INTERNATIONAL JOURNAL OF RESEARCH AND DEVELOPMENT IN ENGINEERING SCIENCES
e-ISSN: 2582 - 4201 Volume: 3 | Issue: 6 | Year: 2021

ONLINE VOTING SYSTEM THROUGH FACIAL RECOGNITION USING FISHERFACES ALGORITHM

Mrs. P. SANGEETA[1], K.KAVYA[2], G.SWATHI[3],

CH.SAI RAM[4], B.HEMA SAI[5]

Assistant Professor[1],CSE, LENDI INSTITUTE OF ENGINEERING AND TECHNOLOGY, A.P., India., Student[2,3,4,5], B.Tech(CSE), LENDI INSTITUTE OF ENGINEERING AND TECHNOLOGY.

ABSTRACT

A new authentication technique in an online voting system using facial recognition of the voter is used. Currently, there are two types of voting systems. They are secret Ballot paper and Electronic Voting Machines (EVM), but both processes have some limitations or demerits. Till now online voting is not yet implemented in India. The present voting system is not safe and secure. The voter who is not eligible can also cast his/her vote by fake means, leading to many problems. So, in this project we have proposed a system or way for voting which is very effective or useful in voting. We have a security level in the voting process that is face recognition by using the FisherFace algorithm.

KEYWORDS — Facial Recognition, Eigen Face, Fisher Face and Surf

I. INTRODUCTION

In India, presently, we are having two kinds of voting mechanisms, first the secret ballot paper and the second one is Electronic Voting Machines (EVM), but the process of voting has some demerits and disadvantages, so current system lacks in safety and security. In our proposed system, we are introducing three levels of verification which is very effective in reducing the false voting scenarios. The first includes the unique ID generation at the registration which would be given to the voter. After which, in the second level of safety when ID is given to the Election Commission Officer, it would be crosschecked by the officer and now the new tier of verification through which the voter needs to go, will greatly enhance the security, here we would be matching the current facial features of a voter with the one present in the database, this would reduce the chances of the fake casting of vote and make the system safer and error-free. In this paper, we will discuss the different types of algorithms used in the field of facial recognition. Along with this, we will also make a comparison between these algorithms. We have also measured the accuracy of these algorithms by practically implementing and evaluating them on the test set. The paper is arranged in IV sections. Section I contains the introduction to the different types of voting mechanisms. Section II contains the Literature Survey which displays the recent work done in this domain. Section III contains the Data Analysis section for our proposed method. It consists of a related study of different types of algorithms used for facial recognition. Section IV has the main conclusion part of our complete research.

II. RELATED WORK

The previous work done in this domain involves reviewing the existing algorithms and comparison of these algorithms based on various features and constraints such as type of database used, and the neural network-based image processing system used for the identification of the facial features [1,2]. The amount of distortion and attenuation plays a big role in generating a clear and transparent image in a localized area of the image frequency as it would be an important aspect while capturing the image and processing it to reduce errors, match it with one that is present in the database[3].

Vigorous technique for naturally coordinating highlights in pictures compared to the equivalent physical point on an item observed from two discretionary perspectives. Unlike standard stereo, matching approaches coordinating methodologies, the presumption

like no earlier information about the relative camera positions and directions. Actually, in this application this is the data wish to decide from the picture attribute matches. Highlights are modified in two or more pictures and portrayed utilizing affine texture invariants.

A robust 4-layer Convolutional Neural Network (CNN), engineering is proposed for the face acknowledgment issue, with an answer that is provided for dealing with facial pictures that contain occlusions, poses, facial expressions. There are many face recognition algorithms, just a set of them meet the sequential limitations of a software-based arrangement without utilizing any committed hardware machine. This paper presents a real-time and strong solution for mobile platforms [6], which in general has limited computation and memory resources as compared to PC platforms. This solution includes grouping two previous real-time implementations for mobile platforms to address the shortcoming of each implementation. The leading execution gives an on the web or on-the-fly light source adjustment for the second usage which is seen as robust to various face postures or orientations. Pattern classification technique by considering every pixel in an image as a coordinate in a high-dimensional space is discussed in [7]. Along with the upside of the perception that the images of a particular face, under fluctuating illumination but fixed pose, lie in a 3D linear subspace of the high dimensional image space if the face is a Lambertian surface without shadowing. In any case, since faces are not genuinely Lambertian surfaces and in fact produce 5 self-shadowing; images will go deviate from this linear subspace. As opposed to explicitly projecting this divergence, linearly project the image into a subspace in a way that limits those regions of the face with huge deviation. Design and execution of the component extraction strategy for Speeded-Up Robust Features (SURF) and Support Vector Machine (SVM) classification technique into the traffic signs recognition application is deliberated [9]. The grant of this application is the importance of the traffic sign with two languages, Indonesia and English. In the SURF strategy, the littlest huge number of key focuses will affect the accuracy level to perceive a picture. Face detection is the property of all the face-processing systems, while in the video the face detection issue has more special importance. By examining the face detection dependent on the Adaboost algorithm, this paper presents a quick and good robust face detection method.

