IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 4, July 2026
Impact Factor: 8.2

Real Time Face Recognition Using Deep Learning

Mohammed Saad[1] and Prof. Kusuma N[2]
[*1]Student, Department of Computer Applications
[*2]Professor, Department of Computer Applications
Maharaja Institute of Technology Mysore, Mandya, Karnataka, India.

Abstract: *Reliable identity verification remains a foundational requirement across educational institutions, workplaces, and small businesses, yet the conventional mechanisms used to satisfy it—passwords, identity cards, and manual attendance registers—remain vulnerable to loss, duplication, sharing, and human error. Biometric facial recognition offers a contactless alternative that removes dependence on something a user must carry or remember, but many contemporary face-recognition pipelines rely on deep-learning architectures that demand substantial computing power, large annotated datasets, and dedicated hardware, placing them out of reach for resource-constrained deployments. This paper presents a real-time facial recognition system built on lightweight, classical computer-vision techniques: the Haar Cascade classifier for face detection and the Local Binary Pattern Histogram (LBPH) algorithm for recognition. The system captures facial images of registered users through a standard webcam, organises them into a per-user training dataset, and trains an LBPH recognizer that is subsequently used to identify individuals from a live video stream in real time, flagging any face that falls below a confidence threshold as an unknown person. Implemented in Python using OpenCV, NumPy, and Pillow, the application requires no specialised hardware beyond an ordinary webcam and runs comfortably on consumer-grade laptops and desktops. Functional testing across registration, dataset creation, detection, training, recognition, and unknown-person handling produced consistent results, with the system achieving an estimated recognition accuracy in the region of 85–90 percent and a precision of approximately 94 percent under adequate lighting. The paper details the system's architecture, module-level design, implementation, and testing outcomes, and closes with a discussion of limitations—principally sensitivity to lighting and pose—and directions for future enhancement, including liveness detection and cloud-based deployment..*

Keywords: *Biometric authentication, facial recognition, OpenCV, Local Binary Pattern Histogram, Haar Cascade, computer vision, real-time systems, contactless identification*

1. INTRODUCTION

Identity verification underpins security and accountability in almost every institutional setting, from classroom attendance to controlled access at workplaces and small businesses. The mechanisms most widely relied upon today—passwords, PIN codes, identity cards, and manual verification—remain vulnerable to duplication, theft, forgetting, and simple human error, and the operational cost of managing them grows with the size of the user population. In an environment where automation is increasingly the norm, biometric authentication has emerged as a more dependable means of confirming a person's identity, and among biometric modalities, facial recognition is particularly attractive because it requires no physical contact and depends only on a camera to capture the subject's face.

The wide availability of high-resolution cameras and mature computer-vision libraries has made face recognition an affordable option for educational institutions, offices, healthcare facilities, and security organisations. However, a considerable share of recent work in this space leans on deep convolutional architectures that, while accurate, demand GPUs, large labelled datasets, and non-trivial training infrastructure—resources that are often unavailable to a small college laboratory, a neighbourhood business, or an individual developer. This creates a gap for a system that is accurate enough for everyday use, yet light enough to run on ordinary consumer hardware.

DOI: 10.48175/IJARSCT-37908

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429 Volume 6, Issue 4, July 2026 Impact Factor: 8.2

This paper addresses that gap with a real-time facial recognition system built around two classical, computationally inexpensive techniques: the Haar Cascade classifier for locating a face within a video frame, and the Local Binary Pattern Histogram (LBPH) algorithm for recognising the identified face against a small, locally stored dataset of registered users. Because detection is handled by a pre-trained Haar Cascade model, faces can be located quickly without any additional training, and because LBPH encodes local texture rather than raw pixel intensity, the recognizer tolerates moderate variation in lighting better than earlier appearance-based methods such as Eigenfaces. Python was selected as the implementation language for its simplicity and its mature computer-vision ecosystem; combined with NumPy for numerical operations and OpenCV for image processing, the result is a lightweight pipeline that performs without requiring expensive hardware or specialised computational resources.

