Face recognition based on eigenfaces

September 21, 2023

Abstract

In more recent years Face Recognition introduced itself in our lives. We use it almost every day and sometimes without even noticing anymore. For example, whenever we post something on Facebook or import a picture inn Apple's Photo app, the software automatically recognizes the faces and assigns labels. Also, the newest smartphones have face recognition as the main security option for accessing the device. In some states, face recognition is even used to reduce crime and prevent violence in the streets. Since there are so many everyday life uses these software have to be able to recognize and handle the problems deriving from having different backgrounds, facial expressions or light conditions. Therefore, face recognition has become a common problem in Machine Learning, and it has been approached in different ways. One of the simplest techniques is eigenfaces: it is quite efficient, and it usually generates good results. Face recognition consists in two main phases: one is about finding the main features of the images and the second step is the one that classifies the various pictures as belonging to one determined person or another.[1]

1 Introduction

Faces can be seen as elaborated multiform structures and need excellent computing techniques for recognition. Different methods have been implemented in order to extract facial features classified into linear and nonlinear subspaces. In our research we implemented the linear Principal Component Analysis and the nonlinear Kernel Principal Component Analysis methods. The first step converts face images into a small set of characteristic feature images called "eigenfaces". Projecting a new image into the subspace covered by the eigenfaces, we can classify the face by comparing the positions in face space of known individuals with its position. The classifiers used are the K-Nearest Neighbor with Euclidean distance and Support Vector Machine. This methods have advantages over other face recognition models thanks to their speed and simpleness, learning capacity and relative insensitivity to modest or gradual changes in the face image. The remainder of this paper is organized as follows: section 2 introduces the LFW Face Database. Section 3 explains the preparation's steps of Database. Section 4 analyzes the problem. Section 5 resumes mathematical features of the methods. Section 6 describes the performance and model evaluations. Section 7 shows the results and their interpretation. Section 8 concludes the paper.[2] [3] [4]

2 Data description

The Labeled Faces in the Wild (LFW) database was published in 2007 at University of Massachusetts. Labeled Faces in the Wild (LFW) is a database of face photographs that contains 13'233 image of faces collected from 5749 people; of these, 1680 people have two or more images, and 4069 people have just a single image. Each image has a unique name of the person photographed, such as no name should correspond to more than one person. All of the images have the same 250 by 250 pixel size and the same JPEG format. Some images are provided in greyscale, while others are in color. Some images include more than one face, but it is the face that includes the central pixel of the image which is reputed the identifying face of the image. All of the images have been collected thanks to the detection's process using the Viola-Jones face detector.[5] [6] [7]

3 Data selection and preparation

To compose this database, Gary B. Huang, Manu Ramesh, Tamara Berg, and Erik Learned-Miller followed different steps. First of all, raw images have been gathered by Tamara Berg at Berkeley. Afterwards the Viola-Jones face detector was executed on each of the raw images. In order to clean the database the duplicate face photos have been eliminated and the database's faces labelled manually with a unique name. Finally the detected faces have been cropped, rescaled and saved in the JPEG format. The image arrays have been flatted because One-Dimensional arrays take lower memory than Multi-Dimensional arrays, leading to a reduction of training's time of the models. Importing the database we opted to change the images' color, setting all of them in greyscale. At the same time, we decided to take the people's images that have at least 100 images in the database in order to achieve better performance. We found 5 people with more or equal than 100 images: Colin Powell, Donald Rumsfeld, George W Bush, Gerhard Schroeder and Tony Blair. The new database has 1140 images. Exploring the frequency distribution of images, we observed that the dataset was unbalanced: in detail, 46% of the images portrayed George W.Bush.

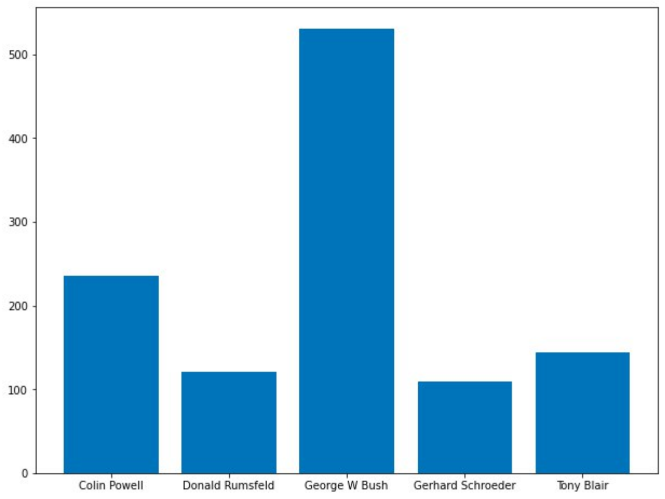


Figure 1: Frequencies distribution with class unbalanced.

In order to avoid problems of overfitting, we decided to rebalance the dataset through data augmentation of images. First of all, we reshaped the existing images creating different copies. In this way we enriched the dataset for all people belonging to the latter. Given that we don't know the distribution of our data and we prefer to use a common scale, we normalized with a min/max normalization the pixel values belonging to the color matrix images. Finally we chose to split the database in train and test set: before the data augmentation we splitted in 80%-20%, while after the data augmentation we used 70% of images to train the models and 30% to test the models.[8] [9]

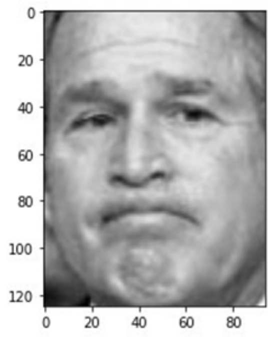


Figure 2: Image of original Dataset.[5]

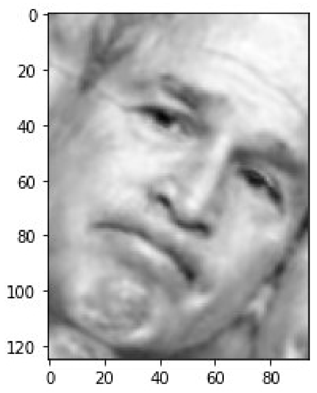


Figure 3: Image reshaped.[5]