EXISTING SYSTEM AND LIMITATIONS

- In the current voting system, the ballot machines are used in which the symbols of various political parties are displayed.
- When we press the button representing the particular party's symbol the voting is done. The chance of a dummy person casting their vote is more in the already existing system.
- Chances of duplicate voting are more because if the authorized person is not honest as he/she is required to be so, he/she might perform the illegal task of voting for a particular party person.
- So, the facial authentication process is used for detecting the right person and also helps to work online, which will help the voters to cast their vote from their place by and of itself.

III. PROPOSED SYSTEM

In this project, we will use the Fisherface algorithm. We will train the machine with a dataset that enables us to detect whether the face is matching with the current person image, with data present in the dataset using the Fisherface algorithm. To detect fraud in voting, we will use Fingerprint authentication and Facial Recognition to improve election results.

A. ADVANTAGES OF PROPOSED SYSTEM

- Transparency to all citizens as single device is used.
- Non-violence is supported as this process is not place oriented.
- No waiting time.

- Time saving, since voting is through online mode.
- Cost-effective and secure.
- High data storage capacity.
- Easily accessible and stable.

B. DATA ANALYSIS

1) EIGENFACES ALGORITHM: Eigenfaces, as its name suggests, involve the use of Eigenvectors for performing facial recognition. The Eigenface approach is repeatedly used for the recognition of faces from the images. The base segments for the recognition process involve the creation of an Eigenface basis and recognition of a new face. The Eigenface approach classifies faces based on general facial patterns [2]. These patterns involve various aspects of the face [shown in figure 1.] based on the training set images. Eigenface system requires training on a dataset of known faces where all images are of the same size and pixels, along with other attributes like grayscale, with values ranging from 0 to 255.

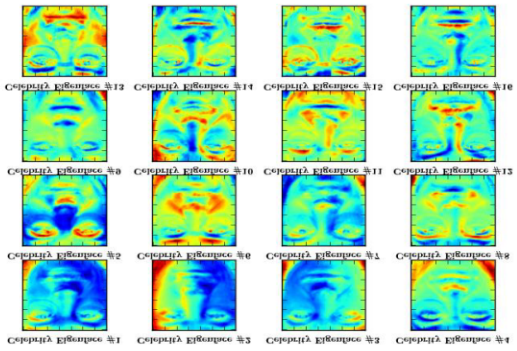

Fig. 1 Grayscale of faces using Fisherface agorithm

Algorithm steps are as follows [4]:

1. For every new independent, the training dataset is formed by apprehending images from the different perspectives of viewing angles to completely recognize every facial feature.

2. After which the images in the training set are recompiled and rebuild so that they have same dimensions and size in pixels using the grayscale feature available in OpenCV.

3. In order to enhance the features, we accelerate the contrast of the image, which becomes another step of feature enhancement.

4. These then are used together to form an Eigenvector.

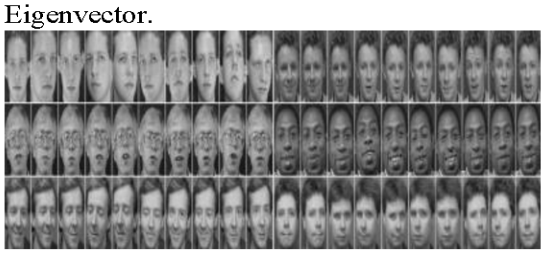

Fig. 2 Sample data for faces

Eigenfaces technique uses Principal Component Analysis (PCA) for finding Eigenvalues. Principal Component Analysis is a method for dimensionality reduction and creates Eigenspace from the data by removing and merging its common attributes. This Eigenspace then consists of the Eigenvectors which help in the mathematical representation of the various pattern. These patterns are a result of the feature matrix that has been discovered from the images in the training dataset. So, as a result, the Principal Component Analysis forms the Eigenspace that contains Eigenvalues, which helps in maximizing the variance. The output of the Eigenface system is the extracted based on result of the individual's face which can be used to match the voter's identity. The voter will be asked to enter their ID number which will be used to fetch their image from the database [5].

