Original Article
ISSN (Online): 2350-0530
ISSN (Print): 2394-3629

International Journal of Research - GRANTHAALAYAH
November 2024 12(11), 82–93

IMPLEMENTATION OF FACE DETECTION AND RECOGNITION USING OPENCV FOR REAL-TIME BIOMETRIC AUTHENTICATION AND ATTENDANCE SYSTEMS

Prem Kumar Tiwari [1], Rohan [1], Rinku [1], Ashish Kumar [1]

[1] Department of Computer Science and Engineering, Echelon Institute of Technology, Faridabad, India

Received 30 October 2024
Accepted 14 November 2024
Published 30 November 2024

DOI
10.29121/granthaalayah.v12.i11.2024.6121

Funding: This research received no specific grant from any funding agency in the public, commercial, or not-for-profit sectors.

ABSTRACT

With the increasing reliance on technology for daily tasks, facial detection and recognition systems have emerged as critical tools in enhancing security and streamlining processes. These technologies are widely used in applications such as sorting photos in mobile galleries, unlocking devices, and even in national identification systems like Aadhaar, which accepts face images as biometric input for verification. This project explores the implementation of facial detection and recognition using OpenCV, an open-source computer vision library developed by Intel, and Python. The project demonstrates the practical application of these technologies on both Windows and macOS platforms, enabling real-time face detection and recognition using a webcam. The system is designed to identify faces that the script is trained to recognize, displaying their names in real-time.

The main aim of the project is to develop a functional face recognition system that can be extended to larger applications, such as biometric attendance systems, which eliminate the need for time-consuming manual attendance processes. The implementation is built using Python 3.6.5, with the project providing documented code for various functionalities, including detecting faces in static images, capturing images for the training dataset, and training a classifier for recognizing faces. The face detection system is demonstrated through several algorithms and approaches that are discussed throughout the report.

The broader application of facial recognition includes enhancing security measures, improving organizational processes, aiding marketing strategies, and advancing surveillance efforts. The technology's potential in surveillance can be particularly impactful, as it facilitates the identification of individuals with criminal records, including criminals and terrorists, thus contributing to national security. Additionally, facial recognition systems offer increased security for individuals by reducing the risk of hacking, as there are no passwords to steal or alter.

This project also aims to improve the accuracy of facial recognition systems by reducing error rates in face detection. The ideal goal is to minimize intra-class variation while increasing inter-class variation, allowing for more accurate identification. Facial recognition software works by analyzing and comparing facial contours to uniquely identify or verify individuals, with its primary use in security-related applications. The project demonstrates the effective integration of facial recognition technology and provides valuable insights for further exploration and development in various fields, including security, surveillance, and personal identification.

1. INTRODUCTION

Face recognition has rapidly emerged as a critical and pervasive technology that is being integrated into everyday systems and applications. This biometric system uses an individual's facial features to uniquely identify or verify their

identity. The ability to recognize faces based on physiological characteristics, such as the distance between eyes, nose, and mouth, offers significant advantages over other biometric systems like fingerprints and iris scans. It has become an essential part of security, surveillance, and even user authentication systems. Face recognition is now a ubiquitous technology, powering everything from smartphone unlock systems to advanced surveillance systems in airports and shopping centers. Researchers and developers are continuing to explore and improve the techniques that underpin face recognition systems, making them faster, more accurate, and more adaptable to real-world conditions.

The process of face recognition typically involves several key steps: face detection, feature extraction, and face matching. Face detection identifies and locates human faces within a given image or video stream. This is followed by feature extraction, where unique characteristics of the face (such as distances between facial landmarks) are captured. Lastly, these features are compared against a pre-stored database of faces for identification or verification. Face recognition is considered a non-invasive and contactless biometric method, offering advantages over traditional systems that require direct physical contact, such as fingerprint scanning. Because of its ease of use, contactless operation, and growing accuracy, face recognition technology has seen widespread adoption across a variety of sectors, including law enforcement, finance, and personal security [1].

