I.J. Information Engineering and Electronic Business, 2022, 3, 30-40
Published Online June 2022 in MECS (http://www.mecs-press.org/)
DOI: 10.5815/ijieeb.2022.03.04

A Comparative Study of Eigenface and Fisherface Algorithms Based on OpenCV and Sci-kit Libraries Implementations

Ismail Aliyu, Muhammad Ali Bomoi and Maryam Maishanu
Department of Mathematical Sciences Abubakar Tafawa Balewa University Bauchi, Nigeria
Email: ialiyu@atbu.edu.ng, mabomoi.ug@atbu.edu.ng, maryammaishanu@yahoo.com

Received: 15 February 2022; Accepted: 25 March 2022; Published: 08 June 2022

Abstract: Facial Recognition is the task of processing an image or video content in order to identify and recognize the faces of individuals. Its area of applications are wide and a lot of research efforts have been invested which led to introduction of techniques/algorithms and programming language libraries for implementation of those techniques. Facial recognition relies heavily on the use of machine learning techniques. Convolutional Neural Network (CNN), a deep learning algorithm has been successfully applied for face recognition task. However, because of its requirements, it may not be applicable in all cases. Where application scenario cannot cope with CNN, it is necessary to resort to other techniques that use traditional Machine Learning (ML) techniques. Previous studies that performed comparison on face recognition algorithms that use traditional ML techniques only disclosed the best algorithm without revealing the best image processing library used. Considering the fact that people now depend on these libraries to build face recognition systems, it is important to empirically show the best library. In this paper an experiment was conducted with aim of assessing the performance of Fisherface and Eigenface algorithms, and that of Scikit-learn and OpenCV libraries. Eigenface and Fisherface algorithms were combined with K-Nearest Neighbors (KNN) and Support Vector Machines (SVM) classifiers respectively. The algorithms were evaluated using LFW dataset, and implemented in two Python libraries for image processing Scikit-learn and OpenCV. This is to enable us determine the best performing technique/algorithm and at the same time the best library, thereby achieving dual aims. Experimental results show that Scikit-learn implementation of Fisherface with KNN recorded the highest F-score of 67.23% while the OpenCV implementation of Eigenface with SVM recorded the lowest F-score of 14.53%. Comparing the algorithms, Fisherface with SVM produced better results than Eigenface with SVM. The same story holds for Fisherface with KNN, and Eigenface with KNN. This suggests that irrespective of classifier, Fisherface outperform Eigenface in terms of accuracy of recognition. Comparing the libraries, Scikit-learn implementations of Fisherface with SVM and Eigenface with SVM, outperform the OpenCV implementation of the same algorithms. This means scikit-learn implementation produces better results than its counterpart, the OpenCV.

Index Terms: Face Recognition, Eigenface, Fisherface, OpenCV, Sci-kit learn

1. Introduction

Over time, Computer vision has grown to be a matured area of research in Computer Science, with image and video processing as its sub-fields. *Computer Vision is concern with enabling computer to derive meaningful information from digital images, video and other visual input.* Computer vision has variety of applications such as; Security [1,2] manufacturing [3], construction [4,5] etc. [6] in their book provides an overview of computer vision along with its applications. Today, Computer Vision related tasks today heavily rely on the use of machine learning techniques. Recently Deep Learning algorithm such as Convolutional Neural Network (CNN) is widely used in the field of computer vision. For example facial expression recognition [8]. Recognition of ethnicity from facial images [9]. [10, 11] reviewed facial emotion recognition using deep learning.

Face recognition is one of the image processing tasks. It is a task of identifying and recognising the face of individual in a given image or video [7]. Face recognition problem goes beyond detecting the presence of human face. Its applications are many. It is an important biometric technique with numerous applications. Typical examples includes unlocking of phones, used by security agencies for crime detection, used by social networking sites etc. For instance, when picture is uploaded onto Facebook, the faces of people in the picture are identified automatically and their names suggest to the user for tagging. Algorithms for different image processing tasks have been proposed over time. Libraries

that permit different image processing task to be accomplished have been developed as well. All these are the results of previous research efforts which the present generation is now enjoying.

The use of neural networks with several layers of neurons, also refer to as Deep Learning have been successful in recent past. Researches have shown that Convolutional Neural Network (CNN), which is a Deep learning algorithm outperform other techniques in terms of accuracy. However, CNN has two issues that limits its applicability; one, it requires powerful machine with enormous processing capacity, second, it also requires huge dataset for model training. Where application scenario could not cope with CNN, it is necessary to resort to other techniques that use traditional Machine Learning (ML) techniques. In addition to deep learning, variety of algorithm for facial recognition exists and at the same time libraries that provide implementation of those algorithms also exist. EigenFace and FisherFace are examples of facial recognition algorithms that use traditional supervised machine learning classifiers. OpenCV and Scikit-learn are examples of image processing libraries.

Students and practitioners alike that come across image processing task, facial recognition in particular, are usually faced with the *problem of deciding which library to use?* Although several factors may influence the choice of a library for a given task, it is important to make answer to such question available to research communities through empirical means rather than relying on bloggers and community of users who most often state their opinions without proof. Previous studies in the literature that compare face recognition algorithms. For instance, [30, 31, 33, 34] compared different face recognition algorithms. However, those studies were limited to only comparison of the algorithms but not the libraries used. Our aim in this paper is to determine the image processing library that provides better implementation of face recognition algorithms. To this end, we implement EigenFace and FisherFace algorithms using two different libraries; scikit-learn and openCV, with a view to achieve the dual aims of determining (1) the algorithm that perform best (2) the library that provides better implementation. It is worthy to note that our intention is not to market any library rather we are driven by the desire to make the knowledge readily available to researchers particularly the new ones. Our contributions are as follows:

i. We reviewed the literature and present taxonomy of facial recognition techniques and explain how deep learning techniques found their way into the area of image processing particularly the facial recognition task.
ii. We implement two different facial recognition algorithms with combination of different classifiers. The techniques were implemented using two different image processing libraries using the same dataset in order to access the performances of both the techniques and the libraries.
iii. We used additional metrics in our evaluation in order to investigate any variation with accuracy, which is the only metric used by most research papers. This is enable use determine whether accuracy metric is sufficient enough to evaluated facial recognition system.