The remainder of this paper is organised as follows. Section II reviews related work in classical and deep-learning-based face recognition. Section III states the problem being addressed. Section IV sets out the objectives and scope of the system. Section V describes the system architecture and module-level methodology. Section VI details the implementation. Section VII reports testing outcomes and results, and Section VIII concludes the paper and outlines directions for future work.

II. RELATED WORK

Research on automated face recognition spans several decades and a range of algorithmic families, from classical appearance- and texture-based methods to modern deep-learning pipelines. Viola and Jones proposed a cascade-classifier approach to object detection built on Haar-like features and the AdaBoost learning algorithm; the method's low computational overhead and strong real-time performance have made it a standard choice for face detection in resource-constrained applications, and it is the detection technique adopted in the present work [1]. For the recognition stage, Ahonen, Hadid, and Pietikäinen introduced the Local Binary Pattern Histogram approach, which encodes local texture information from regions of the face into histograms; this representation has proven markedly more robust to illumination changes than earlier holistic techniques [2].

Principal Component Analysis (PCA), commonly associated with the Eigenfaces method, reduces the dimensionality of facial images by identifying the components that capture the greatest variance across a training set [3]. PCA lowers computational cost but its recognition accuracy degrades noticeably under changes in lighting, facial expression, and head pose, motivating the search for more texture-robust alternatives such as LBPH. At the other end of the complexity spectrum, convolutional neural networks (CNNs) automate hierarchical feature extraction directly from raw pixel data and, in modern architectures, achieve very high recognition accuracy even under challenging conditions [4]. This accuracy, however, comes at the cost of substantial computing power, large labelled training sets, and dedicated GPU hardware—requirements that are difficult to justify for a small-scale deployment such as a single classroom or a small office.

A further strand of literature situates face recognition within IoT, security, and surveillance contexts, where the technology's principal appeal is that it identifies or verifies a user without requiring any deliberate action beyond simply being present in front of a camera. Deployments of this kind typically depend on cloud platforms and mobile applications to achieve scale, which introduces recurring infrastructure cost and network dependency that a purely local, classroom-or office-scale system does not need to bear.

A. Research Gap

Although a considerable body of work has advanced face recognition accuracy through deep learning, comparatively little attention has been paid to systems that are simple, inexpensive, and easy to deploy in small institutional settings such as colleges and small businesses. Most current approaches assume access to substantial computing resources, specialised tools, and large training datasets, none of which are guaranteed in these environments. The system presented in this paper targets that gap directly: by pairing the Haar Cascade detector with the LBPH recognizer, it achieves recognition accuracy sufficient for everyday authentication tasks while remaining light enough to run comfortably on a typical laptop, without a GPU or cloud dependency.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 4, July 2026
Impact Factor: 8.2

III. PROBLEM STATEMENT

Educational institutions, offices, and small businesses all require a dependable means of verifying identity, yet the mechanisms in common use today each carry a distinct weakness. Password-based systems are susceptible to sharing, forgetting, and compromise; identity cards can be lost, stolen, or forged; manual attendance registers are slow and prone to proxy entries and record manipulation; and fingerprint scanners lose reliability when a finger is dirty, wet, or injured. Deep-learning-based facial recognition can outperform all of these on raw accuracy, but its dependence on costly CPUs, GPUs, and large training datasets makes it impractical for many of the settings that would benefit most from an automated, contactless alternative.

The problem addressed in this paper, therefore, is to design and implement a facial recognition system that is reliable enough for everyday identity verification, computationally light enough to run on ordinary desktops and laptops, and simple enough to deploy without specialised infrastructure—while continuing to function correctly under typical indoor lighting conditions and correctly flagging individuals who are not part of the registered user base.

