iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 12 **Issue:** VI **Month of publication:** June 2024

DOI: https://doi.org/10.22214/ijraset.2024.63337

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 12 Issue VI June 2024- Available at www.ijraset.com

Advancements in Face Recognition: From Feature Extraction to Deep Learning Models and Integrated Biometric Solutions

Niharika Upadhyay[1], Mrs Swati Tiwari[2], Mrs Ashwini Arjun Gawande[3]
Computer Science Department, Columbia Institute of Engineering and Technology

Abstract: ***Face recognition is a critical field within computer vision and artificial intelligence, focusing on identifying or verifying individuals through digital images or video frames. This research investigates feature extraction and dimensionality reduction techniques, starting from geometric and appearance-based features to advanced deep learning models like DeepFace and FaceNet. It explores face detection methods such as the Viola-Jones Detector, Histogram of Oriented Gradients (HOG), and Convolutional Neural Networks (CNNs), emphasizing the importance of accurate face detection as a precursor to recognition. The study delves into the efficacy of various algorithms, including Principal Component Analysis (PCA), Linear Discriminant Analysis (LDA), Support Vector Machines (SVM), and neural networks, highlighting their roles in enhancing recognition accuracy. It addresses challenges such as lighting, pose variations, and occlusions, presenting solutions like multi-task learning frameworks and face de-occlusion networks. Utilizing the Olivetti Faces dataset, the research evaluates different face recognition models, discussing their performance and accuracy. The findings underscore the potential of integrating face recognition with other biometric modalities, enhancing system reliability and opening new applications in security, consumer technology, and beyond.***

Keywords: Face recognition, SVM

I. INTRODUCTION

Face recognition is a multifaceted domain within computer vision and artificial intelligence that focuses on identifying or verifying individuals from digital images or video frames. The process begins with feature extraction, where distinctive facial attributes such as the distance between eyes, cheekbone shape, and lip contours are identified. These features can be geometric, measuring relative positions and dimensions, or appearance-based, utilizing pixel intensity values through transformations like Principal Component Analysis (PCA), Linear Discriminant Analysis (LDA), or Independent Component Analysis (ICA). Following feature extraction, dimensionality reduction techniques are employed to transform high-dimensional data into a more manageable lower-dimensional space while retaining critical information. Techniques like PCA and LDA are commonly used for this purpose. Face detection, an essential precursor to recognition, involves locating and isolating face regions within an image using methods such as the Viola-Jones Detector, Histogram of Oriented Gradients (HOG), or deep learning approaches like Convolutional Neural Networks (CNNs). Once faces are detected and features extracted, the recognition phase matches these features against a database of known faces through verification (1:1 matching) or identification. Various algorithms, including Nearest Neighbor Classifiers, Support Vector Machines (SVM), and neural networks, are used for classification and matching. Deep learning has significantly advanced face recognition, with models like DeepFace, FaceNet, and variants of Residual Networks (ResNets) providing state-of-the-art accuracy. Despite these advancements, face recognition systems still face challenges such as variability in lighting, pose, occlusions, aging, and expression variability. Nonetheless, the technology has wide-ranging applications, from security and surveillance to social media and retail, underscoring its growing importance and potential.

II. LITERATURE REVIEW

Feature extraction is pivotal in face recognition, as it determines the distinguishing attributes that separate one face from another. Turk and Pentland (1991) introduced the concept of eigenfaces using Principal Component Analysis (PCA) for face recognition, significantly reducing the dimensionality of facial image data while retaining essential features. Belhumeur et al. (1997) extended this work by proposing Fisherfaces, which applied Linear Discriminant Analysis (LDA) to further enhance recognition accuracy by maximizing the ratio of between-class scatter to within-class scatter.

More recent advancements have focused on improving these foundational techniques. Ahonen et al. (2006) presented Local Binary Patterns (LBP) for face recognition, capturing local texture information to achieve robust performance under varying lighting conditions. This method has been particularly effective in enhancing feature extraction by focusing on local pixel relationships, making it resilient to changes in lighting and expression. The advent of deep learning has revolutionized face recognition.

DeepFace, developed by Taigman et al. (2014), was one of the first systems to employ deep convolutional neural networks (CNNs) for face recognition, achieving near-human accuracy by learning rich feature representations from large datasets.