4 Problem Analysis

Analyzing the problem, it's needed to explain the difference between face detection and face recognition. Face detection just indicates that an algorithm is able to determine that there is a human face in an image or video. Instead face recognition outlines a biometric technology that goes much further than recognizing when a human face is present. While facial recognition isn't 100% precise, it can very accurately identify when there is a high chance that a person's face matches in another picture of the database. Given that raw face images could consume a long time to be recognized since they are made of a huge quantity of pixels, we applied a dimensionality reduction or feature extraction. Feature extraction relates to converting face spaces into feature spaces, where the face database corresponds to a reduced number of features that hold most of the significant information of the initial faces. In order to achieve this reduction we implemented a common method that uses the linear Karhumen-Loeve transformation, also known as linear Principal Component Analysis. The KLT computes the eigenvectors of the covariance matrix of the input face space. These will define a new face space where the images will be represented.[10] The linear Principal Component Analysis, that is commonly used in the face recognition process, is exploited to create the feature space and extract features. Moreover, we chose to implement also a nonlinear PCA based on kernelization. The Kernel PCA is an expansion of linear PCA to constitute nonlinear representations in a higher-dimensional feature space. The Kernel PCA concerns principal components of variables which are nonlinearly related to the input variables. In the classification step we decided to use the K-Nearest Neighbor classifier with Euclidean distance and Support Vector Machine classifier. Initially, given that classes are unbalanced, we chose to use a cost matrix during the implementation of classifiers. Implementing the K-Nearest Neighbor we set the number of classes K equal to the number of database's people. While implementing the classifiers, we identified the parameters using the Grid Search, an optimization algorithm which chooses the best parameters from a list of options that we provided.[2]

5 Mathematical Description of the proposed models

5.1 Linear PCA

PCA creates a set of orthonormal basis vectors, known as principal components that maximize the dispersion of all projected samples. Let $X = [X_1, X_2, \dots .X_n]$ be the sample set of original images. After normalizing the images to unity norm and subtracting the grand mean a new image set $Y = [Y_1, Y_2, \dots .Y_n]$ is obtained. Each Y constitutes a normalized image with dimensionality N, $Y_i = (y_{i1}, y_{i2}, \dots y_{iN})^t, (i = 1, 2, \dots, n)$. The covariance matrix of the normalized image set is defined as:

$$\sum Y = \frac{1}{n} \sum Y_i Y_j^t = \frac{1}{n} Y Y^t \tag{1}$$

and the eigenvector and eigenvalue matrices U, λ are computes as:

$$\sum Y U = U \lambda \tag{2}$$

Note that YY^t is an N x N matrix while $\mathrm{Y}^t\mathrm{Y}$ is an n x N matrix.[11]

If the sample size n is much smaller than the dimensionality N, then the following process saves some computation:

$$(\mathrm{Y}^{\mathrm{t}}\mathrm{Y})\Psi = \Psi\lambda_1 \tag{3}$$

$$\xi = Y\psi \tag{4}$$

where $\boldsymbol{\lambda}_1 = \operatorname{diag}\{\lambda_1, \lambda_2, \ldots, \lambda_{\mathbf{n}}\}$ and $\xi = [\mathrm{U}_1, \mathrm{U}_2, \ldots, \ldots, \mathrm{U}_{\mathrm{n}}]$.

If one assumes that the eigenvalues are sorted in decreasing order, $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_n$, then the first m leading eigenvectors define matrix P:

$$\mathrm{P} = [\mathrm{U}_1, \mathrm{U}_2, \ldots \ldots, \mathrm{U}_{\mathrm{m}}] \tag{5}$$

The new feature set Z with lower dimensionality $m(m << N)$ is derived as:

$$\mathrm{Z} = \mathrm{P}^{\mathrm{t}}\mathrm{Y} \tag{6}$$

5.2 Kernel PCA

Given a set of zero-mean observations $\mathbf{x}_k, k = 1, \ldots, M$, $\mathbf{x}_k \in R^N$, and $\sum_{k=1}^{M} \mathbf{x}_k = 0$, the covariance matrix is:

$$C = \frac{1}{M}\sum_{j=1}^{M} \mathbf{x}_j \mathbf{x}_j^T \tag{7}$$

PCA aims to find the projection direction that maximizes the variance, which is equivalent to finding the eigenvalue from the covariance matrix:

$$\lambda \mathbf{w} = C\mathbf{w} \tag{8}$$

for eigenvalues $\lambda \geq 0$ and $\mathbf{w} \in R^N$. Since $C\mathbf{w} = \frac{1}{M}\sum_{j=1}^{M} (\mathbf{x}_j \cdot \mathbf{w}) \mathbf{x}_j$, all solutions w with $\lambda \neq 0$ must be in the range of $\mathbf{x}_1, \ldots, \mathbf{x}_M$. Therefore:

$$\lambda (\mathbf{x}_k \cdot \mathbf{w}) = (\mathbf{x}_k \cdot C\mathbf{w}), k = 1, \ldots, M \tag{9}$$

In Kernel PCA, each vector $\mathbf{x}$ is projected from the input space, R^N, to a high dimensional feature space, R^F, by a nonlinear map:

$$\Phi : R^N \rightarrow R^F, F \gg N \tag{10}$$

Note that the dimensionality of the feature space can be arbitrarily large. In R^F, the covariance matrix of $\Phi(\mathbf{x})$ is:

$$C^{\Phi} = \frac{1}{M}\sum_{j=1}^{M} \Phi(\mathbf{x}_j) \Phi(\mathbf{x}_j)^T \tag{11}$$

and the corresponding eigenvalue problem is:

$$\lambda w^{\Phi} = C w^{\Phi} \tag{12}$$

All solutions $\mathbf{w}^{\Phi}$ with $\lambda \neq 0$ lie in the span of $\Phi(\mathbf{x}_1) \ldots, \Phi(\mathbf{x}_M)$.

$$\lambda \left(\Phi(\mathbf{x}_k) \cdot \mathbf{w}^{\Phi}\right) = \left(\Phi(\mathbf{x}_k) \cdot C\mathbf{w}^{\Phi}\right) \quad k = 1, \ldots, M \tag{13}$$

and $\mathbf{w}^{\Phi}$ lie in the span of $\Phi(\mathbf{x}_1), \ldots, \Phi(\mathbf{x}_M)$ such that:

$$\mathbf{w}^{\Phi} = \sum_{i=1}^{M} \alpha_i \Phi(\mathbf{x}_i) \tag{14}$$

Using the equations (13) and (14), we have, for $k = 1, \ldots, M$:

$$\lambda \sum_{i=1}^{M} \alpha_i \left(\Phi\left(\mathbf{x}_k\right) \cdot \Phi\left(\mathbf{x}_i\right)\right) = \frac{1}{M} \sum_{i=1}^{M} \alpha_i \left(\Phi\left(\mathbf{x}_k\right) \cdot \sum_{j=1}^{M} \Phi\left(\mathbf{x}_j\right)\right) \left(\Phi\left(\mathbf{x}_j\right) \cdot \Phi\left(\mathbf{x}_i\right)\right) \tag{15}$$

