Face Recognition Using Eigenfaces, LDA, SVM and SRC

Hw3 report written by Xiaodong
March 9, 2020

Summary: The report discusses the result of four different face recognition techniques. These algorithms are Eigenfaces, Fisherfaces, Support Vector Machine (SVM), and Sparse Representation-based Classification (SRC). I will compare the performance of the four methods on the same data set with same training examples and same testing examples being selected. I will also discusses the performance of HOG and LBP features on SVMs. I will then discusses the relationship between the number of training examples and the error rate.

Approach: I used python for this project, Bokeh library for plotting; and I took the following steps for my project.

1. Segment the original data and randomize the training and testing samples.

2. Load data of index of training and index of testing for m = 10,20,30,40,50. where m is the number of training data.

3. Perform the Eigenfaces without using library.

4. Perform LDA without using library.

5. Perform LDA using library.

6 Perform SVM with library.

7. Perform SRC using library.

8. Perform HOG and LBP features on SVMs The following are the detailed explanation of implementation for each step:

1. Segment the original data and randomize the training and testing samples: This step corresponding to file *datasegmentation.py*. I first segmented the original data into 38 people, which means 38 clusters. Most people have 64 images but some people have less than 64 images. This step randomly selected m images from each person as training data and n-m data as testing data, where m =10,20,30,40,50. and n is the number of images for each person. Then m10.pkl,m20.pkl,m30.pkl,m40.pkl,m50.pkl is output as 5 files. The segmentation of data process only run once to keep consistence on the experiments. Thus, make the following methods comparable.

2. Load data of index of training and index of testing for m = 10,20,30,40,50. This step corresponding to file load.py This step load the data from the segmented data file m10.pkl to m50.pkl. load.py is called for all the following files.

3. Perform the Eigenfaces without using library. This step uses PCA which represents each image as a linear combination of Eigenfaces. The eigenvector decomposition of S is given by $Sv_i = TT^Tv_i = \lambda_i v_i$. I also used singular value decomposition to compute the eigenvector where $X = U\Sigma V^T$ the eigenfaces are then the first$k(k <= n)$ columns of U associated with SVD.

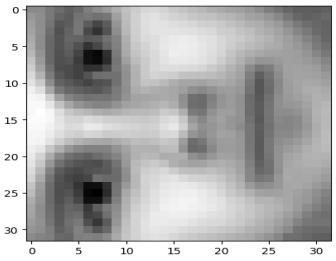


Fig1.average face

As shown in Fig.1, the eigenface vector first computes an average face. Then, the PCA components are then computed.

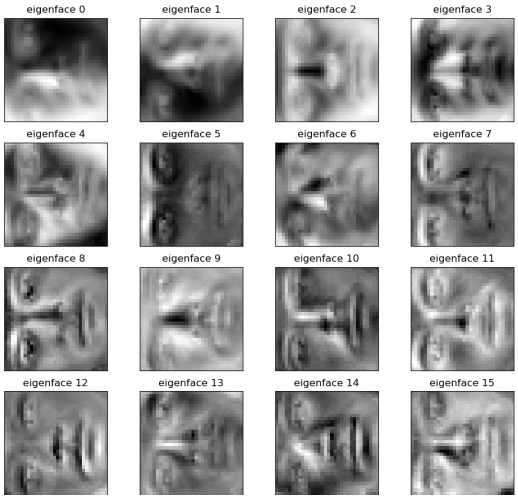


Fig2.First 16 PCAs

As we can see, the first few PCAs primarily shows the lighting effects on the image. Thus, discarding the first few images improves the performance.

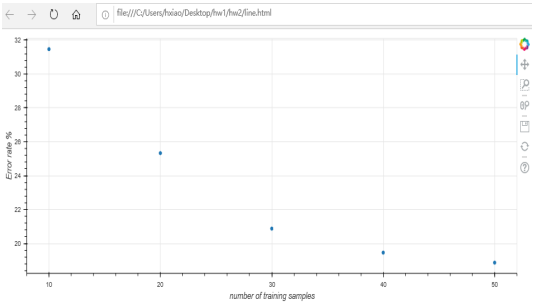
4. Perform LDA without using library.5. Perform LDA using library. LDA has similar approach as PCA. LDA not only maximize the between cluster scatter but also minimize with in class scatter.$S = \sigma^2_{between}/\sigma^2_{within}$ I wrote the LDA algorithm without using library and used SVD $Sv_i = TT^Tv_i = \lambda_i v_i$ to calculate the eigenvectors.

	m10	m20	m30	m40	m50
0	2034.000000	1654.000000	1274.000000	894.000000	514.000000
1	939.000000	557.000000	347.000000	215.000000	124.000000
2	46.165192	33.675937	27.237049	24.049217	24.124514

Table1. Eigenfaces(50_0)

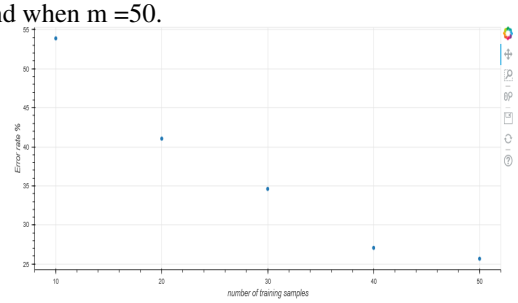
	m10	m20	m30	m40	m50
0	2034.000000	1654.000000	1274.000000	894.000000	514.000000
1	640.000000	419.000000	266.000000	174.000000	97.000000
2	31.465093	25.332527	20.879121	19.463087	18.871595