This remaining content of this paper is organized as follows; Section 2 briefly presents the face recognition, and briefly explain the algorithms used in this paper. The connection of deep learning with face recognition task is also explained. Section 3 presents related works. Methodology is presented in section 4. Experimental settings, dataset used, results and discussion of results are the content of section 5. Section 6 concludes the paper.

2. Face Recognition

2.1 Face Recognition Process

Face recognition process is a multi-steps task. It goes beyond the trivial problem detecting human face in a given picture. [7] stated that face recognition problem can refer to any of the following two problems; (1) Given two pictures containing face; decide if the face in both pictures is that of the same individual. (2) Given a picture containing face(s), decide if the face is that of a particular individual. This of course involves among other things comparing the face in the picture with those in a particular database of faces. This paper is in line with the second problem. This focus of this paper is not to present details of face recognition process and techniques. We refer reader to survey papers [12,13,14]. In this section, we briefly present face recognition process and dwell on the two algorithms that inform the theme of this paper. Face recognition process involves 3 basic steps.

1) Face detection: this involves locating the region of the input image where the face is. Given an input image, a face recognition system can conclude whether the picture is that of humans or not right from this step.
2) Feature extraction – having located the face, this step extracts features of the face (such as nose, eyes, mouth with their geometric distributions) and represents them in a feature vector. Because each face has its own unique features, the next step depends on the features extracted in this step. Eigenface [15], Linear Discriminant Analysis (LDA) [16], [17], Scale-invariant feature transform (SIFT) [18], Local Binary Pattern (LBP) [19,20] techniques are widely used for this task.
3) Face recognition – this takes the features extracted in the preceding step and run comparison check of test face against the database of known faces in order to find a matching face. Now that we have machine learning techniques at our disposal, machine learning algorithms are the most suitable to handle this task. Therefore,

this step involves training a classifier to decide whether the test face is the one that exist in the database of face or not. K-Nearest Neighbour (KNN) [21], Support Vector Machine (SVM), deep neural network such as Convolutional Neural Network (CNN) [22] are examples of classifiers known to address this task.

2.2 Classification of Face Recognition Approaches

Based on the approach employed for their detection and recognition, face recognition systems are classified into 3; Local, Holistic, and Hybrid [12].

i. The local approach considers only some facial features (such as nose, eyes), without considering the entire face.
ii. In holistic approach, the whole face is taken as input data and it is projected into a latent space or in correlation plane.
iii. The hybrid approach combines local and global feature in order to achieve better recognition accuracy. For details of state of the art techniques or methods in each of the mentioned categories, see [12].

As mention earlier, holistic or subspace approach process the entire face and represent the face region by matrix of pixels and this matrix is converted into feature vector to facilitate processing. The subspace approach has advantage of being sensitive to variations (facial expressions, illuminations and poses) and this why it is widely used. Holistic or subspace approach are further categorized into two classes; linear and non-linear techniques. This categorization is based on the method used to represent the subspace. Eigenface [15,23] and Fisherface [24] are the two most popular methods used in face recognition that employ linear technique of subspace representation.

- **Eigenface:** is a method of extracting facial features based on holistic approach. It is uses the Principal Component Analysis (PCA) to reduce the dimension of the data. PCA calculates the Eigenvectors of the covariance matrix, and projects the original data onto a lower dimensional feature space, which are defined by Eigenvectors with large Eigenvalues [12].
- **Fisherface**: – instead of PCA, Fisherface uses Linear Discriminant Analysis (LDA) for dimensionality reduction of the image data. The distinction between the PCA and LDA techniques is that PCA is an unsupervised technique, while LDA is a supervised learning technique.

2.3 Deep Learning and Facial Recognition

Advancement in Machine Learning research yielded a class of neural networks with several layers of neurons. Using these models to make computer learn from data in order to make predictions is popularly called *deep learning*. One deep neural network that has been widely applied in image processing particularly face recognition task is the Convolutional Neural Network (CNN).This emerging technique of deep learning has reshaped the research landscape of face recognition research. [25] provides a survey of recent developments in this regard. Facial recognition with CNN is now the norm because it outperforms traditional machine learning algorithms such as SVM in terms of accuracy [26]. Therefore, deep learning found its way into the facial recognition arena due to 1) its potential to produce accurate results, 2) its ability to handle the task without breaking into sub tasks of face detection, feature extraction, identification etc. However, using deep learning is resource-intensive and it requires large dataset for training. For this reason we opt to use two traditional ML methods (SVM and KNN) together with Eigenface and Fisherface to access the performance of libraries used for facial recognition.