Eigenface has some of the cons which are described as follows:

- Scale-sensitive means it requires some pre-processing of an image before identification.
- Its throughput reduces under varying pose and illumination conditions.
- It cannot be used in cases of extreme pose variations as well articulation.
- It cannot maximize the ratio between-class scatter to within-class scatter.

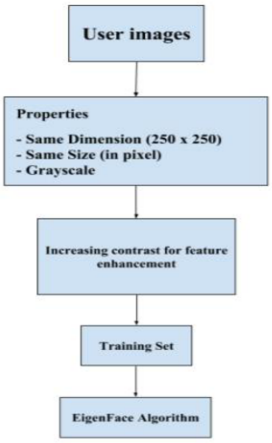

Fig. 3 The training procedure for Eigenface technique.

2) FISHER FACES ALGORITHM:

Fisher face algorithm steps are as follows:

1. Formation of the training dataset. The training dataset is created in a similar way as created in the Eigenface technique.
2. The Fisher face technique then follows its feature retrieving process using PCA and LDA.
3. PCA retrieves Eigenvector from the training image data.
4. LDA helps in finding the directions that are used for classification with the use of both, PCA and LDA time complexity reduces. This means that the image processing becomes fast, and this results in increase in the speed of facial recognition in images with more efficiency and better results. Fisher faces make use of Fisher Linear Discriminant (FLD) to reduce dimensionality. So, FLD is the best way to reduce dimensionality among various classes that classify from each other.