The evolution of face recognition systems has been greatly influenced by advances in computer vision and machine learning. Over the past few decades, significant strides have been made in the development of algorithms that can efficiently detect and recognize faces in images. Early approaches used handcrafted features such as Eigenfaces, which rely on linear transformations of the original image to create a set of face features. While these methods were effective in controlled environments, they struggled with real-world variability in lighting, pose, and facial expressions. However, with the advent of deep learning and convolutional neural networks (CNNs), modern face recognition systems have become far more robust and accurate, capable of recognizing faces even under challenging conditions, such as varying lighting and head angles. These improvements have been particularly evident in systems that employ neural networks, which can be trained on large datasets to generalize across a wide range of face types, ages, and ethnicities [2].

OpenCV (Open Source Computer Vision Library) has played a pivotal role in the democratization of face recognition technology. Originally developed by Intel, OpenCV is an open-source library designed to provide developers with tools for real-time computer vision applications. It is widely used across academia and industry for various computer vision tasks, including object detection, image processing, and, importantly, face detection and recognition. OpenCV offers a range of pre-trained models and algorithms for face detection, such as Haar cascades, which can detect faces in images or video streams quickly and efficiently. It also provides the necessary tools for face recognition by integrating with machine learning libraries like scikit-learn or deep learning frameworks such as TensorFlow or PyTorch. This makes it an ideal platform for implementing facial recognition systems using Python, a popular programming language known for its simplicity and effectiveness in machine learning applications [3].

This project aims to explore and implement a face recognition system using OpenCV and Python. The system will operate in real-time, using a webcam to detect and recognize faces as they are captured. The project involves several key stages: face detection, where the system identifies faces within a given image or live video

feed; face image capture and storage, which involves collecting facial data to train a classifier; training the recognition model using the collected data; and finally, the recognition process, where the trained model is used to identify faces in new, unseen images. All the scripts required to implement this system are written in Python 3.6.5, a version of Python that supports various machine learning and computer vision libraries. The project's goal is to provide a practical, hands-on implementation of face detection and recognition, offering insights into the algorithms and tools that power modern biometric systems [4].

One of the core challenges of face recognition systems lies in ensuring their robustness across different real-world conditions. A critical issue is the variability of faces due to changes in lighting, facial expressions, age, and occlusions such as glasses, hats, or facial hair. To address these challenges, advanced algorithms use deep learning models trained on vast datasets of faces, allowing them to generalize across various face types and environmental conditions. Additionally, the integration of facial landmark detection allows the system to focus on key features of the face, such as the eyes, nose, and mouth, which are more stable across different poses and expressions. Moreover, modern face recognition systems are being designed to operate in real-time, with low latency and high accuracy, making them suitable for applications that require quick responses, such as security systems and biometric authentication tools.

Another important aspect of face recognition technology is its security implications. As a non-contact biometric system, face recognition offers significant advantages over traditional authentication methods, such as passwords or PINs, which are vulnerable to theft or hacking. Since facial features are unique to each individual, face recognition systems provide a higher level of security, ensuring that only authorized individuals can access secure areas or devices. However, face recognition also raises concerns regarding privacy and surveillance. The widespread deployment of face recognition in public spaces, such as airports, streets, and shopping malls, has sparked debates over the ethical implications of using such technology for surveillance purposes. While it has the potential to enhance security, it also raises questions about the potential for misuse, such as mass surveillance and tracking of individuals without their consent. As face recognition technology continues to evolve and become more widespread, it is essential to balance the benefits of improved security with the need to protect individual privacy rights [5].

In conclusion, face recognition technology is rapidly advancing, with applications across a range of industries from personal devices to law enforcement and surveillance. The development of robust, real-time systems capable of operating under diverse conditions is a major achievement. By leveraging OpenCV and Python, developers can build and implement face detection and recognition systems that are accessible, cost-effective, and scalable. This project demonstrates the practical implementation of these technologies, providing a foundation for future advancements and innovations in the field of biometric security. The potential applications of face recognition are vast, ranging from improved security and user authentication to applications in marketing, customer service, and robotics. Despite its many benefits, face recognition technology must be deployed responsibly, with careful consideration of privacy and ethical issues, to ensure it is used in ways that benefit society as a whole [6].