3. Related Works

Studies that performed comparison of facial recognition techniques exist in the literature. However, empirical studies that evaluate libraries used for implementation of facial recognition techniques are very few. [27] proposed a face recognition method based on Fisherface's LDA. The proposed method which is insensitive to variation in lighting and facial expression is compared with Eigenface using Harvard and Yale face databases. They reported that Fisherface's method has low error rate. [28] proposed a facial recognition method which they called Laplacianface. They compare it with Eigenface and Fisherface on three datasets. [29] applied some data transformations (discrete wavelet, cosine transform) in order to see their effect on three popular facial recognition methods namely, Eigenface, Fisherface and ICA. They reported that ICA and Eigenface produced better results than Fisherface. The work of [29] focused on examining the effect of performing transformations on the algorithms not accessing the libraries used for implementation. [30] compared four different face recognition techniques and present general discussion on the training requirements of the techniques. [31] investigate the effectiveness of eight algorithms and evaluated them on six facial databases. [33] compared LBPH, Eigenface and Fisherface algorithms using Haar cascade for facial identification. They implemented the algorithms using OpenCV library. Their work seeks to determine which of the 3 algorithms perform better. OpenCV was used by [34] to evaluate Eigenface, Fisherface and LBPH algorithms for recognising faces in real time-images captured via camera. They reported that Fisher face and LBPH gave best performance. In all the

aforementioned works, comparison of face recognition algorithms was done. Their focus was to only determine the effectiveness of the methods/techniques. None of the studies attempted to evaluate or compare images processing libraries. In fact, some of them remain silent on the library they used. In contrast, our work take step to determine both the effectiveness of the algorithms and the libraries used for implementing them.

An evaluation to determine the accuracy of face detection tools in inferring attributes such as gender, race, and age of person in an image has been carried out by [32]. The face detection tools they evaluated are Face++, IBM Bluemix Visual Recognition, AWS Recognition, and Microsoft Azure Face API. Their work is clearly about detecting of features in the picture that helps to make inference about . Like our work, [35] try to answer the question "which of the face detection tools provides best performance?". The author compares OpenCV, YOLOFace, and face_recognition tools with respect to accuracy and time in recognizing faces. The author's aim was simply to compare how best the tools recognize face in a given image without regard to the underlying techniques used by the tools. However, In addition to differences in libraries, our work investigates the performance of a library for a given technique(s). For example, using Eigenface and KNN, or using Fisherface with SVM, which library performs best?

4. Methodology

To achieve the desired goal, the task in this paper entails building a face recognition system that uses Fisherface and Eigenface with the combination of traditional ML classifiers. This section presents the architecture of the system, metrics, the libraries and environment used to perform the comparisons along with the techniques to be compared. The statistical methods used for the comparisons are presented and some validity threats and alternative methods are also discussed. Figure 1 below shows the system architecture.

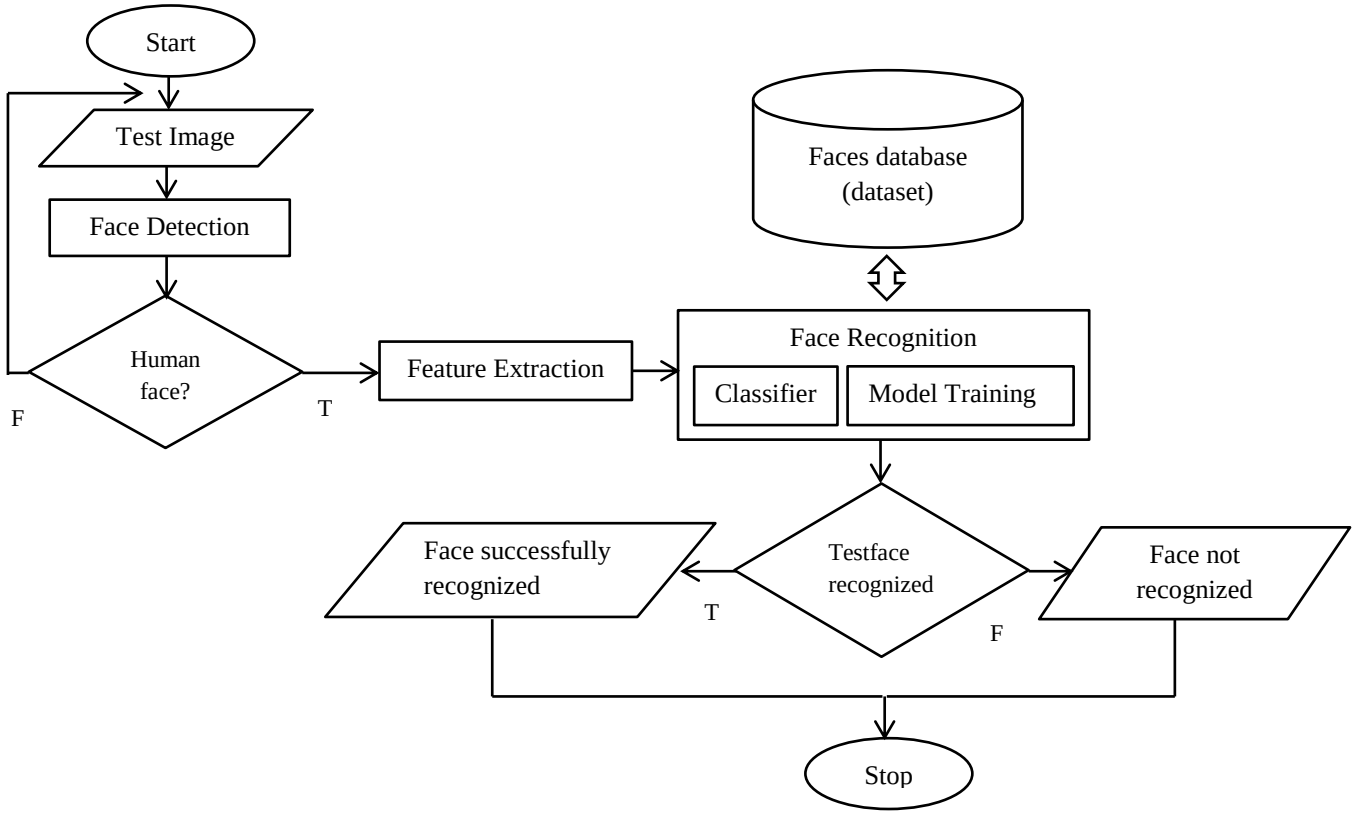

Fig.1. Architecture Face Recognition system

4.1 The Techniques

Face recognition task can be performed different ways and several different techniques have been used together for this purpose. The algorithms selected in this work are Eigenface and Fisherface. Based on the principle they operate, a classifier must be used in order to complete the facial recognition process. Hence two popular classifiers Support Vector Machine (SVM) and K Nearest Neighbour (KNN) are therefore used in conjunction with Eigenface and Fisherface. We take into consideration (1) popularity (2) the way the techniques are applied in various researches and (3) library implementations (because not all libraries provide their implementations) as the reasons for this selection.