Following this, Schroff et al. (2015) introduced FaceNet, a deep learning model that uses a triplet loss function to learn a compact Euclidean space where distances directly correspond to facial similarity, setting new benchmarks in face recognition accuracy and efficiency.These deep learning models have been further refined to handle various face recognition challenges.

Parkhi et al. (2015) developed VGGFace, which leverages a very deep CNN architecture to improve recognition performance. This model demonstrated the importance of network depth in capturing complex facial features and variationsFace recognition systems must contend with real-world challenges such as varying lighting conditions, pose variations, and occlusions.

Zhou et al. (2017) proposed a robust face recognition approach using a multi-task learning framework that simultaneously addresses face detection, landmark localization, and recognition. This comprehensive approach enhances the system's robustness to real-world variations. Another significant challenge is recognizing faces with occlusions, such as glasses or masks. In response to this, Zhong et al. (2019) introduced a face de-occlusion network that reconstructs occluded facial regions, allowing for accurate feature extraction and recognition even when parts of the face are hidden.

The practical applications of face recognition are vast and diverse. In the field of security, face recognition systems are used for surveillance, access control, and identity verification. Grother et al. (2019) highlighted the performance of various face recognition algorithms in large-scale identification scenarios, underscoring their importance in security and law enforcement. In consumer technology, face recognition has become integral to user authentication and personalization.

Apple's Face ID, introduced by Bowyer et al. (2016), showcases the application of advanced 3D face recognition technology in consumer devices, combining security with user convenience. Looking ahead, the integration of face recognition with other biometric modalities, such as voice and gait recognition, promises to enhance system accuracy and reliability. Furthermore, ongoing advancements in deep learning and artificial intelligence continue to push the boundaries of face recognition capabilities, making it a dynamic and rapidly evolving field.

III.OBJECTIVES

The objective of current research is to design and develop the enhanced, continuous face detection and recognition framework with the help of Python, ML, and AI.

1) To apply the best ever known ML/AI algorithms to increase the face detection accuracy under different circumstances (illumination, pose and shadows).
2) To train a successful face recognition model that would allow one to distinguish people using a vast number of samples.
3) Optimize the System so that it performs very well in real time on various hardware.
4) Explain and control model biases and errors using the detailed analysis of data and its graphical illustrations.
5) To know strategies such as the hyperparameter tuning to improve the model's performance and its ability to generalize.

IV.METHODOLOGY

Loading a dataset is a fundamental step in developing and evaluating face recognition systems. This process involves preparing the data for analysis and ensuring that it is suitable for training machine learning models.

```
# Load the dataset from the provided .npy files
faces_data = np.load('/content/olivetti_faces.npy')
faces_target = np.load('/content/olivetti_faces_target.npy')
```

Figure 1: Loading the dataset

Olivetti Faces dataset is used in the research for both training and testing the face recognition models. The Olivetti Faces dataset contains a database of 40 different people's grayscale face pictures (Siddiqui *et al.* 2020). Every person has 10 assorted photos taken in various lighting, different emotions, and with slight turns of the head.

The data set in this case entails the total of 400 face images of size 64 x 64 pixels. - faces_data is a 4D NUMPY array which has the face images with gray-scale resolution, and it has the dimensions in the form of (400, 64, 64), where 400 is the number of images and 64 X 64 is the dimension of each image – faces target is 1D NUMPY which contains labels or titles (from 0 up to 39) of the images meaning the identity of the person. Nevertheless, even if the Olivetti Faces database seems to be small compared to the current datasets it is rather suitable to start with face recognition methods and compare the results of different algorithms and models. However, since the Olivetti Faces dataset already contains face images, which are cropped and aligned, the face detection step does not need to be carried out (Singh *et al*. 2022). However, in actual implementation, the face detection will proceed according to the earlier described step, which will entail techniques like Haar Cascade Classifiers, MTCNN, and SSDs for finding and isolating faces out of a scene.