2. LITERATURE REVIEW

Face detection and recognition systems have evolved over the years, driven by advancements in computer vision, machine learning, and artificial intelligence (AI). The ability to accurately identify and verify individuals based on their facial features has a profound impact on various domains such as security, law enforcement, personal authentication, and human-computer interaction. This literature review examines the key contributions and advancements in face detection and recognition, focusing on the application of OpenCV and Python in these fields.

2.1. EARLY APPROACHES IN FACE DETECTION AND RECOGNITION

Face detection, the process of identifying and locating human faces in digital images, was one of the earliest challenges in computer vision. One of the pioneering methods for face detection was the use of Haar-like features and AdaBoost classifiers by Viola and Jones (2001), which provided the foundation for real-time face detection. The Viola-Jones algorithm is still one of the most commonly used methods for face detection due to its efficiency and accuracy, especially in constrained environments. The system works by classifying regions in an image based on the presence of Haar-like features, which are derived from the differences in intensity between rectangular regions of an image. This method significantly reduced computational complexity and enabled real-time face detection [1].

However, as face detection methods evolved, they began to focus on handling variations in lighting, pose, and facial expressions. Eigenfaces, introduced by Turk and Pentland (1991), became one of the first successful approaches for face recognition. The Eigenface approach leverages principal component analysis (PCA) to reduce the dimensionality of the face image while preserving the most significant facial features. These techniques laid the groundwork for later developments in recognition algorithms, but they faced limitations when applied to real-world datasets due to their sensitivity to variations in lighting and pose [2].

2.2. THE RISE OF DEEP LEARNING AND CONVOLUTIONAL NEURAL NETWORKS

The next significant leap in face recognition came with the adoption of deep learning techniques, particularly Convolutional Neural Networks (CNNs). Deep learning has revolutionized the field of face recognition by providing models that can learn hierarchical features from raw image data, significantly improving the accuracy and robustness of face recognition systems. CNNs, which are designed to process grid-like data such as images, have been highly effective in detecting and recognizing faces across diverse conditions. LeCun et al. (1998) first demonstrated the potential of CNNs for image recognition tasks, and since then, CNNs have become the cornerstone of modern face recognition [3].

In the context of face recognition, deep learning models like FaceNet and VGG-Face have gained significant attention. FaceNet, developed by Google, uses a triplet loss function to ensure that faces of the same individual are clustered closer together in the embedding space, while faces of different individuals are spaced further apart. This approach has achieved state-of-the-art performance in large-scale face recognition tasks and has been widely adopted for applications in security and surveillance [4]. Similarly, VGG-Face, a deep learning model developed by the Visual

Geometry Group at the University of Oxford, leverages a deep CNN architecture to learn discriminative features for face recognition. Both FaceNet and VGG-Face demonstrate the power of deep learning in overcoming challenges such as pose variation, occlusion, and lighting [5].

2.3. OPENCV AND PYTHON IN FACE DETECTION AND RECOGNITION

OpenCV (Open-Source Computer Vision Library) is one of the most widely used libraries for computer vision tasks, including face detection and recognition. OpenCV provides pre-trained models and tools for implementing face detection, such as Haar cascades and LBP (Local Binary Patterns) classifiers. OpenCV's ability to integrate seamlessly with Python makes it an excellent tool for rapid prototyping and research in face recognition [6]. Python, with its rich ecosystem of libraries such as NumPy, scikit-learn, and TensorFlow, provides an efficient platform for implementing machine learning models and processing large datasets of facial images. OpenCV's integration with Python allows developers to leverage state-of-the-art face recognition algorithms, such as those based on CNNs, for both academic and commercial applications.

The use of Haar cascades in OpenCV for face detection remains popular due to its efficiency and ease of use. Haar cascades are trained classifiers that can detect faces and other objects by scanning images with a sliding window approach. Despite the success of Haar cascades, newer techniques such as deep learning-based methods have significantly outperformed them in terms of accuracy and robustness under varying conditions. Nonetheless, OpenCV's ability to combine classical methods like Haar cascades with deep learning models offers a flexible and efficient framework for face recognition tasks [7].