Table 1. Face Recognition Techniques

Technique	Library	Technique
CV_Eigen	OpenCV	Eigenface with SVM
CV_Fisher	OpenCV	Fisherface with SVM
S_Eigen_K	scikit-learn	Eigenface with KNN
S_Eigen_S	scikit-learn	Eigenface with SVM
S_Fisher_K	scikit-learn	Fisherface with KNN
S_Fisher_S	scikit-learn	Fisherface with SVM

4.2 Image Processing Libraries

Variety of libraries for Image processing exists. Notable, these include OpenCV [36], Scikit [37], Scipy, Python Imaging Library (PIL), Mahotas, SimpleITK, Pgmagick [38]. The aforementioned libraries are based on python programming language. Over the years, python has become a language of choice for machine learning researchers and data scientist simply because of its clean syntax, loose type, and rich functionalities for manipulating large arrays. Of course libraries for other language may exist but in this work we use OpenCV and scikit learn libraries which are python based. We selected the two libraries for two reasons (1) they provides implementations of the algorithms we are using in this work. (2) they are widely used for face recognition and other image processing related tasks.

4.3 Performance Metrics

The following performance metrics were used to evaluate the facial recognition techniques' implementation by the two libraries: Accuracy, Precision, Recall, F-score, Fallaout and training time. The formula for computing accuracy, A is given by equation (1).

$$A = \frac{TP+TN}{N} \tag{1}$$

Where TP is the number of true positive, TN is the number of true negative and N is the total number of the samples. The limited version of the LFW dataset used in this work contains 10 classes, individuals, with 4 images each resulting in a population of 40. If only one out of the four test images for Person1 gets classified correctly this would give a TP value of 1 and a TN value of 36 which gives an accuracy of $\frac{37}{40}$, 92.5%, even though Person1 only got correctly classified one out of four times. The precision P, is computed by equation (2)

$$P = \frac{TP}{TP+FP} \tag{2}$$

where FP is the number of false positive while TP is as defined above. Recall R, calculated by taking all TPs and divide it by the number of test images for the respective class, P, which mathematically equation 3. For example, if only one out of the four test images of Tony Blair get classified correctly and the remaining three got classified incorrectly, this would give a TP value of 1, which gives a recall of $\frac{1}{4}$, 25%. This gives a better representation of the performance of the technique than accuracy, since it only cares about the TPs, rather than the TNs. A high recall does imply that the technique performs well and has a high rate of correct predictions.

$$R = \frac{TP}{P} \tag{3}$$

F-score is arguably the best metric to determine a method/technique's ability of doing correct and incorrect predictions. F-score is computed by equation 4.

$$F = \frac{P*R}{P+R} * 2 \tag{4}$$

Where P is precision, and R is the Recall. The Fallout, F_o is given by equation (5)

$$F_o = \frac{FP}{FP+TN} \tag{5}$$

Fallout, F_o is the FP rate, meaning the probability of a false alarm, or an incorrect prediction. The optimal of this is 0%. The training and test times are measure by the difference between the final and initial times.

5. Experiment

Both Eigenface and Fisherface are techniques that aid the process of facial recognition and are combined with a classifier in order to complete the facial recognition process. Two popular classifiers namely SVM and KNN are used in conjunction with Eigenface and Fisherface. The overall aim of this work is to establish the basis to determine the best performing algorithm and library. The experiment is expected to reveal (1) algorithm/technique that produces best result in terms of accuracy (2) the classifier with outstanding performance, and finally (3) the library that produces the best results.

5.1 Dataset

The dataset used in this work is Labelled Faces in the Wild (LFW) [7]. It is a benchmark dataset used in facial recognition researches. It contains 13,233 target faces images labelled with names of individuals. Some images contain more than one face. The images are available in 250 by 250 pixel JPEG format. The database contains images of 5749 different individuals. Of these, 1680 people have two or more images in the database. The remaining 4069 people have just a single image in the database.

Every technique used in the experiment needs labels for the training phase, since they are supervised techniques. Out of the LFW dataset the 10 individuals with the most images are selected to build a limited version of the dataset. Each of these 10 individuals have 40 images intended for training and four images for testing. This gives a total of 440 images that are split into 400 training images and 40 testing images in total. This split of the images was done randomly. One more random split has been done to group these images into ten parts for 10-fold cross-validation, which will be used to mitigate the possibility of the dataset getting a misrepresenting split.

5.2 Settings

Both OpenCV and Scikit-learn libraries have implementation of a number of classifiers including KNN and SVM. They equally have implementations of Eigenface and Fisherface algorithms. The number of components, a parameter used for dimensionality reduction is set to 150. A higher number should in theory result in both better predictions and longer training times. The default value is none for scikit-learn, while the default value is 0 for OpenCV.

For scikit-learn, we set whiten equals to True. whitten: Removes noise and improves the predictive accuracy. The default value is False. For KNN, the choice of k is important as it has a significant effect in the result. The scikit-learn documentation mentioned that 25 should be sufficient. However, we randomly tested numbers within between 1 and 100 to find the best k value for each of the models which use KNN. The k-value which gave the best result was 13 for the Eigenface and 17 for the Fisherface implementation. The gamma parameter defines how far the influence of a single training example reaches, with low values meaning ‘far’ and high values meaning ‘close’. The default value is: “auto_deprecated”. We used the value ‘scale’ for gamma. For SVM, we set kernel type as linear. Since the dataset used in the experiments has the same amount of testing and training images which would make the class weights balanced. We set class_weight = ‘balanced’.