Defining an $M \times M$ matrix K by:

$$K_{ij} = k\left(\mathbf{x}_i, \mathbf{x}_j\right) = \Phi\left(\mathbf{x}_i\right) \cdot \Phi\left(\mathbf{x}_j\right) \tag{16}$$

We can reformulate the equation (15) as:

$$M\lambda K\alpha = K^2\alpha \tag{17}$$

where α indicates a column vector with entries $\alpha_1, \ldots, \alpha_M$. The solutions of the equation (17) is the same to the following eigenvalue problem:

$$M\lambda\alpha = K\alpha \tag{18}$$

In this paper, we decided to use the polynomial kernel of degree d, that is:

$$k\left(\mathbf{x}_i, \mathbf{x}_j\right) = \left(\mathbf{x}_i \cdot \mathbf{x}_j\right)^d \tag{19}$$

We projected the vectors in R^F to a lower dimensional space spanned by the eigenvectors $\mathbf{w}^\Phi$. Let $\mathbf{x}$ be a test sample whose projection is $\Phi(\mathbf{x})$ in R^F, then the projection of $\Phi(\mathbf{x})$ onto the eigenvectors $\mathbf{w}^\Phi$ are the nonlinear principal components represented by Φ :

$$\mathbf{w}^\Phi \cdot \Phi(\mathbf{x}) = \sum_{i=1}^{M} \alpha_i \left(\Phi\left(\mathbf{x}_i\right) \cdot \Phi(x)\right) = \sum_{i=1}^{M} \alpha_i k\left(\mathbf{x}_i, \mathbf{x}\right) \tag{20}$$

In other words, we can extract the first $q(1 \leq q \leq M\ )$ nonlinear principal components using the kernel function without the expensive operation to expressly project the patterns to a high dimensional space R^F. The first q components is equivalent to the first q nonincreasing eigenvalues of Equation (18). [12]

5.3 K-Nearest Neighbor classifier

The K-Nearest Neighbor classifier $(\mathrm{k-NN})$ is a method for classifying objects by getting the closest k neighbors in the feature space. The K-NN rule classifies the query point z by attributing it the class label C most frequently represented among the k nearest prototypes; such as, by finding the k neighbors with the minimum distances between z and all prototype feature points $\{\mathrm{z_{ci}}, 1 \leq \mathrm{c} \leq \mathrm{C},$ $1 \leq \mathrm{i} \leq \mathrm{n}_c\}$, as follows:

$$\left\{d_1\left(zz_{c_1 i_1}\right), .d_k(zz_{ckik})\right\} = \min_{1 \leq c \leq c, 1 \leq i \leq n_c} d\left(zz_{ci}\right). \tag{21}$$

where $\mathrm{n_c}$ is the number of samples per class. The number of distance computation is $\sum_{c=1}^{C} n_c$ Different parameters are used with $\mathrm{k-NN}$, such as value of k nearest neighbors and distance model. In this paper, the Euclidean norm distance $d\left(zz_{ci}\right) = \|z - z_{ci}\|$ is used and $\mathrm{k} = 1$ to find the class of the closest query point. [10]

5.4 Support Vector Machines

Support Vector Machines method is based on the following idea: the input vectors are mapped into a high-dimension space Z via specific non-linear transformation, in which the optimal hyperlane can be formed. If the dot product in the optimal classification function is replaced with a inner product function (kernel function), the related discriminate function will turn into the following:

$$f(x) = \operatorname{sgn}(\sum_{\mathrm{sv}} y_i a_i K(x_i, x) - b). \tag{22}$$

The discriminate function in the high-dimension space is known as Support Vector Machine.

Different nonlinear discriminate SVM can be formed adopting different kernel function $K(x, x_i)$. The most common kernel functions are:

- Polynomial function
$$K(x, x_i) = [(x \cdot x_i) + 1]^d. \tag{23}$$
The corresponding SVM is a classifier of polynomial of degree d.
- Radial basis function
$$K(x, x_i) = \exp\left\{-\frac{|x - x_i|^2}{\sigma^2}\right\}. \tag{24}$$
The SVM is a Gaussian Radial Basis Function classifier.
- Sigmoid function
$$K(x, x_i) = \tanh(v(x \cdot x_i) + c). \tag{25}$$
The SVM is a multi-layer perceptron.[3]

6 Performance and model evaluation

6.1 PCA and Support Vector Machine

First of all we applied the linear Principal Component Analysis on the initial dataset. In order to optimize feature extraction with Principal Component Analysis, we fine tuned the number of components equal to 200. Before using the Support Vector classifier, the Grid Search algorithm has been executed to discover the best combination of C and gamma parameters. Moreover, the Support Vector's kernel is settled to Radial Basis Function type and the classes' weights are adjusted in order to give weights inversely proportional to class frequencies in the input data to deal with the class imbalance. The relative eigenfaces can be seen in figure 16. The performance of the model on train and test set are resumed in the figures 4 and 5.

6.2 PCA and K-Nearest Neighbor

In this step we changed the Support Vector classifier with the K-Nearest Neighbor classifier. The Grid Search algorithm found the best combination between type of weights and leaf size. The number of clusters is equal to the number of classes. The performance indicators are resumed in the tables 6 and 7.

6.3 Kernel PCA and Support Vector Machine

We implemented the Kernel Principal Component Analysis choosing a polynomial function of second degree. Using the Grid Search algorithm, we found the best combination of needed parameters. The scores of Support Vector classifier are summarized in the tables 8 and 9.

6.4 Kernel PCA and K-Nearest Neighbor

After the feature extraction through Kernel PCA, we classified applying K-Nearest Neighbor classifier. The indicators are written in the tables 10 and 11.

6.5 Data augmentation: Kernel PCA and Support Vector Machine

Given that Kernel Principal Component Analysis works well in high dimensional reductions, we tried to increase the dimensions of dataset in order to achieve better performance indicators. The new scores on Support Vector classifier are resumed in the tables 12 and 13.

6.6 Data augmentation: Kernel PCA and K-Nearest Neighbor

Finally we defined the Kernel Principal Component Analysis with K-Nearest Neighbor classifier on the augmented dataset. The results are in the tables 14 and 15.

7 Results obtained and their interpretation

Observing the results the linear Principal Component Analysis outperforms the Kernel Principal Component Analysis. More in detail, the model PCA-SVM achieves better results than other models, as we can see in figure 17. Applying the Kernel PCA-SVM model, we reach an accuracy equal to 86%, that is a bit lower than the 89% of Linear PCA. Despite of different parameters combinations, the K-Nearest Neighbor classifier performs worst than Support Vector Machine classifier, resulting unsuitable in face recognition problems. Despite of we increased the number of the images through our data augmentation algorithm in order to achieve better results in Kernel Principal Component Analysis, which should perform better with an high number of images for each class, we can observe that both Support Vector Machine and K-Nearest Neighbor classifiers performs worst. In particular, the best combination of parameters found with the Grid Search method, leads to a decrease of performance predicting the test set, indicating that augmenting the number of images rotating or flipping them is not enough for the Kernel Principal Component Analysis and could also lead to misclassification on the test set.