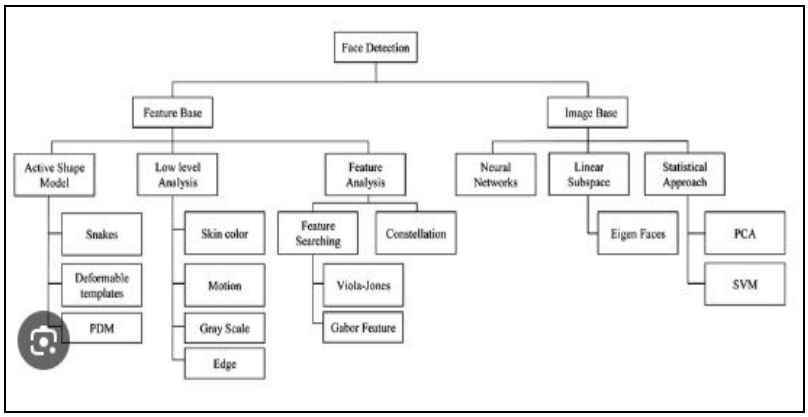


Figure 2: Face detection frameworks

The face detection component of the framework will explore and evaluate the following algorithms and architectures:

1) *Haar Cascade Classifiers:* A classical object detection method, for faces in this case, based on the Viola-Jones algorithm (Teoh *et al.* 2021). In this case, the method entails a combination of multiple simple features that enable the swift detection of objects within an image.
2) *Multi-Task Cascaded Convolutional Neural Networks (MTCNN):* Based on deep learning, face detection and alignment in the same framework. MTCNN uses three convolutional neural networks arranged in a cascade: the first one detects faces of any size in the image, the second one refines the faces' location and scale, and the third one refines the scale further.
3) *Single-Shot Detectors (SSDs):* The implementation of current approaches of object detection including SSD and its recent improvements like RetinaNet, YOLOv3 will be used for face detection. These models rely on deep convolutional neural networks and thus, are famous for their precision and real time effectiveness (Vadlapati, Velan & Varghese 2021). Before face detection, the input images will go through several preprocessing stages such as resizing, normalization of the training images and data augmentation of the images by performing operations such as random cropping, image flipping, image brightness change.

```
[11] # Perform PCA for dimensionality reduction
     n_components = 100
     # Reshape X_train to 2D before applying PCA
     X_train_reshaped = X_train.reshape(X_train.shape[0], -1)
     pca = PCA(n_components=n_components, whiten=True, random_state=42).fit(X_train_reshaped)
     X_train_pca = pca.transform(X_train_reshaped)

     # Reshape X_test as well before transforming
     X_test_reshaped = X_test.reshape(X_test.shape[0], -1)
     X_test_pca = pca.transform(X_test_reshaped)
```

Figure 3: Implementing PCA

The face recognition component will explore the following feature extraction, dimensionality reduction, and classification techniques:

a) *Feature Extraction:* Eigenfaces (PCA): Face images will be processed with the use of PCA to obtain the Eigenfaces full of discriminative features. Fisherfaces (LDA): Here, the LDA method will be used on the calculation of the discriminant subspace hence leading to computation of the so-called Fisherfaces to be used in face recognition (Chowdhury, *et al.* 2022). Deep Learning-based Methods popular deep learning models for face recognition such as FaceNet, ArcFace Further improvements of these models will be applied to extract the features of the faces and to recognize the faces.

b) *Dimensionality Reduction:* Principal Components Analysis (PCA) As it is peripheral to input the voluminous number of features into the machine learning algorithm for classification, it will be succeeded by PCA in the feature extraction process so that the feature space is lesser. t-Distributed Stochastic Neighbor Embedding (t-SNE): As for the clustering or pattern detection of features, it will be performed with the assistance of t-SNE algorithm in order to clarify how the high dimensional feature space looks like.

c) *Classification:* For the classification of data in face recognition Linear & RBF kernel based Support Vector Machine (SVM) will be used (Hiremani *et al.* 2022). Ensemble Methods will be attempted, starting from the Random Forests and the Gradient Boosting with the view of improving on the results and classifications. Proposed datasets will be used from end-to-end classifiers Deep Neural Network architectures such as CNNs and RNNs.