For OpenCV implementation using SVM, we used default values of gamma, kernel, and class_weight and ‘scale’, ‘linear’ and ‘balanced’ respectively.

5.3 Results

The results represent the performance per individual where 10 different persons were predicted, and the average for all the metrics is taken, and that is for each algorithm. The average scores for each of the techniques are shown in table 2 below.

Table 2. Average scores of the techniques

Name	Accuracy	Precision	Recall	F-Score	Fallout
CV_Eigen	0.8310	0.1534	0.1550	0.1453	0.0939
CV_Fisher	0.8387	0.1927	0.1935	0.1834	0.0896
S_Eigen_K	0.8733	0.3815	0.3667	0.3111	0.0704
S_Eigen_S	0.9314	0.6873	0.6569	0.6463	0.0381
S_Fisher_K	**0.9370**	**0.7044**	**0.6850**	**0.6723**	**0.0350**
S_Fisher_S	0.9315	0.6688	0.6575	0.6409	0.0381

To display the results in graph form, the average scores were converted into percentage and plotted in bar chart. The accuracy of the techniques is shown in Fig 2, where FisherFace and KNN implemented with Scikit-learn library (S_Fisher_K) achieve the highest accuracy score of 93.7%. Similarly, scikit-learn implementation of Fisherface and KNN has the best recall of 68.50%, as shown in Fig 3. In terms of precision, scikit-learn implementation of Fisherface

with KNN once again recorded the highest precision score of 70.44% as shown in Fig 4. F-score is arguably the best metric to determine a method/technique's ability of doing correct and incorrect predictions. As shown in Fig 5, scikit-learn implementation of Fisherface and KNN (S_Fisher_K) is the best with 67.23% followed by scikit-learn implementation of Eigenface and SVM 64.63%. The graph in Fig 6 below shows the average fallout of the different techniques. Fallout is the probability of a false alarm, or an incorrect prediction and the optimal of this is 0%. The best fallout is achieved by S_Fisher_K with the optimal of 3.50% and the worst by the OpenCV implementation of Eigenface with 9.39%.