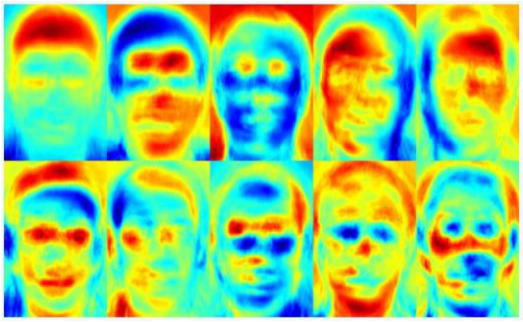

Fig. 4 Identification of faces using Fisherface agorithm

C. PRECISION VALUES

factor	name	1	2	3	4	5	6	7	8	9	10	...	91
f1-score	Gerhard Schroeder	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.235294	...	0.754717
	Colin Powell	0.000000	0.000000	0.197802	0.327869	0.270833	0.376068	0.347826	0.455172	0.479261	0.577181	...	0.874172
	weighted avg	0.287607	0.288870	0.334238	0.357985	0.347321	0.387414	0.383976	0.458691	0.428880	0.524970	...	0.840676
	macro avg	0.125301	0.125852	0.166531	0.190929	0.179953	0.208547	0.204063	0.268808	0.249035	0.394605	...	0.806085
	Donald Rumsfeld	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.210526	...	0.764706
	Tony Blair	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.281250	0.060000	0.240000	...	0.753247
	George W Bush	0.626506	0.629259	0.634855	0.625277	0.628931	0.666667	0.672489	0.707617	0.686916	0.710027	...	0.863582
precision	Colin Powell	0.000000	0.000000	0.529412	0.416667	0.590909	0.511628	0.487805	0.464789	0.515625	0.573333	...	0.857143
	Gerhard Schroeder	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.800000	...	0.933333
	George W Bush	0.457478	0.459064	0.470768	0.479592	0.468750	0.508361	0.511628	0.576609	0.542435	0.617925	...	0.831461
	Donald Rumsfeld	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.333333	...	0.896552
	weighted avg	0.210012	0.210740	0.330864	0.310319	0.343044	0.344074	0.340419	0.418874	0.396504	0.548931	...	0.847300
	macro avg	0.091496	0.091813	0.200036	0.179252	0.211932	0.203998	0.199887	0.293872	0.258755	0.521168	...	0.854286
	Tony Blair	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.428571	0.285714	0.281250	...	0.852941
recall	Tony Blair	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.209302	0.046512	0.209302	...	0.674419
	macro avg	0.199726	0.200000	0.219229	0.233672	0.226218	0.253090	0.250232	0.314489	0.285753	0.383311	...	0.773061
recall	Tony Blair	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.209302	0.046512	0.209302	...	0.674419
	macro avg	0.199726	0.200000	0.219229	0.233672	0.226218	0.253090	0.250232	0.314489	0.285753	0.383311	...	0.773061
	weighted avg	0.456140	0.459054	0.473684	0.470760	0.476608	0.508772	0.508772	0.543860	0.532164	0.564327	...	0.845029
	Gerhard Schroeder	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.137931	...	0.689655
	George W Bush	0.969631	1.000000	0.974522	0.898089	0.955414	0.968153	0.980892	0.917197	0.936306	0.834395	...	0.942675
	Donald Rumsfeld	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.153846	...	0.666667
	Colin Powell	0.000000	0.000000	0.121622	0.270270	0.175676	0.297297	0.270270	0.445946	0.445946	0.581081	...	0.891892
support	George W Bush	157.000000	157.000000	157.000000	157.000000	157.000000	157.000000	157.000000	157.000000	157.000000	157.000000	...	157.000000
	Tony Blair	43.000000	43.000000	43.000000	43.000000	43.000000	43.000000	43.000000	43.000000	43.000000	43.000000	...	43.000000
	Donald Rumsfeld	39.000000	39.000000	39.000000	39.000000	39.000000	39.000000	39.000000	39.000000	39.000000	39.000000	...	39.000000
	macro avg	342.000000	342.000000	342.000000	342.000000	342.000000	342.000000	342.000000	342.000000	342.000000	342.000000	...	342.000000
	Colin Powell	74.000000	74.000000	74.000000	74.000000	74.000000	74.000000	74.000000	74.000000	74.000000	74.000000	...	74.000000
	Gerhard Schroeder	29.000000	29.000000	29.000000	29.000000	29.000000	29.000000	29.000000	29.000000	29.000000	29.000000	...	29.000000
	weighted avg	342.000000	342.000000	342.000000	342.000000	342.000000	342.000000	342.000000	342.000000	342.000000	342.000000	...	342.000000

Fig. 5 Precision Values

Based on the precision values, by comparing both the fisher face and eigen faces algorithm the Fisher Face algorithm gives more accurate data than eigenfaces and hence Fisher face algorithm is more efficient.

IV. IMPLEMENTATION

Fisher Face implementation is as follows:

```
import matplotlib.pyplot as plt

from sklearn.model_selection import
train_test_split

from sklearn.datasets import fetch_lfw_people

from sklearn.metrics import
classification_report

from sklearn.decomposition import PCA
```

```
from sklearn.neural_network import
MLPClassifier

from sklearn.discriminant_analysis import
LinearDiscriminantAnalysis as LDA


# Load data

lfw_dataset =
fetch_lfw_people(min_faces_per_person=100)

 _, h, w = lfw_dataset.images.shape

X = lfw_dataset.data

y = lfw_dataset.target

target_names = lfw_dataset.target_names


# split into a training and testing set

X_train, X_test, y_train, y_test =
train_test_split(X, y, test_size=0.3)

for i in range(1, 101):

    # Compute a PCA

    n_components = i

    pca = PCA(n_components=n_components,
whiten=True).fit(X_train)


    # apply PCA transformation

    X_train_pca = pca.transform(X_train)

    X_test_pca = pca.transform(X_test)

    sklearn_lda = LDA(n_components=4)

    X_train_pca_lda =
sklearn_lda.fit_transform(X_train_pca,
y_train)

    X_test_pca_lda =
sklearn_lda.fit_transform(X_test_pca, y_test)


    # train a neural network

    print("Fitting the classifier to the training
set")

    clf =
MLPClassifier(hidden_layer_sizes=(1024,),
batch_size=256, verbose=True,
early_stopping=True).fit(X_train_pca_lda,
y_train)

    y_pred = clf.predict(X_test_pca_lda)

    print("for ncomponent {}".format(i))

    report = classification_report(y_test, y_pred,
target_names=target_names,
output_dict=True)

    reports[i] = report

    print("For ncomponebt
{}".format(n_components))

    print(classification_report(y_test, y_pred,
target_names=target_names))

ef_accuracy = {}

plt_names = ['Colin Powell', 'George W Bush',
'Tony Blair', 'Donald Rumsfeld', 'Gerhard
Schroeder', 'macro avg', 'weighted avg']

comparision_factor = ['precision']

ef_cr = pd.DataFrame()

for report in reports:

    for key in reports[report]:

        if key == 'accuracy':

            ef_accuracy[report] =
reports[report][key]

        else:

            for inner_key in reports[report][key]:

                index = key+'_'+inner_key

                ef_cr.at[index, 'factor'] = inner_key

                ef_cr.at[index, 'name'] = key

                ef_cr.at[index, report] =
reports[report][key][inner_key]

ef_cr = ef_cr.sort_values('factor')

multi = ef_cr.set_index(['factor', 'name'])

multi

import matplotlib.pyplot as plt
```

```
import numpy as np

fig, axs = plt.subplots(4, 2, figsize=(15, 15))

plt_names = ['Colin Powell', 'George W Bush',
'Tony Blair', 'Donald Rumsfeld', 'Gerhard
Schroeder', 'macro avg', 'weighted avg']

comparision_factor = 'precision'

for i in range(len(axs)):

    for j in range(len(axs[i])):

        if i*2+j != 7:

            axs[i,
j].title.set_text(comparision_factor + ' for ' +
plt_names[i*2+j])

            x = range(1, 101)

            y = multi.loc[(comparision_factor,
plt_names[i*2+j]), :]

            axs[i, j].plot(x, y)

        else:

            axs[i, j].title.set_text(' Acuracy ')

            x = range(1, 101)

            y = [ef_accuracy[i] for i in
sorted(ef_accuracy)]

            axs[i, j].plot(x, y)
```