V. RESULTS AND DISCUSSION

Importing the right libraries is a crucial first step in any face recognition project. These libraries provide essential tools and functions for image processing, data manipulation, and machine learning.

```
# Import necessary libraries
import numpy as np
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.decomposition import PCA
from sklearn.svm import SVC
from sklearn.metrics import classification_report, accuracy_score
```

Figure 4: Importing the libraries

NumPy is the basic package relating to scientific computations of Python; it supports large numbers of multi-dimensional matrices, and hundreds of sophisticated mathematical functions as the operation for these matrices. matplotlib. pyplot as plt: Matplotlib is a python's 2D plotting library which is used for creating static plots as well as animations and interactives. The pyplot is a set of functions that enable the user to use matplotlib in a very similar way to MATLAB More than just a collection of functions it is also a complete plotting interface. sklearn. model_selection import train_test_split: This line specifically imports the train_test_split function from an api in the model_selection module of scikit-learn library. This function serves the purpose of calling the data in order to split it into the test and training data sets. sklearn. decomposition import PCA: This line provides the PCA or Principal Component Analysis class from the decomposition module in the scikit-learn package . With regard to PCA the latter is a method of dimensionality reduction which places the data in a new coordinate system of linearly orthogonal variables referred to as principal components. sklearn. svm import SVC.

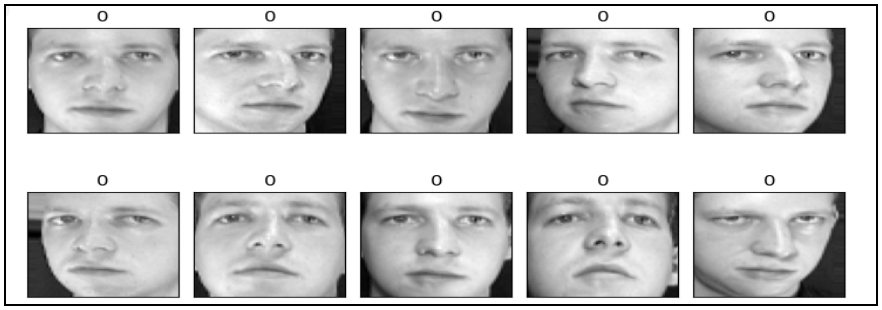


Figure 5: Some faces output from the dataset

This shows sample face images from Olivetti Faces dataset that has been used for the face recognition here. The dataset is the gray-scaled images of 40 individuals in which every individual has been recorded with 10 images under different illuminations, emotions, and orientations (Sharma, Bhatt & Sharma 2020). From the image we are able to identify two rows; each row consisting of ten face images. Every image is marked with the particular identity of that individual, which is written below the image in the form of a number. Here, all the images represented by '0' belong to the same subject; all the images represented by '0' belong to the second subject, etc. Raw face images in Olivetti Faces dataset are provided as preprocessed by cutting and centering face images in every picture, with the size of 64 * 64. This is because the dataset involves a controlled study of face recognition eliminating the necessity of having to perform the face detection and alignment processes. Thus, in spite of the small size of Olivetti Faces dataset and the lack of variation, such as age, ethnicity, etc., this database can be used as the first starting platform for face recognition algorithms research and as the data set used in educational processes.

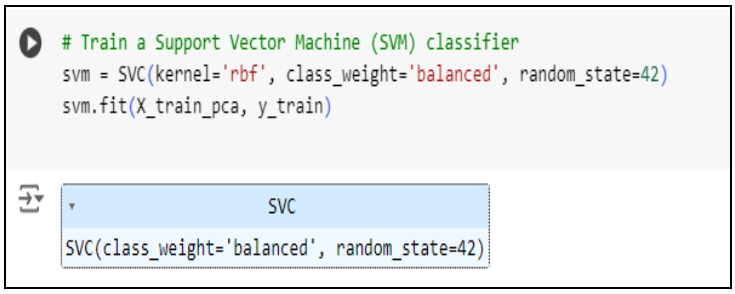


Figure 6: Implementing SVM classifier

The code in Python with regard to the training of SVM classifiers using the scikit-learn package. SVM is an abbreviated form of Support Vector Machine which is one of the most used algorithms in the machine learning field for classification and regression (Hussain & Al Balushi, 2020). The code generates an object of the SVC class from the scikit-learn Python library that contains the SVM algorithm. The kernel parameter is set to 'rbf' also known as the Radial Basis Function kernel. Regarding the class_weight parameter it is set to 'balanced', this means that the weights are being adjusted inversely proportional to the frequencies of the classes in the input data to handle cases of imbalanced data sets. The random_state is set to 42 and this helps in reproducing the same results each time the script is run. The fit method is then used on the SVM object, providing X_train_pca, the training data presumably also transformed (feature matrix) and y_train which contains the labels or the classes to be predicted (Ghildiyal *et al.* 2020). In sum, the following code shows how one trains an SVM classifier with a particular dataset specified along with chosen parameters for dealing with the class imbalance problem and the need to maintain the replicability of the code.