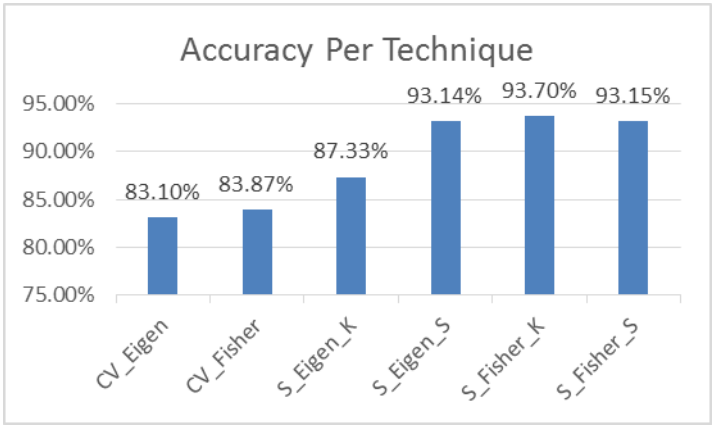

Fig. 2. Accuracy of Techniques measured in percentage

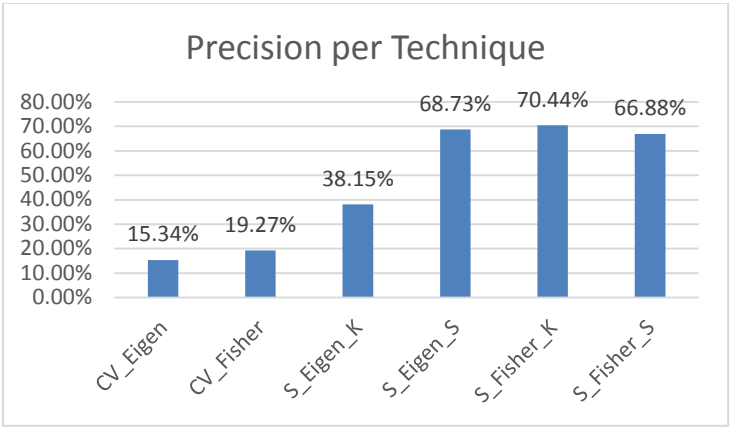

Fig. 3. Precision per technique measured in percentage

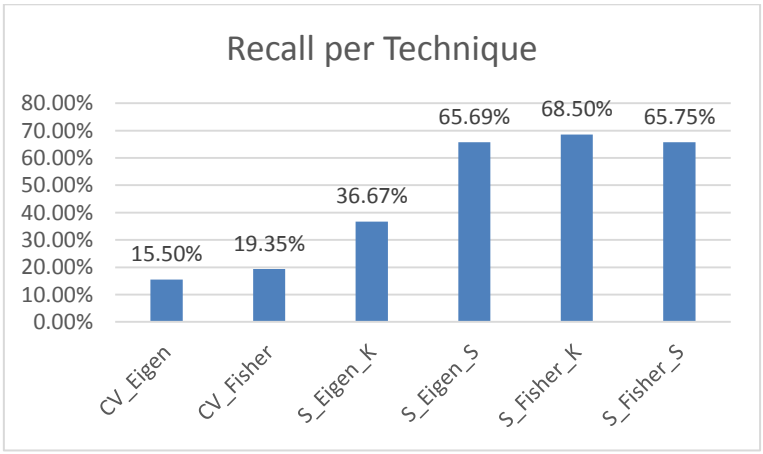

Fig. 4. Recall per technique measured in percentage

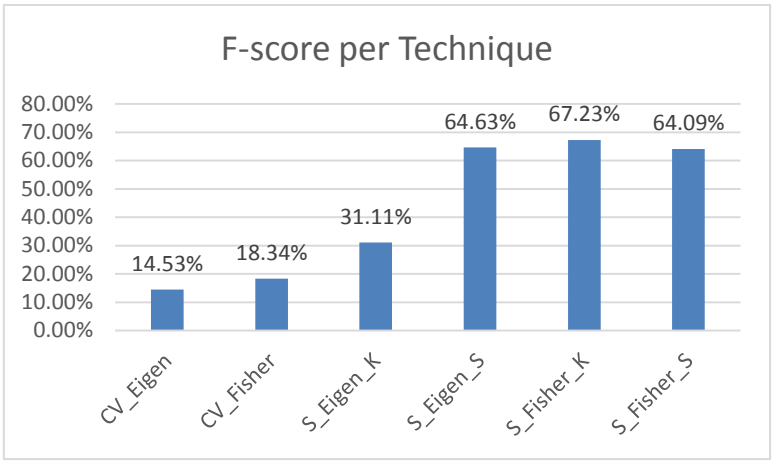

Fig. 5. F-score per Technique measured in percentage

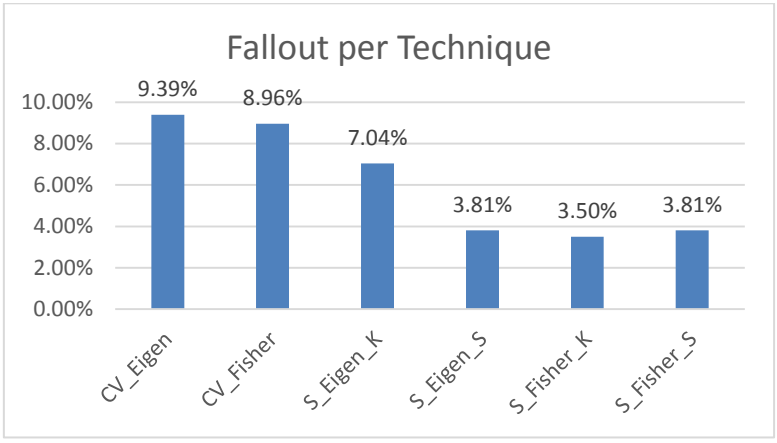

Fig. 6. Fallout per Technique measured in percentage

5.4 Training & Testing Time

Training time is the time it takes to train the model for a specific technique on the 400 training images in the dataset. The results are shown in Table 4. Green indicates the technique which had the shortest training time and red indicates the technique with the longest training time.

Table 3. Training and Prediction Time

Technique	Training Time(sec)	Prediction Time(sec)	
CV_Eigen	3.68	0.0982	OpenCV, EigenFace & SVM
CV_Fisher	4.19	0.0075	OpenCV, FisherFace & SVM
S_Eigen_K	**0.38**	0.0070	Scikit, EigenFace & KNN
S_Eigen_S	0.43	0.0034	Scikit, EigenFace & SVM
S_Fisher_K	1.02	0.0029	Scikit, FisherFace & KNN
S_Fisher_S	0.93	**0.0007**	Scikit, FisherF, SVM

5.5 Analysis & Discussion of Results

In this paper, we used metrics such as recall, precision and F-score in order to have wide and better understanding of the performance of the techniques/algorithms we compared in this work. From the results, the Scikit-learn implementation of Fisherface using KNN (S_Fisher_K) has the overall best score in every single performance metric. Comparing the F-score of OpenCV implementation of Fisherface with the SVM (CV_Fisher) and the scikit implementation of the same technique, the Scikit implementation has an F-score of 64.09% which is ahead of OpenCVs 18.34%. This means the scikit implementation 3.5 times better than the OpenCV. Another aspect of the results is the difference in performance comparing Fisherface with the classifiers KNN and SVM using the scikit implementation. The results show that Fisherface with KNN performs better than Fisherface with SVM regarding recall, precision and

F-score metrics. One reason could be that Fisherface with SVM uses the default settings, while Fisherface with KNN was tested using different k-values to find the optimal k-value, but this should be explored further. Examining the results further, the algorithms/techniques that performed well are EigenFace with SVM, FisherFace with KNN, and FisherFace and SVM. Although FisherFace with KNN outperform the rest, but the difference between their F-scores is not that much. This means they recognized faces with significant level of accuracy. Furthermore, the results were achieved using scikit learn library. This confirms that scikit learn outperform OpenCV in the experiment carried out. It can also be concluded that irrespective of classifier, the results shows FisherFace outperform the Eigenface algorithm. With regards to time, OpenCV implementations takes considerably longer time to train than the scikit-learn implementations of the same techniques. The scikit-learn implementation of Fisherface with SVM achieves the fastest prediction time with a time that is four times faster than the same implementation with KNN. This contradicts what most papers state about the time difference when comparing KNN with SVM that KNN predicts faster. SVM may be faster to predict on small datasets compared to KNN. The most surprising aspect about the results is the difference between the OpenCV and scikit implementations of the same technique. The results show that the scikit implementations perform significantly better in all metrics except for prediction time compared to the OpenCV counterparts even though they are different implementations of the same technique.

6. Conclusion

In this paper, Eigenface and Fisherface algorithms for face recognition are compared based on OpenCV and scikit learn image processing libraries. The Eigenface and Fisherface algorithms were combined with two traditional machine learning classifiers namely; KNN and SVM. The two techniques were implemented using scikit-learn and OpenCV image processing libraries. This is to enable us to achieve dual aims of (1) knowing the technique that perform best and (2) the image processing library that provides better implementation. We use more metrics in addition to accuracy metric in order to have wide and better understanding of the performance of the techniques/algorithms we compared. The results show that FisherFace with KNN outperform FisherFace with SVM, EigenFace with KNN, and EigenFace with SVM. Accordingly, Fisherface algorithm outperforms Eigenface in both the two library implementations. The higher/best scores were achieved using the scikit learn library. This shows scikit-learn library outperform OpenCV. The results also suggest that irrespective of classifier, FisherFace outperform the Eigenface algorithm. Now that researcher and other developers of system embedded with face recognition algorithms relied heavily on libraries, like the ones we compared in this work, it is important to have studies that assess the strength and weaknesses of these libraries. This will help in drawing the attention of creators/developers of these libraries to refine the existing ones and produce more robust and scalable libraries in the future for the benefits of users and researchers in the field of computer vision.

Acknowledgement

We are grateful to anonymous reviewer(s) for their thoughts and comments. Indeed their comments were very helpful.

References

[1] C. Chen, R. Surette, and M. Shah, “Automated Monitoring for Security Camera Networks: Promise from Computer Vision Labs,” *Secur. J.*, vol. 34, pp. 389–409, 2020.

[2] J. Zhao, R. Masood, and S. Seneviratne, “Review of Computer Vision Methods in Network Security,” *IEEE Commun. Surv. Tutorials*, vol. 23, no. 3, pp. 1838–1878, 2021, doi: https://doi.org/10.1109/COMST.2021.3086475.

[3] Y. Wang, P. Zheng, X. Xu, H. Yang, and J. Zou, “Production Planning for cloud based additive Manufacturing – A computer Vision based approach,” *J. Robot. Comput. Manuf.*, vol. 58, pp. 145–157, 2019, doi: https://doi.org/10.1016/j.rcim.2019.03.003.

[4] S. Paneru and I. Jeelani, “Computer Vision Applications in Construction: Current state, Opportunities & Challenges,” *J. Autom. Constr.*, vol. 132, 2021, doi: https://doi.org/10.1016/j.autcon.2021.103940.

[5] B. H. W. Guo, Y. Zou, Y. Fang, Y. M. Goh, and P. X. . Zou, “Computer Vision Technologies for Safety Science and Management in Construction: A critical Review and future Research Direction,” *J. Saf. Sci.*, vol. 135, 2020, doi: https://doi.org/10.1016/j.ssci.2020.105130.

[6] D. Forsyth and J. Ponce, *Computer Vision: A modern Approach*, 2nd ed. Prentice Hall Publishers, 2011.

[7] G. B. Huang, M. Ramesh, T. Berg, and E. Learned-miller, “Labeled Faces in the Wild : A Database for Studying Face Recognition in Unconstrained Environments,” in *workshop on faces in Real-Life images: detection. alignment, and recognition*, 2008, pp. 1–11.

[8] H. Ge, Z. Zhu, Y. Dal, B. Wang, and X. Wu, “Facial Expression Recognition,” *J. Comput. Methods Programs Biomed.*, vol. 215, 2022.

[9] G. Sunitha, K. Geetha, S. Neelakandan, A. K. S. Pundir, S. Hemalatha, and V. Kumar, “Intelligent Deep Learning Based Ethnicity Recognition and Classification using Facial Images,” *J. Image Vis. Comput.*, 2022, doi: 10.1016/j.imavis.2022.104404.

[10] S. J. Goyal, A. K. Upadhyay, and R. S. Jadon, “A Brief Review of Deep Learning Based Approaches for Facial Expression and Gesture Recognition Based on Visual Information,” *Mater. Today*, vol. 2, pp. 462–469, 2020.

[11] W. Mellouk and W. Handouzi, "Facial Emotion Recognition using Deep Learning : Review and Insights," in *Procedia Computer Science. 2nd International Workshop on the Future of Internet of Everything*, 2020, vol. 175, pp. 689–694, doi: 10.1016/j.procs.2020.07.101.
[12] Y. Kortli, M. Jridi, A. Al Falou, and M. Atri, "Face Recognition Systems : A Survey," *Sonsors (Basel)*, vol. 20, no. 2, p. 342, 2020, doi: 10.3390/s20020342.
[13] W. Zhao, R. Chellappa, P. J. Phillips, and A. Ronsenfield, "Face Recognition: A literature survey. (CSUR)," *ACM Comput. Surv.*, vol. 35, no. 4, pp. 399–458, 2003.
[14] M. Lal, K. Kumar, R. H. Arain, and A. Maitlo, "Study of Face Recognition Techniques : A Survey," *Int. J. Adv. Comput. Sci. Appl.*, vol. 9, no. 6, pp. 42–49, 2018.
[15] M. Turk and A. Pentland, "Eigenfaces for Recognition," *J. Cogn. Neurosci*, vol. 3, pp. 71–86, 1991.
[16] H. J. Seo and P. Milanfar, "Face Verification using the Lark Representation," *IEEE Trans. Inf. Forensics Secur.*, vol. 6, pp. 1275–1286, 2011.
[17] M. Annalakshmi, S. M. M. Roomi, and A. . Naveedh, "A hybrid technique for gender classification with SLBP and HOG features," *Clust. Comput.*, vol. 22, pp. 11–20, 2019.
[18] A. Vinay, D. Hebbar, V. S. Shekhar, K. B. Murthy, and S. Natarajan, "Two Novel Detector-descriptor Based Approaches for Face Recognition Using Sift and Surf," *Procedia Comput. Sci*, vol. 70, no. 185–197, 2017.
[19] T. Napoleon and A. Alfalou, "Pose Invariant Face Recognition: 3D Model from Single Photo," *Opt. Lasers Eng*, no. 89, pp. 150–161, 2017.
[20] A. Haji Rassouliha, T. P. . Gamage, M. D. Parker, M. P. Nash, A. J. Taberner, and P. . Nielsen, "FPGA implementation of 2D Cross-Correlation for Real-time 3D Tracking of Deformable Surfaces," in *IEEE 28th International Conference on Image and Vision Computing*, 2013, pp. 352–357.
[21] I. Kambi Beli and C. Guo, "Enhancing Face Identification Using Local Binary Patterns and k-nearest Neighbors," *J. Imaging*, vol. 3, no. 37, 2017.
[22] F. Schroff, D. Kalenichenko, and J. F. Philbin, "A Unified Embedding for Face Recognition and Clustering," in *IEEE conference on computer vision and pattern recognition*, 2015, pp. 815–823.
[23] M. Slavkovic and D. Jevtic, "Face Recognition using Eigenface Approach," *Serbia J. Electr. Eng.*, vol. 9, no. 1, pp. 121–130, 2012.
[24] K. Simonyan, O. . Parkhi, A. Vedaldi, and A. Zisserman, "Fisher Vector Faces in the Wild," 2013.
[25] M. Wang and W. Deng, "Deep Face Recognition : A Survey," *J. Neurocomputing*, vol. 429, pp. 215–244, 2020, doi: 10.1016/j.neucom.2020.10.081.
[26] Sightcorp, "Face Recognition Using Deep Learning," 2022. http://sightcorp.com/knowledge-base/face-recognition-deep-learning/ (accessed Feb. 09, 2022).
[27] P. . Belhumer, J. . Hespanha, and D. . Kriegman, "Eigenfaces vs Fisherfaces: Recognition using Class Specific Linear Projection," *IEEE Traansactions Pattern Anal. Mach. Intell.*, vol. 19, no. 17, pp. 711–720, 1997, doi: 10.1109/34.598228.
[28] X. He, S. Yan, Y. H. Hu, and H.-J. Zhang, "Learning a Locality Preserving Subspace for Visual Recognition," in *of 9th IEEE International Conference on Computer Vision*, 2003, pp. 385–392.
[29] M. Sharkas and M. Abou Elenien, "Eigenfaces vs. Fisherfaces vs. ICA for face Recognition: A Comparative Study.," in *9th International conference on signal processing*, 2008, pp. 914–919.
[30] F. . Bhat and M. . Wani, "Performance Comparison of Major Classical Face Recognition Techniques," in *13th International Conference on Machine Learning and Applications*, 2014, pp. 521–528, doi: 10.1109/ICMLA.2014.91.
[31] D. Sadhya, A. Gautam, and S. K. Singh, "Performance Comparison of Some Face Recognition Algorithms on Multi-Covariate Face Databases," in *4th International Conference on Image Information Processing (ICIPP)*, 2017, pp. 1–5, doi: 10.1109/ICIIP.2017.8313741.
[32] S. Jung, J. An, H. Kwak, J. Salminen, and B. J. Jansen, "Assessing the Accuracy of Four Popular Face Recognition Tools for Inferring Gender, Age, and Race," in *12th International AAAI Conference on Web and Social Media (ICWSM 2018)*, 2018, pp. 624–627.
[33] A. . Jagtab, V. Kangale, K. Unune, and P. Gosavi, "A Study of LBPH, Eigenface, Fisherface and Haar-like Features for Face Recognition using OpenCV," in *IEEE International Conference on Intelligent Sustainable Systems (ICISS)*, 2019, pp. 219–224, doi: 10.1109/ISS1.2019.8907965.
[34] Z. Arya and V. Tiwari, "Automatic Face Recognition and Detection Using OpenCV , Haar Cascade and Recognizer for Frontal Face," *Int. J. Eng. Res. Appl.*, vol. 10, no. 6, pp. 13–19, 2020, doi: 10.9790/9622-1006051319.
[35] A. T. Kabakus, "An Experimental Performance Comparison of Widely Used Face Detection Tools," *Adv. Distrib. Comput. Artif. Intell. J.*, vol. 8, no. 3, pp. 5–12, 2019.
[36] G. Bradski, "OpenCV Library," *J. Softw. Tools Prof. Program.*, vol. 25, no. 11, pp. 120–123, 2000.
[37] V. Der Walt, J.L . Schonberger, N. Juan, B. Francois, J.D Warner, Y. Neil, G. Emmanualle, Y. Tony "Scikit-image: Image Processing in Python," *PeerJ*, pp. 1–18, 2014, doi: 10.7717/peerj.453.
[38] G. S. Panwar, "Top 8 Image-Processing Python Libraries Used in Machine Learning," 2020. .

Authors' Profiles

Ismail Aliyu obtained his Msc degree in Software Engineering from University of Sheffield, UK in 2013. He obtained his first degree in Computer Science from Abubakar Tafawa Balewa University Bauchi Nigeria, 2010. He is currently pursuing his PhD in Ahmadu Bello University Zaria. His research interest is in Artificial Intelligence specifically the application of Machine Learning and Deep Learning algorithms to solve problems in NLP & Computer Vision areas.

Muhammad Ali Bomoi obtained his B.Tech degree in Computer Science from Abubakar Tafawa Balewa University Bauchi in 2021. He also possesses CCNA certification. As a young graduate, his research interests are in networks and computer vision with focus on image processing.

Maryam Maishanu obtained her first and masters degrees in in Computer Science from Abubakar Tafawa Balewa University Bauchi Nigeria in 2000 and 2013 respectively. She is a Cisco Certified Academy Instructor (CCAI). Her research interests include Artificial Intelligence and cyber security.

How to cite this paper: Ismail Aliyu, Muhammad Ali Bomoi, Maryam Maishanu, " A Comparative Study of Eigenface and Fisherface Algorithms Based on OpenCV and Sci-kit Libraries Implementations", International Journal of Information Engineering and Electronic Business(IJIEEB), Vol.14, No.3, pp. 30-40, 2022. DOI: 10.5815/ijieeb.2022.03.04