Table2. Eigenfaces(50_discard_first_3_vectors)



error rate PCA

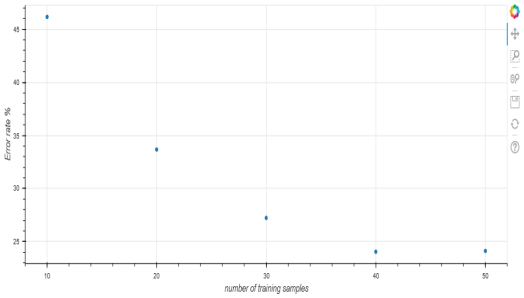
As shown in the previous tables, discarding the first 3 components improves the performance by 3.2 %. The error rate of Eigenfaces is 18.3% with 3 components discarded and when m =50.



error rate LDA my code

	m10	m20	m30	m40	m50
0	2034.000000	1654.000000	1274.000000	894.000000	514.000000
1	1117.000000	684.000000	448.000000	252.000000	138.000000
2	54.916421	41.354293	35.164835	28.187919	26.848249

Table3. LDA: my_code



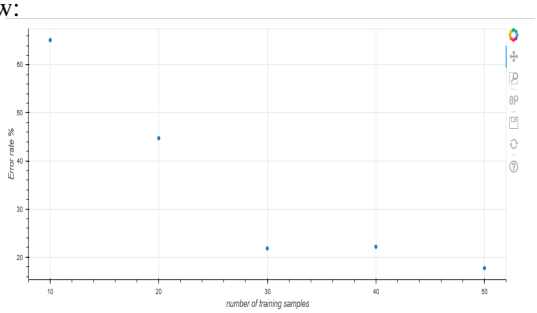
error rate LDA using lib

	m10	m20	m30	m40	m50
0	2034.000000	1654.000000	1274.000000	894.000000	514.000000
1	1096.000000	679.000000	441.000000	242.000000	132.000000
2	53.883972	41.051995	34.615385	27.069351	25.680934

Table4. Fisher vector plot: use_library

However, there is no significant improvement of LDA. I tried with using LDA library too, but I produced similar results. 26%

6 Perform SVM with library. SVM is support vector machines used for face classification thus achieve recognition. Linear kernels are used for my code. The results are as follow:



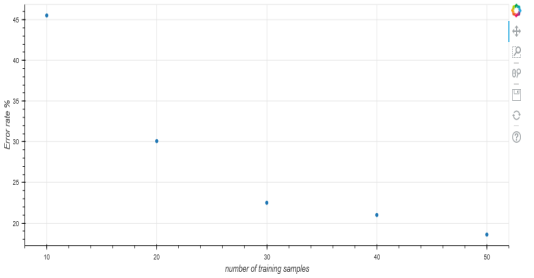
	m10	m20	m30	m40	m50
0	2034.000000	1654.000000	894.000000	894.000000	514.000000
1	1325.000000	740.000000	195.000000	198.000000	91.000000
2	65.142576	44.740024	21.812081	22.147651	17.704286

Table 5. Svm

svm

An error rate of 17% is achieved when using m = 50;

7. Perform SRC using library. SRC use sparse representation based - classification. Which has dictionary as$(U^*, V^*) = argmin0.5||Y - UV||_2^2 + alpha * ||U||_1$ The results are shown as follow:



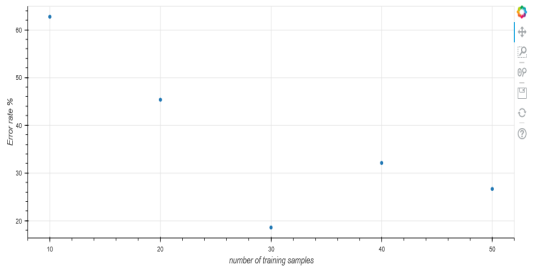
SRC

	m10	m20	m30	m40	m50
0	2034.000000	1654.000000	1274.000000	894.000000	514.000000
1	926.000000	497.666667	286.750000	187.777778	95.600000
2	45.526057	30.088674	22.507849	21.004226	18.599222

Table 8.SRC

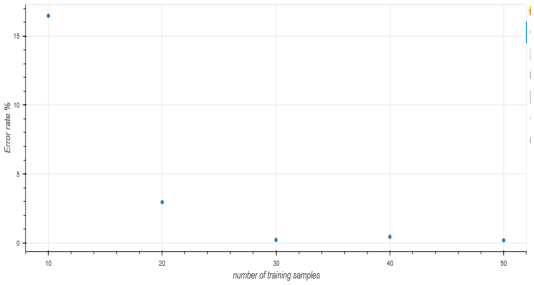
An error rate of 18.5% is achieved when using m = 50;

8. Perform HOG and LBP features on SVMs; Using HOG and LBP features can potentially improve the recognition performance; I tested it on SVMs and the results are amazing for LBP.



	m10	m20	m30	m40	m50
0	2034.00000	1654.000000	894.000000	894.000000	514.000000
1	1276.00000	750.000000	166.000000	287.000000	137.000000
2	62.73353	45.344619	18.568233	32.102908	26.653696

Table 6.svm_hog feature



	m10	m20	m30	m40	m50
0	2034.00000	1654.000000	894.000000	894.000000	514.000000
1	335.00000	49.000000	2.000000	4.000000	1.000000
2	16.47001	2.962515	0.223714	0.447427	0.194553

Table 7. Svm_lbp

An error rate of 0.19% is achieved when using m = 50 with LBP. An error rate of 26.6% is achieved when using m = 50 with HOG features.

Results: By observation, the LBP features performs exceptionally good with SVM with an error rate of 0.19% when m=50. We can also clearly see the relationship between the number of training data and error rate. As the ratio of training data increases, the error rate decreases.

Reflections: The SRC and LDA can perform better with more fine tuning the algorithm. From my experiment, SVM has the best error rate out of all methods. LBP feature produces the lowest error rate compared with other features.

References