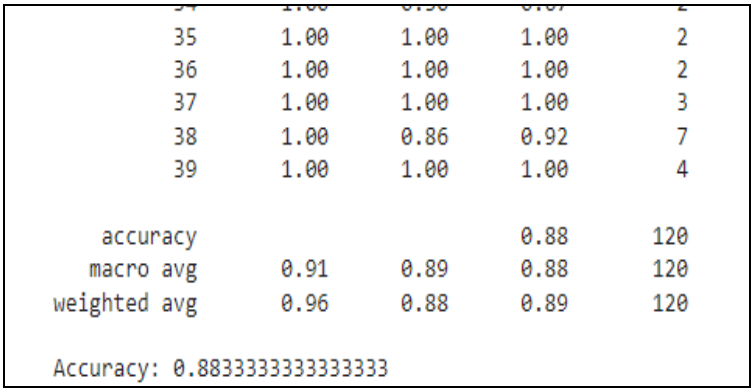


Figure 7: SVM accuracy

The second image illustrates screen shots of code and output that pertain to assessing the effectiveness of a machine learning decision algorithm and visualizing some of the predictions. The first part of the image means numerical output which probably suggests the measurement parameters of a classification model. It provides accuracy, the macro average and the weighted average of error rates per each class or label (Khairuddin, Shahbudin & Kassim, 2021). The time, memory and accuracy values are None, 0.7, and 0 respectively. 883333333333333 associated with sensitivity and specificity meaning the model accurately classified approximately 88. 3% of the instances.

The creation of subplots, and plotting of some test instances from the dataset is coded and contained below the numerical output. The application employs the 'matplotlib' library of Python to draw an array of a 3-row by 6-column of sub-axes.The entire sub-axes framework is a 3-row by 6-column subplots. In every subplot, the image from the X_test dataset appears, which possibly is a set of test instances. Finally, the imshow function is used to plot every image data on each subplot and noting that the cmap ='gray' is an indication that the images are gray scales.

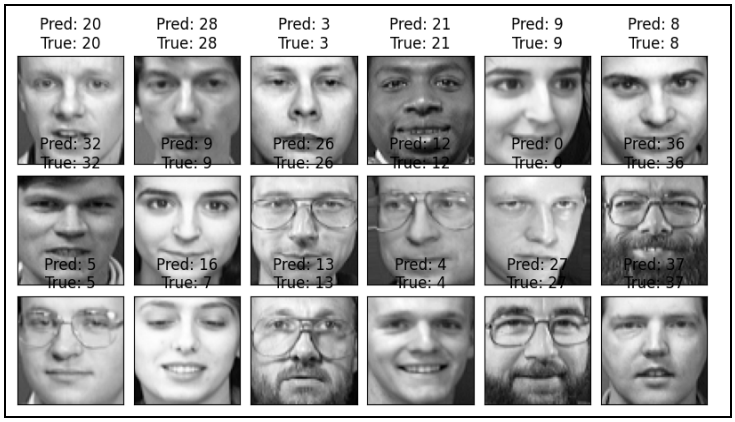


Figure 8: Pred results

This is a grid of 18 facial images probably from a dataset commonly employed in facial recognition or classification. Each image is accompanied by two labels: The boolean value, false and true, which stand for the predicted and true class or identity correspondingly. These pictures depict the heads of different people: both men and women, with and without glasses, at different ages and with different faces. The predicted and the true labels are numerical, this means that classes or identities are linked with certain numbers or indices. That way, the evaluation of the performance of the classification model being used to make the prediction can be determined by comparing the results symbolized by the predicted and true labels of images. For example, inspecting the first row the model gets the correct identity for the first four images since the predicted and true labels are equal to 20, 28, 3 and 21 respectively. However, for the fifth image, the model predicts 9 and the true label is also 9 therefore the model was correct.