V. CONCLUSION

Face recognition has been since its advent a more stable and trustworthy form of authentication by including this feature with our present voting system we could increase the abilities of the system and can make it more secure and free from false voting. In this paper, we have delivered a comparative study based on the properties of the three types of algorithms, i.e., Eigenfaces, SURF (Speeded Up Robust Features), and Fisherfaces. Along with these, we have also analysed their accomplishment based on how they classify faces in the images. Our training set consists of 2316 images. The images in the training set were developed for further enhancement of their features. Each build-up set constituted of 4 more samples per image. So, the entire set constituted of 2316*4, that is, 9264 images. On the basis of our research, we observed that the accuracy of the algorithms based on the training data came out to be, 77% for the Eigenface algorithm, 80% for the Fisher face algorithm, and 88% for the SURF algorithm. The training data consists of 2316 labelled images. Apart from this, we observed another summing-up that the SURF algorithm only gives higher precision when the image has some common features in comparison to the training data. In future work, we plan on enlarging the training dataset and applying other important techniques like SIFT, deep learning neural network, etc.

REFERENCES

[1] Choudhary, N., Agarwal, S., & Lavania, G. (2019). Smart Voting System through Facial Recognition. International Journal of Scientific Research in Computer Science and Engineering, 7(2), 7-10.

[2] F. Song, D. Zhang, J. Wang, H. Liu, and Q. Tao, "A parameterized direct LDA and its application to face recognition," Neurocomputing, Vol.71, 2007.

[3] A. Baumberg, “Reliable feature matching across widely separated views”, in CVPR, 2000

[4] H. Bay, T. Tuytelaars, L. Van Gool, “SURF: speeded up robust features”, in ECCV, 2006.

[5] KashifHussainMemon, Dileep Kumar and Syed Muhammad Usman, “Next Generation A Secure EVoting System Based On Biometric Fingerprint Method”, 2011.

[6] P. Belhumeur, J. Hespanha, and D. Kriegman, “Eigenfaces vs. Fisherfaces: Recognition Using Class Specific Linear Projection”, IEEE Transactions on Pattern Analysis and Machine Intelligence, 19(7):711--720, 1997.

[7] P. Viola and M. J. Jones, "Robust real-time face detection," International Journal of Computer Vision, Vol. 57, pp. 137-154, 2004.

[8] Srikrishnaswetha, K., Kumar, S., & Mahmood, M. R. (2019). A study on smart electronics voting machine using face recognition and aadhar verification with iot. In

Innovations in electronics and communication engineering (pp. 87-95). Springer, Singapore.

[9]Patel, T., Chokshi, M., & Shah, N. (2013). Smart device-based election voting system endorsed through face recognition. International Journal of advance research in computer science and software engineering, 3(11).