, 1997.."