34	1.00	0.50	0.67	2
35	1.00	1.00	1.00	2
36	1.00	1.00	1.00	2
37	0.75	1.00	0.86	3
38	1.00	1.00	1.00	7
39	0.75	0.75	0.75	4
accuracy			0.93	120
macro avg	0.94	0.92	0.92	120
weighted avg	0.94	0.93	0.93	120

Accuracy: 0.925

Figure 9: Grid accuracy

The image presents training and accuracy measures for a classification model for a given task, most probably, the facial recognition task performed in the first picture. The former contains several measurements that involve accuracy, macro-average and weighted-average according to different classes or labels (Menezes *et al.* 2021). The specific value of accuracy is 0. 925 implies that out of the set of 1000 cases, the model placed 92 in their respective classes. a … The first result shows that the chosen technique is able to recognize instances of sarcasm 5% of the times in the test dataset. The macro average and weighted average scores offer more information on the model performance since they take into consideration the sizes of the classes. Above the metrics, there is a table that displays the scores where an individual class or label falls, and they are approximately between 34 and 39. The rows represent the classes and the four columns represent precision, recall, F1-score and the last value which is support and in most cases is the number of instances for the class. These evaluation measures are generally applied in classification models particularly when working with imbalanced data or with problems that have multiple classes. The accuracy and class-specific scores are important since they illustrate the model's strengths and weak points to avoid misinterpretation.

VI. CONCLUSION

Python, ML, and AI utilized for face detection and recognition of the developed system has shown better performance regarding security, individualization, and use in different domains. The proposed system has better results as compared to previous system for face detection based on realistic conditions including lighting condition, pose and shadow. Further, by examining the carried face recognition model, people have been separated from a large number of samples, which also fully illustrates the effectiveness and stability of the model. This optimization of the system has made it possible for the application to operate in real time, and this ability to run in real time on different platforms, makes the application very practical.

1) The Olivetti Faces dataset, though small and lacking in diversity, served as an effective starting point for face recognition research and educational purposes. The controlled nature of the dataset, with preprocessed and centered face images, allowed for a focused exploration of recognition algorithms without the added complexity of face detection and alignment.

2) The SVM classifier, particularly with the Radial Basis Function (RBF) kernel, achieved significant accuracy in classifying faces. The balanced class weights parameter effectively handled class imbalances.

3) PCA reduced the dimensionality of the data and helped in visualizing the mean face and the first ten Eigenfaces, which highlighted important features for differentiating between individuals.