8 Conclusion

In this research, we made a comparison between Principal Component Analysis and Kernel Principal Components Analysis for feature extraction. The performance comparison showed that Eigenface approach performs lightly better than Kernel PCA, due to the fact that Kernel Principal Component Analysis fits better with high dimensional reduction. The Support Vector Machine and the K-Nearest Neighbor classifier has been used to investigate the performance of the Kernel PCA and PCA. In order to improve the results achieved from this approach, a future research could be the exploitation of a larger dataset using Kernel PCA for feature extraction and Support Vector Machine classifier, with more images for each classes or using a different way to increase the dataset. Another future research could be the implementation of a Convolutional Neural Network, one of the most common model technique used in face recognition, also because they perform better and because the data augmentation method that we used is especially suitable for them.

References

[1] Fares Jalled. Face recognition machine vision system using eigenfaces. 2017.

[2] Quan Wang. Kernel principal component analysis and its applications in face recognition and active shape models. 07 2012.

[3] Jianke Li, Baojun Zhao, Hui Zhang, and Jichao Jiao. Face recognition system using svm classifier and feature extraction by pca and lda combination. pages 1–4, 2009.

[4] M.A. Turk and A.P. Pentland. Face recognition using eigenfaces. pages 586–591, 1991.

[5] Gary B. Huang, Manu Ramesh, Tamara Berg, and Erik Learned-Miller. Labeled faces in the wild: A database for studying face recognition in unconstrained environments. Technical Report 07-49, University of Massachusetts, Amherst, October 2007.

[6] Gary B. Huang and Erik Learned-Miller. Labeled faces in the wild: Updates and new reporting procedures. Technical Report UM-CS-2014-003, University of Massachusetts, Amherst, May 2014.

[7] Paul Viola and Michael Jones. Robust real-time object detection. *International Journal of Computer Vision - IJCV*, 57, 01 2001.

[8] Tamara L. Berg, Alexander C. Berg, Jaety Edwards, Michael Maire, Ryan White, Yee Whye Teh, Erik Learned-Miller, and David A. Forsyth. Names and faces in the news. 2004.

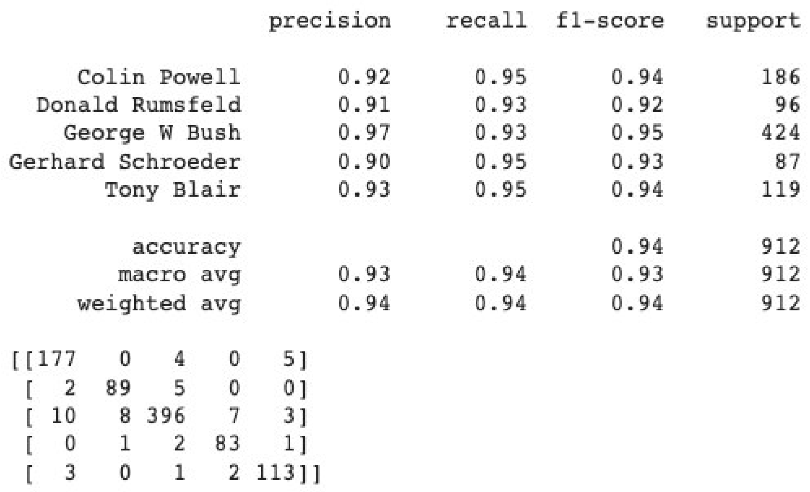
[9] Jamal Shah, Meghdad Sharif, Mudassar Raza, and Aisha Azeem. A survey: Linear and nonlinear pca based face recognition techniques. *iajit*, 11 2013.

[10] Hala Ebied. Feature extraction using pca and kernel-pca for face recognition. *2012 8th International Conference on Informatics and Systems, INFOS 2012*, pages MM–72, 01 2012.

[11] Tahia Fahrin Karim, Molla Shahadat Hossain Lipu, Md. Lushanur Rahman, and Faria Sultana. Face recognition using pca-based method. 3:158–162, 2010.

[12] Ming-Hsuan Yang, N. Ahuja, and David Kriegman. Face recognition using kernel eigenfaces. pages 37 – 40 vol.1, 02 2000.

Appendix



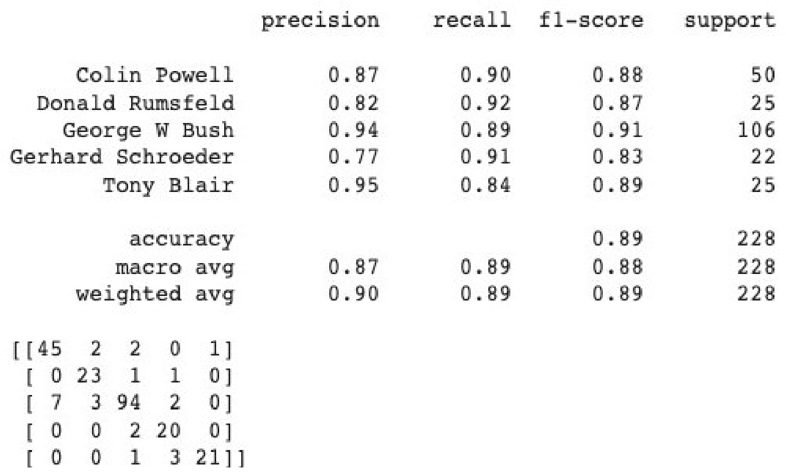
```
                    precision    recall  f1-score   support

      Colin Powell       0.92      0.95      0.94       186
   Donald Rumsfeld       0.91      0.93      0.92        96
     George W Bush       0.97      0.93      0.95       424
 Gerhard Schroeder       0.90      0.95      0.93        87
        Tony Blair       0.93      0.95      0.94       119

          accuracy                           0.94       912
         macro avg       0.93      0.94      0.93       912
      weighted avg       0.94      0.94      0.94       912

[[177   0   4   0   5]
 [  2  89   5   0   0]
 [ 10   8 396   7   3]
 [  0   1   2  83   1]
 [  3   0   1   2 113]]
```

Figure 4: Performance indicators and confusion matrix of PCA-SVM model on train set.