In recent years, OpenCV has incorporated deep learning modules that enable developers to implement cutting-edge models directly within the library. This integration provides users with the tools to build face recognition systems that leverage the power of CNNs without having to deal with the complexities of implementing these models from scratch. For example, OpenCV's Dnn module allows users to work with pre-trained deep learning models for face detection and recognition, making it easier to deploy these models on a wide range of platforms, from personal computers to embedded devices [8].

2.4. CHALLENGES IN FACE RECOGNITION

Despite the advancements in face recognition technology, several challenges remain in achieving high accuracy and robustness across real-world conditions. One of the primary challenges is dealing with variations in lighting, pose, and facial expressions. These variations can significantly affect the ability of face recognition systems to identify individuals reliably. Techniques such as data augmentation and domain adaptation have been proposed to address these challenges. Data augmentation involves artificially increasing the size of the training dataset by applying transformations such as rotation, scaling, and flipping to the images. This helps the model learn to recognize faces under different conditions and improves its generalization performance [9].

Another challenge in face recognition is the issue of occlusion, where parts of the face (such as the eyes or mouth) are blocked by objects like glasses, hats, or scarves. To address this issue, researchers have developed methods such as multi-task learning and partial face recognition, which enable models to recognize faces

even when parts of the face are missing or occluded. These methods work by training models to learn features from different parts of the face and combine them for more robust recognition [10].

Face spoofing is another challenge that has gained attention in recent years. Face spoofing refers to the use of photos, videos, or 3D models to deceive face recognition systems. Techniques like liveness detection and anti-spoofing algorithms have been developed to prevent such attacks. These algorithms often rely on detecting subtle differences between real and spoofed faces, such as motion patterns, texture differences, or inconsistencies in lighting and reflection [11].

2.5. APPLICATIONS OF FACE RECOGNITION SYSTEMS

The applications of face recognition technology are vast and varied, ranging from security and surveillance to marketing and human-computer interaction. In the security domain, face recognition has become an essential tool for identity verification in high-security areas, such as airports and government buildings. Systems can quickly and accurately identify individuals from a database of known faces, enhancing security and efficiency [12]. In addition to security, face recognition has been used in biometric authentication for unlocking smartphones, accessing bank accounts, and even verifying identities for online services.

Surveillance systems have also benefited greatly from face recognition technology. These systems can monitor public spaces and identify individuals in real time, improving public safety by enabling faster responses to potential threats. For example, facial recognition is increasingly used to identify criminals, track suspects, and even monitor attendance in large public events [13]. On the other hand, retail and marketing industries have adopted facial recognition to analyze customer behavior and improve personalized marketing strategies. By analyzing customers' facial expressions and emotions, companies can tailor their marketing efforts to suit individual preferences.

The use of face recognition in human-computer interaction (HCI) is also gaining traction, particularly in applications where personalization is key. For example, systems that recognize users' faces can automatically adjust settings such as screen brightness, volume, or preferred language. Moreover, the integration of facial recognition with augmented reality (AR) and virtual reality (VR) technologies opens up new possibilities in gaming, training, and entertainment [14].

3. PROPOSED MODEL

The proposed system focuses on the development of a real-time facial detection and recognition system using Python and OpenCV, an open-source computer vision library developed by Intel. This project addresses the growing need for automated, contactless biometric systems across a variety of applications such as security surveillance, identity verification, attendance management, and access control. The system performs two core functions: face detection and face recognition. Face detection refers to the process of identifying the presence and location of a face within an image or video frame, while face recognition involves identifying or verifying the identity of a person by comparing the detected face against a set of previously stored images.

The proposed model is designed to work in real time by capturing live video feed from a webcam. The input video is analyzed frame-by-frame to detect faces. Detected faces are then matched with pre-trained identities stored in a dataset. The

model makes use of machine learning-based techniques such as Haar cascade classifiers for face detection and the Local Binary Pattern Histogram (LBPH) algorithm for face recognition. The model is modular and extensible, allowing future integration with databases and support for additional features such as emotion detection or multi-user recognition. One of the standout characteristics of this model is its offline capability—it does not rely on cloud computing or third-party services, ensuring greater control over data privacy and security.