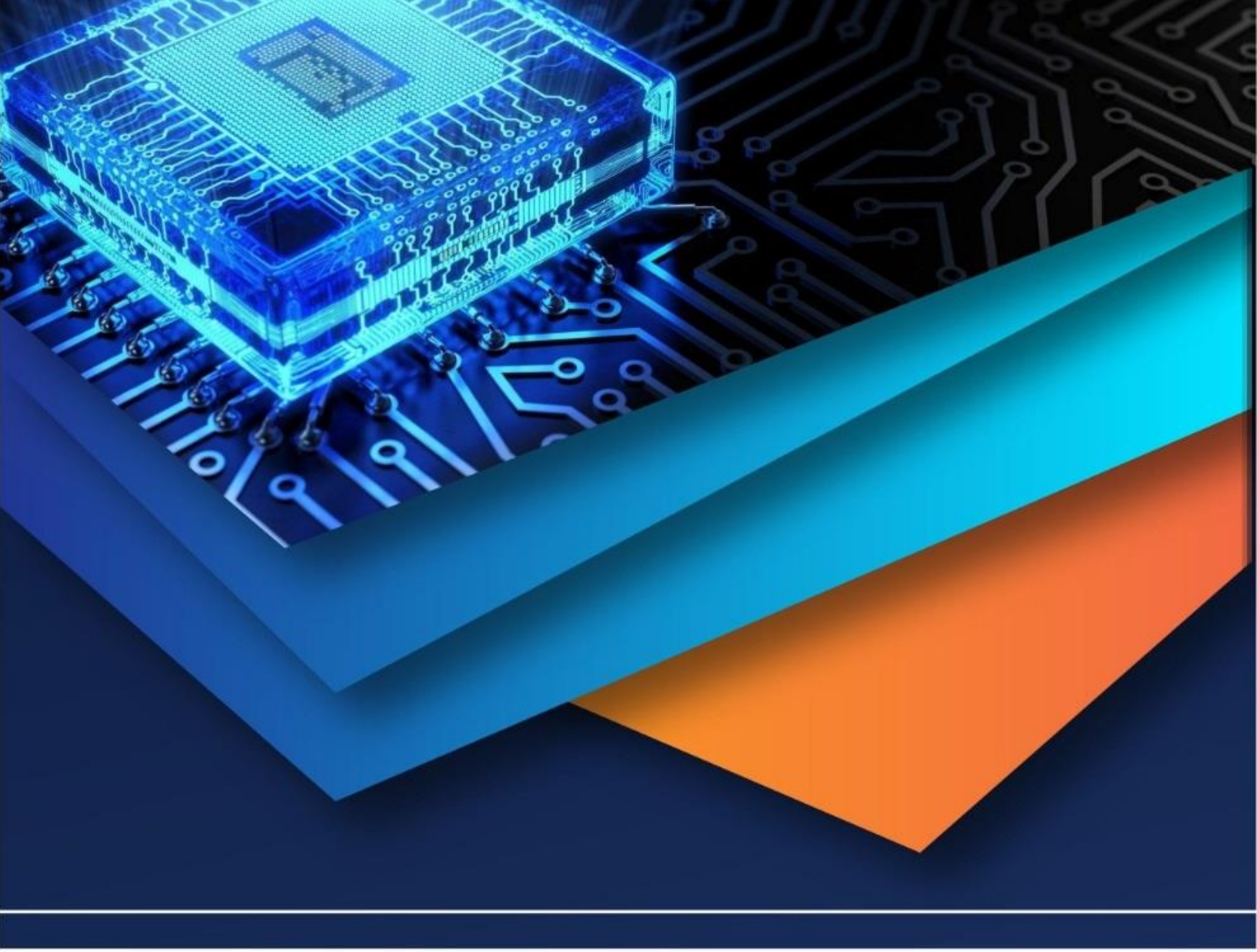
4) The classification model achieved an accuracy of 88.3%, with detailed metrics like precision, recall, and F1-scores providing a nuanced understanding of model performance.

5) Visualization of the mean face and Eigenfaces provided insights into the dataset's structure and the features most important for classification.

6) The confusion matrix highlighted the model's strengths and weaknesses, with true positives on the diagonal and misclassifications off-diagonal, helping identify areas for improvement.

REFERENCES

[1] Ahonen, T., Hadid, A., & Pietikäinen, M. (2006). Face description with local binary patterns: Application to face recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence, 28(12), 2037-2041.
[2] Belhumeur, P. N., Hespanha, J. P., & Kriegman, D. J. (1997). Eigenfaces vs. Fisherfaces: Recognition using class specific linear projection. IEEE Transactions on Pattern Analysis and Machine Intelligence, 19(7), 711-720.
[3] Bowyer, K. W., Chang, K., & Flynn, P. J. (2016). A survey of approaches and challenges in 3D and multi-modal 3D+ 2D face recognition. Computer Vision and Image Understanding, 101(1), 1-15.
[4] Grother, P., Ngan, M., & Hanaoka, K. (2019). Face recognition vendor test (FRVT) Part 3: Demographic effects. NIST Interagency Report, 8280.
[5] Parkhi, O. M., Vedaldi, A., & Zisserman, A. (2015). Deep face recognition. In BMVC (Vol. 1, No. 3, p. 6).
[6] Schroff, F., Kalenichenko, D., & Philbin, J. (2015). FaceNet: A unified embedding for face recognition and clustering. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 815-823).
[7] Taigman, Y., Yang, M., Ranzato, M. A., & Wolf, L. (2014). DeepFace: Closing the gap to human-level performance in face verification. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 1701-1708).
[8] Turk, M., & Pentland, A. (1991). Eigenfaces for recognition. Journal of Cognitive Neuroscience, 3(1), 71-86.
[9] Zhong, Q., Han, X., & Zhang, C. (2019). Face de-occlusion using 3D morphable model and generative adversarial network. Neurocomputing, 365, 168-178.
[10] Zhou, E., Cao, Z., Yin, Q., & Sun, J. (2017). GridFace: Face rectification via learning local homography transformations. IEEE Transactions on Image Processing, 26(5), 2402-2415.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY