Final project BioE 594

Face and Gender Recognition

Giulia Crocioni

May 2019

Face recognition computational models are of great interest because of their wide practical applications, such as criminal identification, image processing, human-computer interaction and security systems [1]. The face recognition problem can be treated as a 2-D recognition problem considering that faces may be described by a set of 2-D views. In Principal Component Analysis (PCA) the faces are projected onto a feature space, in which "eigenfaces" represent the significant features and are eingenvectors of the set of face images. A weighted sum of the eigenface features is computed in the projection operation, and then the comparison between these weights and the ones of an unknown face allows its recognition. In the second part of the project, gender recognition was explored using Convolutional Neural Networks approach. Finally, the gender classification was also done using Support Vector Machine (SVM) approach to perform supervised learning over the feature space.

1 Methods

1.1 Face recognition using Eigenfaces Approach [2]

The first step consists in forming a training data set. Let a face image I_i be a 2D m by n array of intensity values. It can be represented as a 1D vector by concatenating rows, and so transforming the image into a vector of length $N = mn$. Such vectors $\boldsymbol{x_i}$ $(i = 1, 2, ..., M)$ of length N form X, the matrix of learning images. Then the vector of mean values $\boldsymbol{\Psi}$ is determined and subtracted from each image vector.

$$\boldsymbol{\Psi} = \frac{1}{M}\sum_{i=1}^{M}\boldsymbol{x_i} \tag{1}$$

$$\boldsymbol{\phi_i} = \boldsymbol{x_i} - \boldsymbol{\Psi} \tag{2}$$

$\boldsymbol{\phi_i}$ vectors form a new training matrix $A = (\boldsymbol{\phi_1}, \boldsymbol{\phi_2}, ...\boldsymbol{\phi_M})$ (size N by M). Then the covariance matrix $\boldsymbol{C}$ is calculated, and its eigenvectors $\boldsymbol{e_i}$ and eigenvalues λ_i are found.

$$\boldsymbol{C} = \frac{1}{M}\sum_{n=1}^{M}\boldsymbol{\phi_n}\boldsymbol{\phi_n}^T = \boldsymbol{A}\boldsymbol{A}^T \tag{3}$$

$$\boldsymbol{C}\boldsymbol{e_i} = \lambda_i\boldsymbol{e_i} \tag{4}$$

The size of C is $N \times N$, and consequently it has N eigenvalues and N eigenvectors. This means that if the image size is 128×128 there would be 16.384 eigenvectors and 16.384 eigenvalues. But being the rank of convariance matrix limited by the number of training set images, given M images only M-1 eigenvectors correspond to non-zero eigenvalues. According to one theorem in linear algebra the eigenvectors $\boldsymbol{e_i}$ and eigenvalues λ_i can be obtained by finding eigenvectors and eigenvalues of the matrix $\boldsymbol{C_1} = \boldsymbol{A}^T\boldsymbol{A}$, which has dimensions $M \times M$. Thus,

$$\boldsymbol{A}^T\boldsymbol{A}\boldsymbol{v_i} = \mu_i\boldsymbol{v_i} \tag{5}$$

with $\boldsymbol{v_i}$ and μ_i eigenvectors and eigenvalues of the matrix $\boldsymbol{C_1}$, respectively. If both sides of the equation 5 are multiplied with $\boldsymbol{A}$ from the left:

$$\boldsymbol{C}(\boldsymbol{A}\boldsymbol{v_i}) = \mu_i(\boldsymbol{A}\boldsymbol{v_i}) \tag{6}$$

Thus, the first $M-1$ eigenvectors $\boldsymbol{e_i}$ and eigenvalues λ_i of the matrix $\boldsymbol{C}$ are given by $\boldsymbol{A}\boldsymbol{v_i}$ and μ_i, respectively.
The highest variance is reflected by the eigenvector associated with the highest eigenvalue. Because of the exponentially decrease of eigenvalues, about 90% of the total variance is contained in the first 5% to 10% eigenvectors. Thus, eigenvectors are sorted so that the first vector corresponds to the highest eigenvalue. The sorted eigenvectors (column vectors) are normalized and

form a new matrix $\boldsymbol{E}$ of size $N \times D$, with D desired number of eigenvectors. Data matrix $\boldsymbol{A}$ is projected using $\boldsymbol{E}$:

$$\boldsymbol{Y} = \boldsymbol{E}^T \boldsymbol{A} \tag{7}$$

with $\boldsymbol{Y} = (\boldsymbol{y_1}, ..., \boldsymbol{y_M})$. Finally faces of the test set are recognized. The image of the face that has to be recognized is transformed into a vector $\boldsymbol{P}$. Then the mean value $\boldsymbol{\Psi}$ is subtracted and the obtained vector is projected with the matrix of eigenvectors (eigenfaces):

$$\boldsymbol{w} = \boldsymbol{E}^T (\boldsymbol{P} - \boldsymbol{\Psi}) \tag{8}$$

The classification is done by determining the Euclidean distance ϵ_i between $\boldsymbol{w}$ and each vector $\boldsymbol{y_i}$ of the matrix $\boldsymbol{Y}$. If $\boldsymbol{A}$ and $\boldsymbol{B}$ are two vectors of length $\boldsymbol{D}$, the Euclidean distance between them is:

$$d(\boldsymbol{A}, \boldsymbol{B}) = \sqrt{\sum_{i=1}^{D} (a_i - b_i)^2} = \|\boldsymbol{A} - \boldsymbol{B}\| \tag{9}$$

The test face is considered to be unknown if the minimum distance between test face and training faces is higher than an arbitrary threshold θ. Otherwise the test face is considered to be known and belongs to the person $s = argmin[\epsilon_i]$.
The most common way to determine the threshold is to first calculate the minimum distance of each image from the training data from the other images, place it in a vector *rast* and then:

$$\theta = 0.8 * max(rast) \tag{10}$$

1.2 Gender detection using Deep Learning

After the implementation of the face recognition algorithm, faces gender detection was also explored. A Keras model was used adapting the code from `https://github.com/arunponnusamy/gender-detection-keras`. Keras is a library written in Python which enable experimentation with Neural Networks. The Keras backend implementation chosen for this application was TensorFlow, an open-source symbolic tensor manipulation framework developed by Google. [3] The training set was composed by 2200 face images, 1100 for male class and 1100 for female class. It was used to create the

Keras model by training SmallerVGGNet, one of the Convolutional Neural Networks of Keras. The VGG network architecture, introduced by Simonyan and Zisserman in 2014, uses 3x3 convolutional layers stacked on top of each other. In order to make training easier, smaller versions of VGG can be trained, such as SmallerVGGNet, which has less weight layers respect to the original architecture. [4]

After the loading of the pre-trained model, the faces in the test image are detected through the function `detect_face()`, which uses a Deep Neural Network module of OpenCV, a library aimed at real-time computer vision. After the faces detection, the gender of each face is predicted and printed on the test image, together with the confidence of the prediction.

1.3 Gender detection using SVM

SVM approach can be used also to identify the gender of a person starting from his/her face photograph. The algorithm was adapted from `https://github.com/imsiddhartha/Gender-Detection-From-Facial-Features`. It uses Scikit-Learn (`sklearn`), a Python machine learning library which features classification, regression, SVMs, and other data mining and analysis tools. Scikit-Learn contains for example the `svm` library, in which there are built-in classes for different SVM algorithms.

After the face images opening, their histograms are equalized. Then the Scikit-Learn function `train_test_split()` is used to create the train and the test subsets, in particular randomly using 20% of dataset as test sets, and the remaining 80% as training set. Then the support vector classifier class `SVC` is used to classify the data. The parameter of this class is the kernel type, which is chosen to be the "linear" one, meaning that it classifies linearly separable data. Thus, the algorithm is trained on the training set, which is passed as a parameter to the function `fit()`, used to fit the model. The function `predict()` is used on the test set to predict values based upon the model trained by `svm`. Finally, the confusion matrix and the accuracy of the alogorithm are displayed.

2 Results

2.1 Face recognition using Eigenfaces Approach

The dataset of faces used is from "MIT face recognition project". The subject is photographed in a frontal position and the dimensions of the photos are 128×128, with a completely black background. They are grayscale and intensity levels of gray are taken as image features. The training set contains 83 faces of 25 people (from 1 to 4 images per person) and some of them are shown in Figure 1. The mean face of the training set is shown in Figure 2. The test set has 27 images of different people (25 known and 2 unknown) and some of them are shown in Figure 3.

Figure 1: Some faces of the training set.

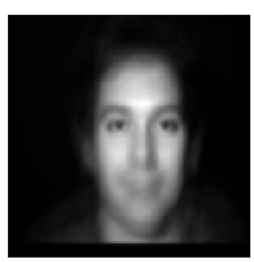

Figure 2: Mean face of the training set.

Figure 3: Some faces of the test set.

Some of the training set faces after the subtraction of the mean face are shown in Figure 4.

Figure 4: Some faces of the training set after the subtraction of the mean face.

Some of the training set faces after their projection onto the largest eigenvectors are shown in Figure 5.

Every image name (both in train and test set) has a number which identifies a subject (e.g. 01, 02, ..., 27). If the minimum distance between test face and training faces is lower than the threshold θ, the test face is considered to be known. When the algorithm associates one face of the test set to one face

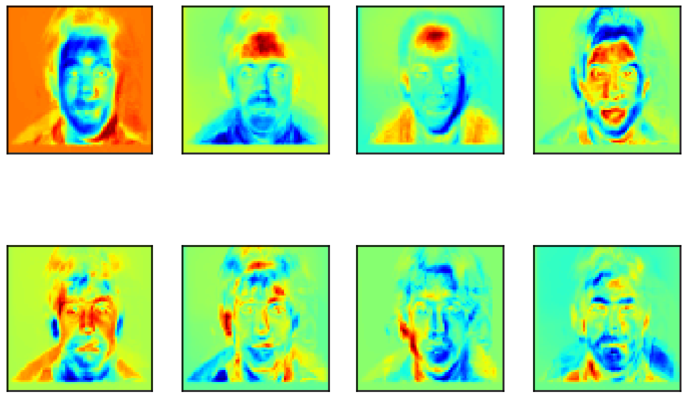

Figure 5: Some faces of the training set after their projection onto the largest eigenvectors.

of the training set, it also checks if the numbers of the two images are the same and in this case it increases the counter of correct matchings by one. When two images are plotted (Figure 6) the train image title is displayed in green to indicate a correct matching. In Figure 7 the case in which the two subjects do not match is shown. The title of the training image is now in red.

Figure 6: On the left, subject 10 image from the test set. On the right, the face of the train set associated to the first one by the algorithm. The title of the train set face is in green because the matching is correct.

Figure 7: On the left, subject 26 image from the test set. On the right, the face of the train set associated to the first one by the algorithm. The title of the train set face is in red because the matching is not correct. In particular, subject 26 is one of the unknown subjects.

When the minimum distance between test face and training faces is higher than the threshold θ, the test face is considered to be unknown. The case in which the prediction is correct is shown in Figure 8 . Subject 27 is not in the training set, and the algorithm classified her as unknown. The opposite case is shown in Figure 9, in which the subject of the test face is known because he is in the training set, but the algorithm classified him as unknown.

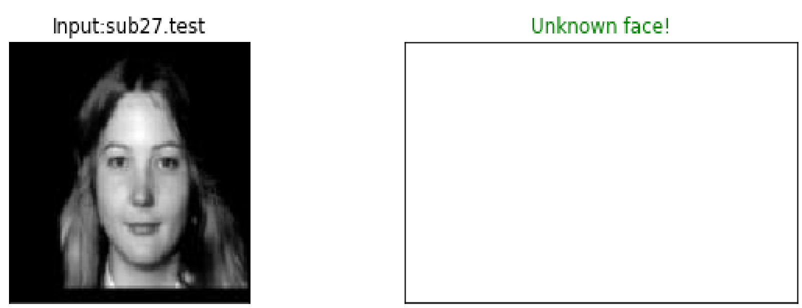

Figure 8: On the left, subject 27 image from the test set. On the right, the algorithm classified it as unknown face. The title is in green because the prediction is correct.

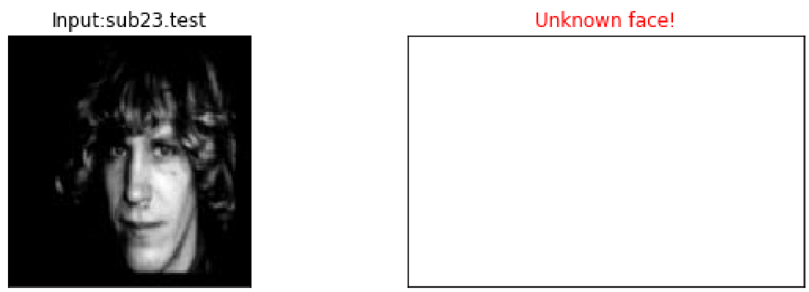

Figure 9: On the left, subject 23 image from the test set. On the right, the algorithm classified it as unknown face. The title is in red because the prediction is not correct. In particular, subject 23 is one of the known subjects.

In order to choose the necessary number of principal components, the cumulative proportion of variance explained vector was considered. In Figure 10 it is plotted in function of principal components number. Table 1 shows the results of the algorithm choosing different numbers of principal components. Note that the recognition rate given by choosing the first 15 eigenvectors is equal to the one given by choosing the first 82 eigenvectors. This is due to the fact that with the first 15 principal components the cumulative proportion of variance explained is already significant (83.6%).

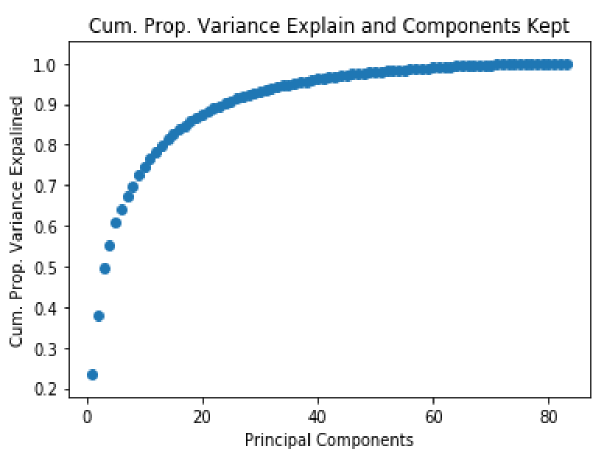

Figure 10: Cumulative proportion of variance explained vector in function of number of principal components

Number of principal components	Cumulative proportion of variance explained	Recognition rate
5	63.9%	77.78% (21/27)
10	76.4%	88.88% (24/27)
15	83.6%	92.59% (25/27)
25	91.3%	92.59% (25/27)
50	97.98%	92.59% (25/27)
82	100%	92.59% (25/27)

Table 1: Cumulative proportion of variance explained and recognition rate in function of number of principal components considered

In order to analyse the best threshold θ that can be chosen using $rast$ vector, its maximum ($max(rast)$) was multiplied for different coefficients, with a minimum of 0.1 and a maximum of 1. The % of correct predictions in function of the multipliers is shown in Figure 11 and the best results are obtained using a coefficient from 0.5 to 0.8. The latter was the multiplier used in the algorithm to compute θ.

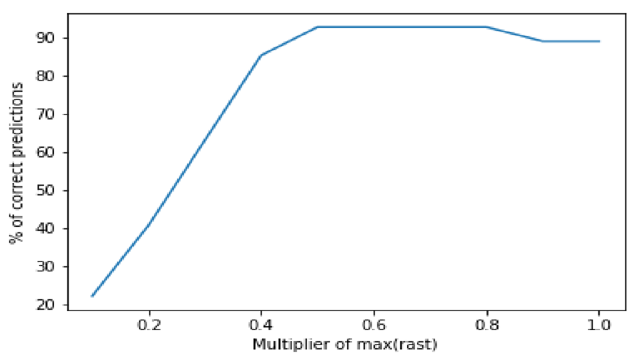

Figure 11: The % of correct predictions in function of different multipliers of $max(rast)$.

2.2 Gender detection using Deep Learning

The test set taken from "MIT face recognition project" was used also in the second part of the project. It has 27 images of people of different ages, among which there are 7 females and 20 males. An example of output given by the gender detection algorithm previously mentioned is shown in Figure 12. The face is identified and highlighted by means of a green rectangle, above which the predicted gender and the confidence of the prediction made are printed.

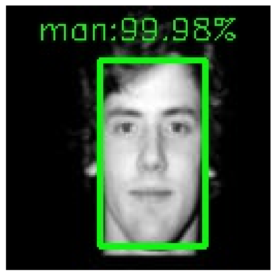

Figure 12: Example of output image of the gender detection alogorithm, which uses a Keras pre-trained model.

In Table 2 the confusion matrix of the obtained results is shown. The overall accuracy is 77.78 %, but there is a clear difference between male and female classes. In fact, while 19 out of 20 males were correctly recognized by the model, for female class only 2 out of 7 predictions were correct. Since it has been reported that the Keras model acheived around 96% of training accuracy and $\sim$90% of validation accuracy, the lower percentage obtained with the test set taken from "MIT face recognition project" may be due to the extremely dark images, in which the hair or other details are not always distinguishable. As example figure 13 shows one of the test set images, in which the photographed subject (female) does not have really marked female features (e.g. long hair).

Table 2: Confusion matrix of gender predictions using Deep Learning

		Actual class	
		Male	Female
Predicted class	Male	19	5
	Female	1	2

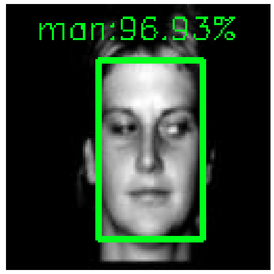

Figure 13: One of the test image for which the prediction was wrong.

2.3 Gender detection using SVM

The dataset consisted of 208 face images (128 males and 80 females), taken from "MIT face recognition project". Consequently, the images in the training set are 166 (80 %), while the images in the test set are 42 (20 %). The first letter of every image's name is an `m` or an `f`, depending on the gender of the subject, male or female, respectively. This was done in order to make possible the comparison between the predictions done by the algorithm and the actual gender of the subject. In Table 3 the confusion matrix of the obtained results is shown. The overall accuracy is 80.95 %, and the difference between male and female class is less evident than in Subsection 2.2. This may be due to the fact that in this case the training set was constituted by images from the same dataset of the test images, while in the previous subsection case the training set was composed by face images from a different dataset.

Table 3: Confusion matrix of gender predictions using SVM

	Actual class		
		Male	Female
Predicted class	Male	21	5
	Female	3	13

Finally, different kernel types were compared. Kernels `linear` and `poly` give an overall accuracy of 80.95 %, and the confusion matrix obtained is the one shown in Table 3. Thus, they resulted the best kernels to be used in this classification problem with this dataset. In fact, the accuracy obtained using `rbf` and `sigmoid` kernels was only 61.90 %.

References

[1] Matthew Turk and Alex Pentland. Eigenfaces for Recognition. *Journal of Cognitive Neuroscience*, 3(1):71–86, January 1991.

[2] Marijeta Slavkovic and Dubravka Jevtic. Face recognition using eigenface approach. *Serbian Journal of Electrical Engineering*, 9(1):121–130, 2012.

[3] Keras backends. https://keras.io/backend/.

[4] Adrian Rosebrock. ImageNet: VGGNet, ResNet, Inception, and Xception with Keras, 2017. https://www.pyimagesearch.com/2017/03/20/imagenet-vggnet-resnet-inception-xception-keras/.