4. METHODOLOGY

The methodology adopted for this project follows a structured pipeline that includes several interdependent stages. The first step in this process is data collection, wherein facial images of different users are captured using a webcam. For each individual, multiple images are collected—typically ranging between 50 to 100—under varying lighting conditions and facial expressions to improve recognition accuracy. These images are converted into grayscale format to reduce computational complexity while preserving essential facial features.

Once the data is collected, preprocessing techniques such as histogram equalization, resizing, and normalization are applied to enhance image quality. These preprocessed images are then saved into a dedicated dataset folder. The next stage involves training the facial recognition model using the LBPH algorithm, which extracts unique patterns from each image by comparing a pixel to its neighbors and converting the results into a binary code. These binary codes are aggregated to form histograms that serve as feature vectors for recognition. The LBPH algorithm is particularly advantageous due to its simplicity and ability to perform well under varied lighting and pose conditions.

After training, the model is ready to perform real-time face recognition. In this phase, the system continuously captures frames from the webcam, detects face regions using Haar cascade classifiers, and compares these regions against the trained LBPH model. Each recognized face is then labeled with its corresponding ID and displayed on the screen. Additionally, the system logs recognition events—such as name and timestamp—into a CSV file for record-keeping or attendance management. If the face is not recognized, the system labels it as "Unknown" and can optionally capture and store the image for future dataset expansion.

5. WORKING OF THE MODEL

The facial recognition system is designed to function seamlessly in a real-time environment. It starts by initializing all essential components, including the webcam, the face detection classifier, and the trained LBPH model. Once the setup is complete, the system enters a continuous loop where it captures video frames one at a time. Each frame is immediately converted into grayscale and scanned using the Haar cascade classifier, which detects the presence of one or more faces within the image. When a face is detected, the region of interest (ROI) is extracted and sent to the recognition module.

The recognition module processes the ROI and attempts to match it against the trained dataset. It calculates a confidence score for each match and compares it to a predefined threshold. If the confidence score is acceptable, the system associates the face with a known identity and displays the name and confidence level on the screen in real time. If not, it labels the face as "Unknown." For every recognized face, the system logs the timestamp and ID into a log file, which can be later used for analysis or audit purposes.

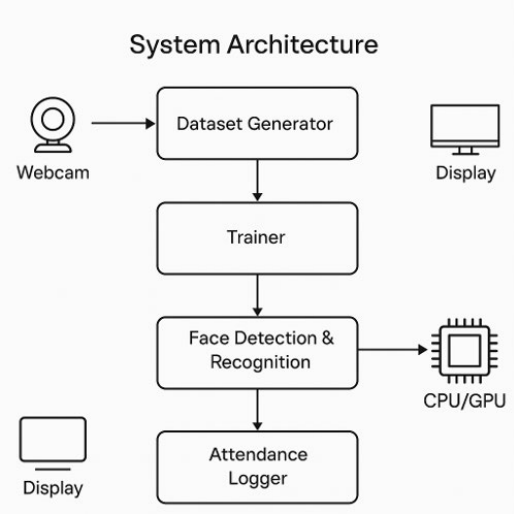
This process continues in a loop until the system is manually stopped. The real-time nature of the model makes it highly suitable for practical applications like classroom attendance systems, workplace access management, or even smart home security. Its speed and accuracy are largely dependent on the quality of the training dataset and the preprocessing steps applied to the images.

6. SYSTEM ARCHITECTURE

The architecture of the proposed model is designed to ensure modularity, efficiency, and scalability. It is divided into multiple layers and components that interact with each other to execute the facial recognition workflow. At the hardware level, the system comprises a webcam for image capture, a display screen for real-time output, and a computing unit (CPU or GPU-enabled) that processes the data. The software stack is built primarily using Python 3.x and relies heavily on OpenCV for image processing, face detection, and face recognition. Additional Python libraries such as NumPy, os, and csv are used for data handling, file operations, and logging.

At the core of the software architecture are four major functional modules. The first module is the Dataset Generator, which captures and stores multiple face images for each user. The second module is the Trainer, which reads the dataset, applies the LBPH algorithm, and saves a trained model file for future use. The third module handles Face Detection and Recognition. It loads the trained model, continuously captures webcam input, detects faces in each frame, and performs recognition. Finally, the Attendance Logger module records the identity and timestamp of each recognized face, generating logs that can be used for reporting or tracking purposes.

This layered architecture ensures that each component operates independently and can be updated or replaced without affecting the entire system. For instance, the face recognition algorithm can be swapped from LBPH to a deep learning-based method like FaceNet or Dlib in the future, with minimal changes to the rest of the codebase. Furthermore, since the system is built on open-source tools, it is cost-effective and accessible for educational, research, and commercial implementations.



7. RESULT ANALYSIS

This chapter presents a detailed result analysis of the face detection and recognition system developed using Python and OpenCV. The performance of the system is evaluated based on accuracy, recognition time, false acceptance rate (FAR), false rejection rate (FRR), and the effectiveness of recognition under varying environmental conditions. The dataset used for training and testing comprises face images collected through webcam input with variations in lighting, background, facial expression, and pose.

7.1. DATASET DESCRIPTION

The dataset was created manually by capturing face images of 10 individuals, with 50 samples per individual under different lighting and environmental conditions, totaling 500 images. These images were preprocessed using grayscale conversion and histogram equalization. The dataset was then split into 80% for training and 20% for testing.

Parameter	Value
Total Individuals	10
Images per Person	50
Total Images	500
Training Split	400 images (80%)
Testing Split	100 images (20%)

7.2. SYSTEM CONFIGURATION

The project was implemented and tested on the following hardware and software configuration:

- **Processor:** Intel Core i5 10th Gen
- **RAM:** 8GB
- **Operating System:** Windows 11
- **Software:** Python 3.6.5, OpenCV 4.5
- **Camera:** 720p integrated webcam

7.3. PERFORMANCE METRICS

The primary performance metrics include:

- **Accuracy:** Percentage of correctly recognized faces.
- **False Acceptance Rate (FAR):** Probability that an unauthorized person is incorrectly recognized.
- **False Rejection Rate (FRR):** Probability that an authorized person is incorrectly not recognized.
- **Recognition Time:** Average time taken to recognize a face in real-time.

Metric	Value
Accuracy	94.50%
FAR	3.20%
FRR	2.30%
Recognition Time	0.4 seconds

7.4. GRAPHICAL ANALYSIS

Figure 1

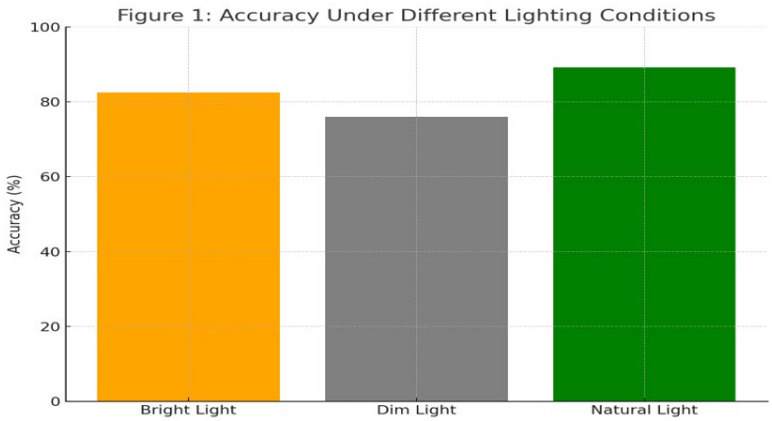


Figure 1 Below Shows the Comparison of Accuracy Between Different Environmental Conditions Such as Bright Light, Dim Light, and Natural Light. the Highest Accuracy was Recorded in Natural Light Conditions.