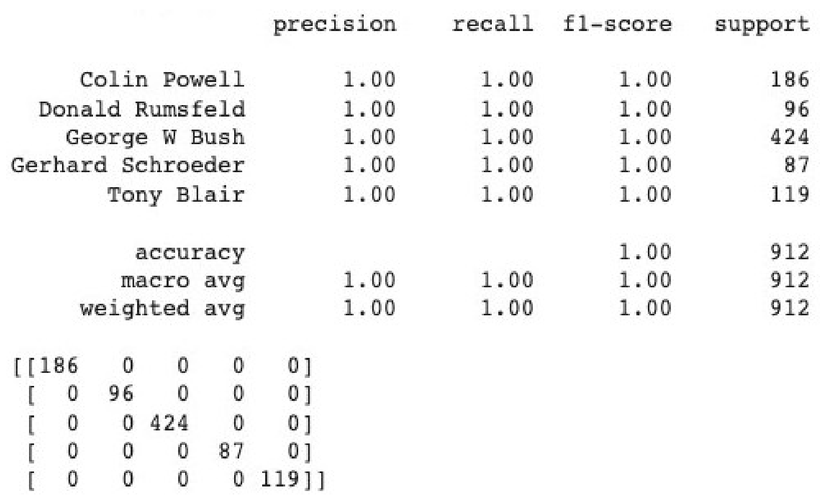
```
                    precision    recall  f1-score   support

      Colin Powell       0.87      0.90      0.88        50
   Donald Rumsfeld       0.82      0.92      0.87        25
     George W Bush       0.94      0.89      0.91       106
 Gerhard Schroeder       0.77      0.91      0.83        22
        Tony Blair       0.95      0.84      0.89        25

          accuracy                           0.89       228
         macro avg       0.87      0.89      0.88       228
      weighted avg       0.90      0.89      0.89       228

[[45  2  2  0  1]
 [ 0 23  1  1  0]
 [ 7  3 94  2  0]
 [ 0  0  2 20  0]
 [ 0  0  1  3 21]]
```

Figure 5: Performance indicators and confusion matrix of PCA-SVM model on test set.



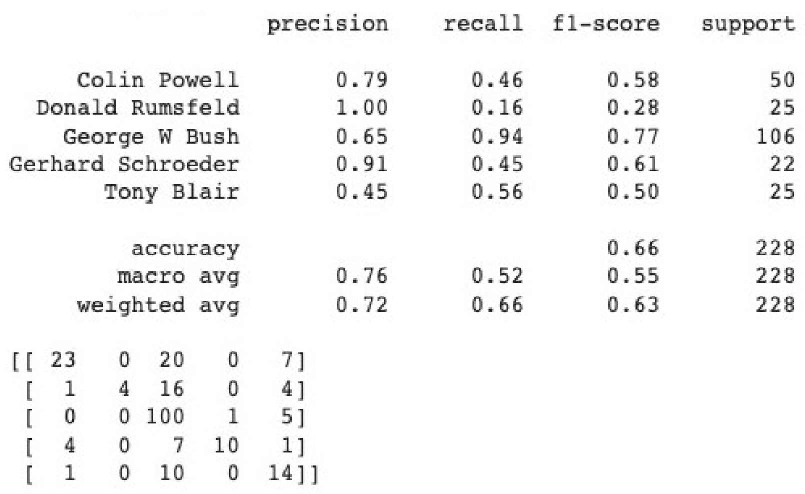
```
                   precision    recall  f1-score   support

      Colin Powell       1.00      1.00      1.00       186
   Donald Rumsfeld       1.00      1.00      1.00        96
     George W Bush       1.00      1.00      1.00       424
 Gerhard Schroeder       1.00      1.00      1.00        87
        Tony Blair       1.00      1.00      1.00       119

          accuracy                           1.00       912
         macro avg       1.00      1.00      1.00       912
      weighted avg       1.00      1.00      1.00       912

[[186   0   0   0   0]
 [  0  96   0   0   0]
 [  0   0 424   0   0]
 [  0   0   0  87   0]
 [  0   0   0   0 119]]
```

Figure 6: Performance indicators and confusion matrix of PCA-KNN model on train set.



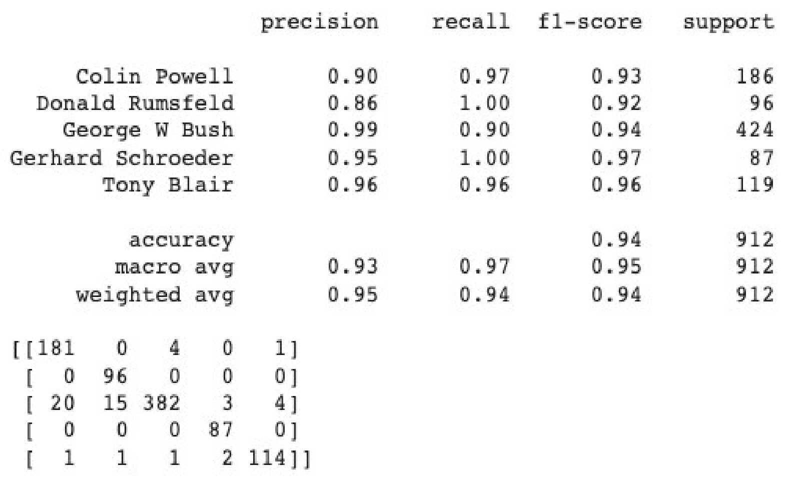
```
                   precision    recall  f1-score   support

      Colin Powell       0.79      0.46      0.58        50
   Donald Rumsfeld       1.00      0.16      0.28        25
     George W Bush       0.65      0.94      0.77       106
 Gerhard Schroeder       0.91      0.45      0.61        22
        Tony Blair       0.45      0.56      0.50        25

          accuracy                           0.66       228
         macro avg       0.76      0.52      0.55       228
      weighted avg       0.72      0.66      0.63       228

[[ 23   0  20   0   7]
 [  1   4  16   0   4]
 [  0   0 100   1   5]
 [  4   0   7  10   1]
 [  1   0  10   0  14]]
```

Figure 7: Performance indicators and confusion matrix of PCA-KNN model on test set.



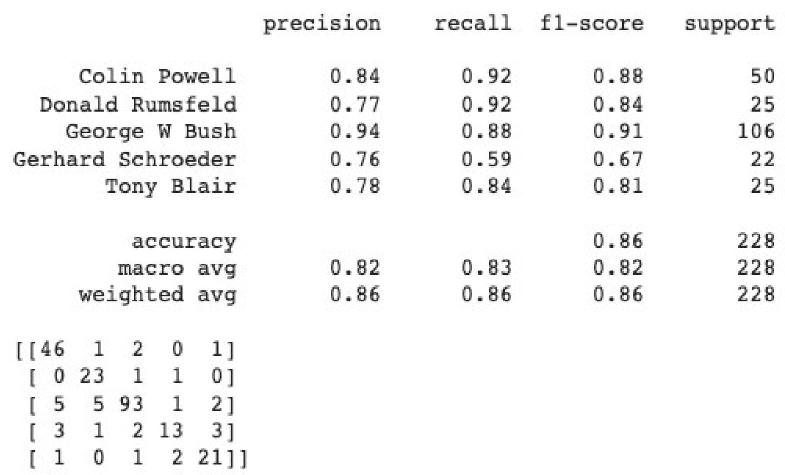
```
                   precision    recall  f1-score   support

      Colin Powell       0.90      0.97      0.93       186
   Donald Rumsfeld       0.86      1.00      0.92        96
     George W Bush       0.99      0.90      0.94       424
 Gerhard Schroeder       0.95      1.00      0.97        87
        Tony Blair       0.96      0.96      0.96       119

          accuracy                           0.94       912
         macro avg       0.93      0.97      0.95       912
      weighted avg       0.95      0.94      0.94       912

[[181   0   4   0   1]
 [  0  96   0   0   0]
 [ 20  15 382   3   4]
 [  0   0   0  87   0]
 [  1   1   1   2 114]]
```

Figure 8: Performance indicators and confusion matrix of Kernel PCA-SVM model on train set.



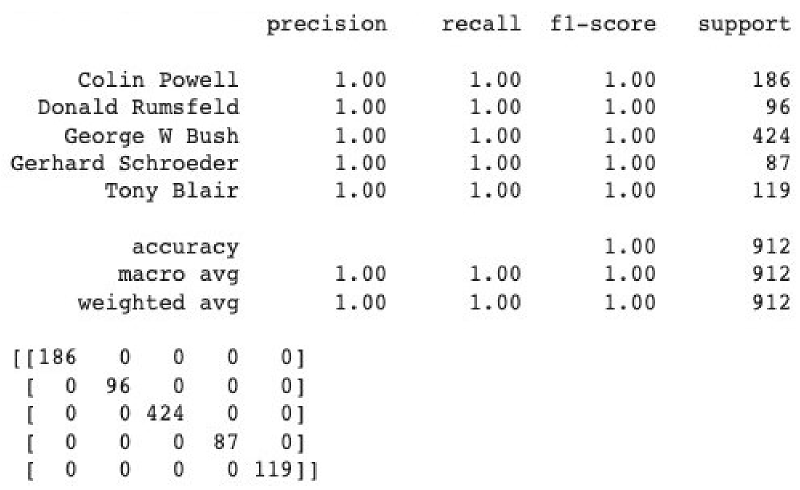
```
                   precision    recall  f1-score   support

     Colin Powell       0.84      0.92      0.88        50
  Donald Rumsfeld       0.77      0.92      0.84        25
    George W Bush       0.94      0.88      0.91       106
Gerhard Schroeder       0.76      0.59      0.67        22
       Tony Blair       0.78      0.84      0.81        25

         accuracy                           0.86       228
        macro avg       0.82      0.83      0.82       228
     weighted avg       0.86      0.86      0.86       228

[[46  1  2  0  1]
 [ 0 23  1  1  0]
 [ 5  5 93  1  2]
 [ 3  1  2 13  3]
 [ 1  0  1  2 21]]
```

Figure 9: Performance indicators and confusion matrix of Kernel PCA-SVM model on test set.



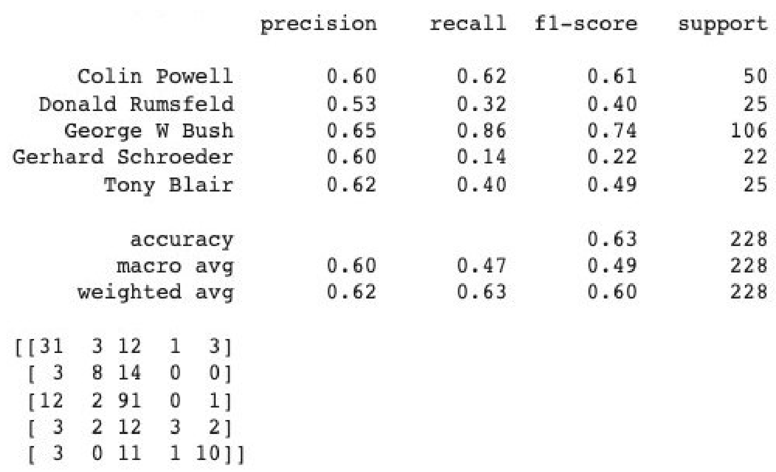
```
                   precision    recall  f1-score   support

     Colin Powell       1.00      1.00      1.00       186
  Donald Rumsfeld       1.00      1.00      1.00        96
    George W Bush       1.00      1.00      1.00       424
Gerhard Schroeder       1.00      1.00      1.00        87
       Tony Blair       1.00      1.00      1.00       119

         accuracy                           1.00       912
        macro avg       1.00      1.00      1.00       912
     weighted avg       1.00      1.00      1.00       912

[[186   0   0   0   0]
 [  0  96   0   0   0]
 [  0   0 424   0   0]
 [  0   0   0  87   0]
 [  0   0   0   0 119]]
```

Figure 10: Performance indicators and confusion matrix of Kernel PCA-KNN model on train set.



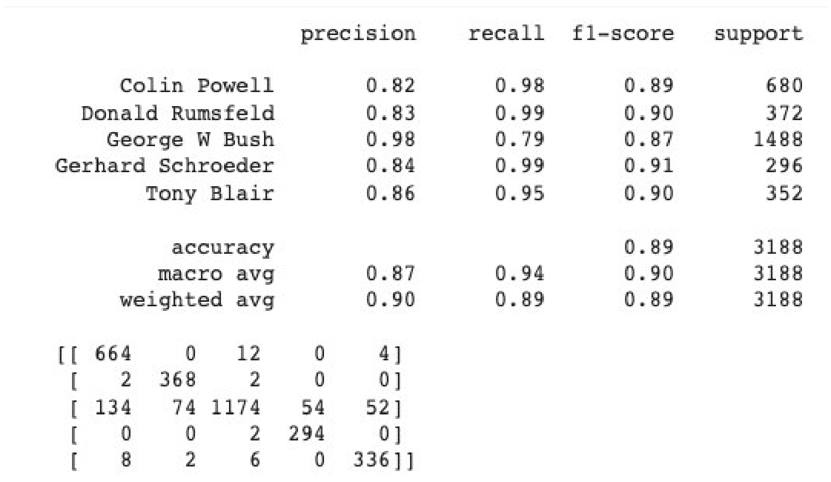
```
                   precision    recall  f1-score   support

     Colin Powell       0.60      0.62      0.61        50
  Donald Rumsfeld       0.53      0.32      0.40        25
    George W Bush       0.65      0.86      0.74       106
Gerhard Schroeder       0.60      0.14      0.22        22
       Tony Blair       0.62      0.40      0.49        25

         accuracy                           0.63       228
        macro avg       0.60      0.47      0.49       228
     weighted avg       0.62      0.63      0.60       228

[[31  3 12  1  3]
 [ 3  8 14  0  0]
 [12  2 91  0  1]
 [ 3  2 12  3  2]
 [ 3  0 11  1 10]]
```

Figure 11: Performance indicators and confusion matrix of Kernel PCA-KNN model on test set.



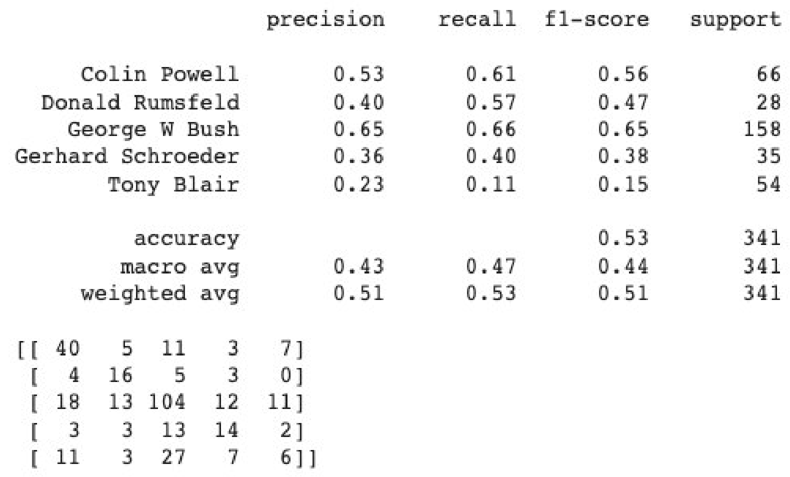
```
                   precision    recall  f1-score   support

      Colin Powell       0.82      0.98      0.89       680
   Donald Rumsfeld       0.83      0.99      0.90       372
     George W Bush       0.98      0.79      0.87      1488
 Gerhard Schroeder       0.84      0.99      0.91       296
        Tony Blair       0.86      0.95      0.90       352

          accuracy                           0.89      3188
         macro avg       0.87      0.94      0.90      3188
      weighted avg       0.90      0.89      0.89      3188

[[ 664    0   12    0    4]
 [   2  368    2    0    0]
 [ 134   74 1174   54   52]
 [   0    0    2  294    0]
 [   8    2    6    0  336]]
```

Figure 12: Performance indicators and confusion matrix of Kernel PCA-SVM model on augmented train set.



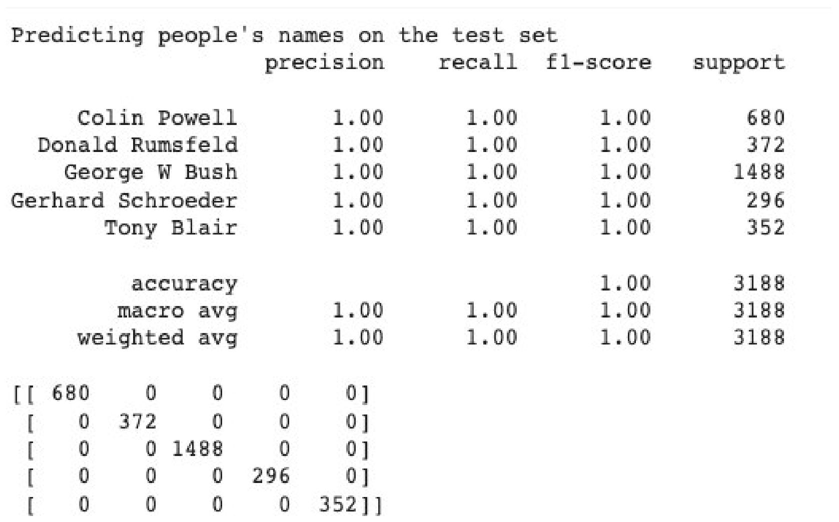
```
                   precision    recall  f1-score   support

      Colin Powell       0.53      0.61      0.56        66
   Donald Rumsfeld       0.40      0.57      0.47        28
     George W Bush       0.65      0.66      0.65       158
 Gerhard Schroeder       0.36      0.40      0.38        35
        Tony Blair       0.23      0.11      0.15        54

          accuracy                           0.53       341
         macro avg       0.43      0.47      0.44       341
      weighted avg       0.51      0.53      0.51       341

[[ 40   5  11   3   7]
 [  4  16   5   3   0]
 [ 18  13 104  12  11]
 [  3   3  13  14   2]
 [ 11   3  27   7   6]]
```

Figure 13: Performance indicators and confusion matrix of Kernel PCA-SVM model on augmented test set.



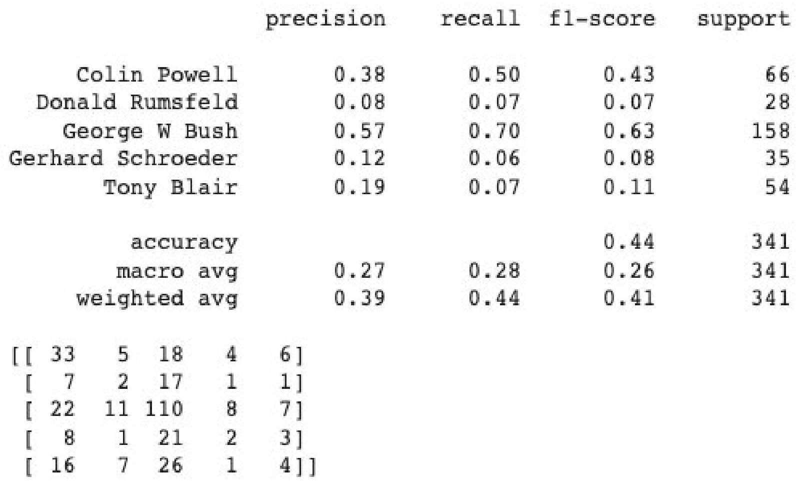
```
Predicting people's names on the test set
                   precision    recall  f1-score   support

     Colin Powell       1.00      1.00      1.00       680
  Donald Rumsfeld       1.00      1.00      1.00       372
    George W Bush       1.00      1.00      1.00      1488
Gerhard Schroeder       1.00      1.00      1.00       296
       Tony Blair       1.00      1.00      1.00       352

         accuracy                           1.00      3188
        macro avg       1.00      1.00      1.00      3188
     weighted avg       1.00      1.00      1.00      3188

[[ 680    0    0    0    0]
 [    0  372    0    0    0]
 [    0    0 1488    0    0]
 [    0    0    0  296    0]
 [    0    0    0    0  352]]
```

Figure 14: Performance indicators and confusion matrix of Kernel PCA-KNN model on augmented train set.



```
                   precision    recall  f1-score   support

     Colin Powell       0.38      0.50      0.43        66
  Donald Rumsfeld       0.08      0.07      0.07        28
    George W Bush       0.57      0.70      0.63       158
Gerhard Schroeder       0.12      0.06      0.08        35
       Tony Blair       0.19      0.07      0.11        54

         accuracy                           0.44       341
        macro avg       0.27      0.28      0.26       341
     weighted avg       0.39      0.44      0.41       341

[[ 33   5  18   4   6]
 [  7   2  17   1   1]
 [ 22  11 110   8   7]
 [  8   1  21   2   3]
 [ 16   7  26   1   4]]
```

Figure 15: Performance indicators and confusion matrix of Kernel PCA-KNN model on augmented test set.

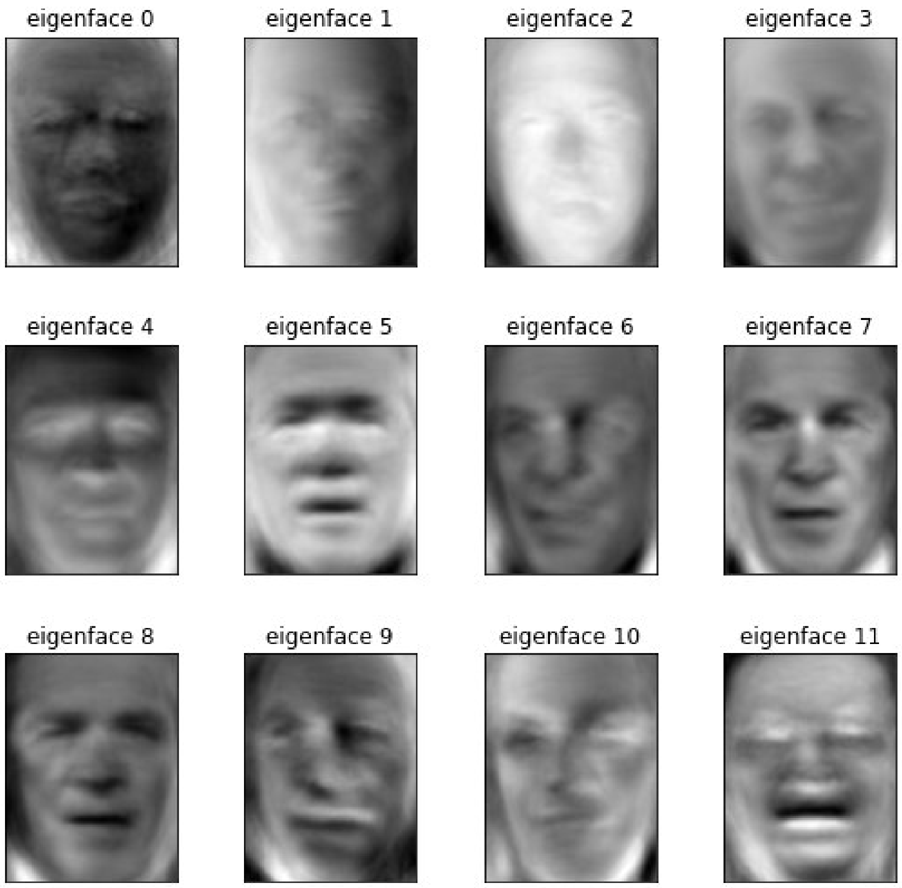


Figure 16: Eigenfaces of PCA-SVM.

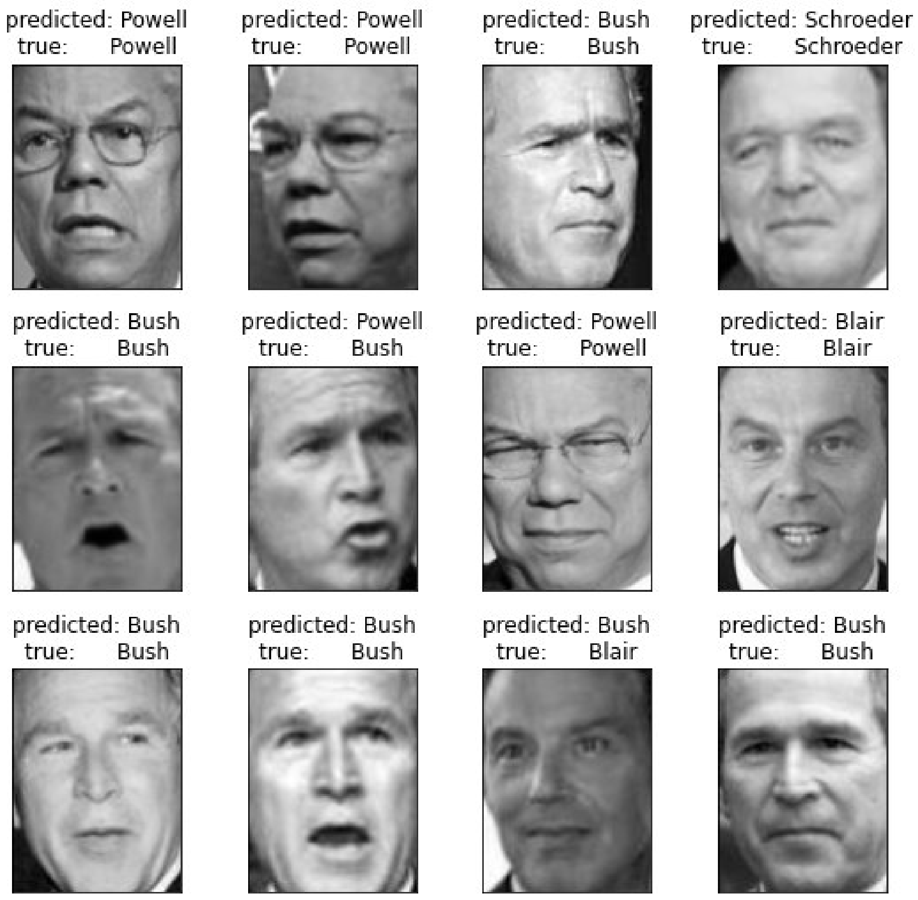


Figure 17: Prediction of PCA-SVM.