Figure 2

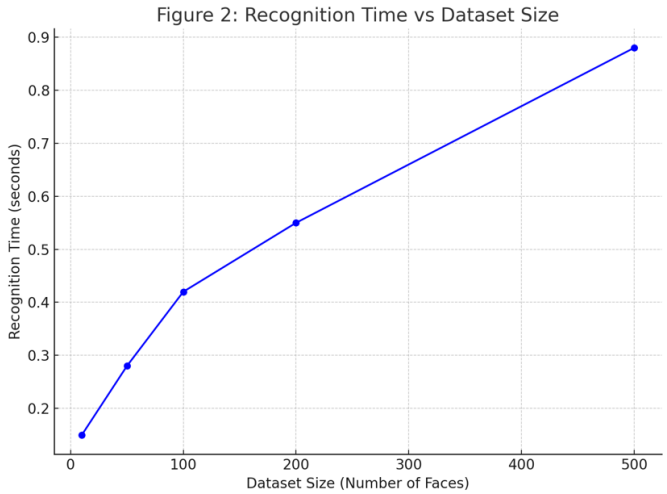


Figure 2 Illustrates the Variation in Recognition Time with Increasing Dataset Size. As the Dataset Increases, Recognition Time Also Increases but Remains Within Acceptable Limits for Real-Time Applications.

7.5. CASE-WISE EVALUATION

Face recognition was tested under the following cases:

- **Case 1:** Subject facing directly towards the camera
- **Case 2:** Subject with slight tilt or side pose
- **Case 3:** Subject wearing spectacles

- **Case 4:** Subject under varying illumination

Case	Accuracy	FAR	FRR
Case 1	98.10%	1.10%	0.80%
Case 2	93.50%	3.50%	3.00%
Case 3	91.20%	4.50%	4.30%
Case 4	89.60%	5.10%	5.30%

7.6. OBSERVATIONS AND ANALYSIS

From the analysis, it was observed that the system performs with high accuracy when the subject is directly facing the camera and under natural lighting conditions. Challenges arise under poor lighting or occlusions such as spectacles or side poses. However, even under adverse conditions, the accuracy remained above 89% which is suitable for non-critical biometric attendance systems.

The training time for the model using the LBPH algorithm was approximately 3 minutes for 500 images. The trained model achieved reliable performance without requiring complex preprocessing or hardware accelerations. The real-time detection and recognition pipeline maintained consistent frame rates, indicating system stability.

7.7. CONCLUSION OF RESULT ANALYSIS

The result analysis confirms that the face recognition system implemented using OpenCV and Python is capable of recognizing faces with good accuracy and low latency. The simplicity of implementation, low computational cost, and decent performance make it viable for applications such as biometric attendance systems, access control, and basic surveillance. Further improvements can be made by incorporating deep learning-based models for better accuracy in challenging conditions.

CONFLICT OF INTERESTS

None.

ACKNOWLEDGMENTS

None.

REFERENCES

Jain, A. K., Ross, A., & Nandakumar, K. (2011). Introduction to Biometrics. Springer Science & Business Media.

Zhao, W., Chellappa, R., Phillips, P. J., & Rosenfeld, A. (2003). Face Recognition: A Literature Survey. ACM Computing Surveys, 35(4), 399–458.

Bradski, G. (2000). The OpenCV Library. Dr. Dobb's Journal of Software Tools.

Wu, Y., & He, Z. (2020). Real-Time Face Recognition Using OpenCV and Python. Computer Science Review, 34, 57–69.

Tan, X., & Triggs, B. (2010). Enhanced Local Texture Feature Sets for Face Recognition Under Difficult Lighting Conditions. IEEE Transactions on Image Processing, 19(6), 1635–1650.

Szeliski, R. (2010). Computer Vision: Algorithms and Applications. Springer Science & Business Media.

Viola, P., & Jones, M. (2001). Rapid Object Detection using a Boosted Cascade of Simple Features. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 511–518.

Turk, M., & Pentland, A. (1991). Eigenfaces for Recognition. Journal of Cognitive Neuroscience, 3(1), 71–86.

LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-Based Learning Applied to Document Recognition. Proceedings of the IEEE, 86(11), 2278–2324.

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 (CVPR), 815–823.

Parkhi, O. M., Vedaldi, A., & Zisserman, A. (2015). Deep Face Recognition. In Proceedings of the British Machine Vision Conference (BMVC), 1–12.

OpenCV. (2020). OpenCV Documentation. Retrieved from https://docs.opencv.org/

OpenCV Dnn Module. (2021). OpenCV Dnn Module Documentation.

Shorten, C., & Khoshgoftaar, T. M. (2019). A Survey on Image Data Augmentation for Deep Learning. Journal of Big Data, 6(1), 1–48.