IV. OBJECTIVES AND SCOPE

A. Objectives

- Develop an OpenCV-based, real-time facial recognition application in Python.
- Accurately detect human faces in a live webcam feed using the Haar Cascade classifier.
- Capture and organise facial images of registered users into a structured training dataset.
- Train an LBPH face recognizer on the collected dataset.
- Identify registered individuals in real time by comparing live facial input against the trained model.
- Correctly flag individuals who do not match any registered user as an unknown person.
- Keep the system's footprint small enough to run on typical consumer hardware without additional processing power.
- Demonstrate a practical, contactless application of computer vision and biometric authentication to everyday identification problems.

B. Scope

The system is designed to verify the identity of registered users by analysing facial features captured through a standard webcam, using the Haar Cascade algorithm for face detection and the LBPH recognizer for matching against the trained dataset. Registered users are identified by name; anyone not present in the system is reported as unknown, and identification proceeds automatically in real time without manual intervention. Because the system depends only on a webcam and does not require physical identity cards, passwords, or dedicated biometric hardware, it is well suited to schools, offices, laboratories, libraries, and small businesses that need a fast, low-cost way to confirm identity. Its lightweight design allows it to run on ordinary desktops and laptops without specialised graphics hardware or cloud services, while the modular structure of its components leaves room for future upgrades—improved recognition models, cloud-hosted datasets, mobile clients, or additional biometric factors—without requiring the system to be rebuilt from scratch.

V. SYSTEM ARCHITECTURE AND METHODOLOGY

The system is organised into six loosely coupled modules—dataset collection, face detection, dataset management, model training, face recognition, and result display—each responsible for a single, well-defined task. This separation keeps the codebase easy to debug, test, and extend: a change to how faces are detected, for instance, does not require any change to how recognition results are displayed. Fig. 1 illustrates the overall system architecture, and Fig. 2 summarises the end-to-end process flow.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 4, July 2026
Impact Factor: 8.2

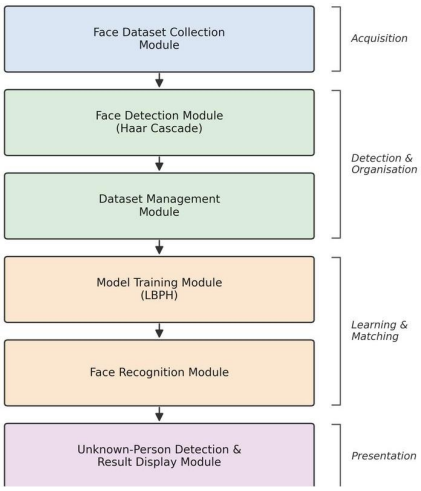

Fig. 1. System architecture showing the six interconnected modules, from face dataset collection through to result display.

A. Face Dataset Collection Module

This module captures multiple facial images of each registered user through a webcam and stores them in a per-user directory, associating every image with the corresponding user ID for later use in training.

B. Face Detection Module

Live webcam frames are converted to grayscale and passed through the Haar Cascade classifier, which locates facial regions, draws a bounding rectangle around each detected face, and extracts the facial region for downstream processing.

C. Dataset Management Module

All facial images collected during registration are organised, stored, and retrievable on demand, allowing the training module to read a clean, structured dataset and allowing existing user records to be updated as needed.

D. Model Training Module

Training images are read, converted to grayscale, and processed to extract local binary pattern features, from which the LBPH recognizer builds per-user facial histograms. The resulting trained model is persisted to disk for reuse during recognition.

E. Face Recognition Module

During live operation, each detected face is compared against the trained model, and a confidence score is computed. If the score exceeds the recognition threshold, the associated user name is returned; otherwise the module reports that no match was found.

F. Unknown Person Detection and Result Display

Whenever the confidence score falls below the recognition threshold, the system labels the detected face as an unknown person rather than guessing an identity. The result display module then overlays the outcome—either the recognized user's name or the label “Unknown Person”—directly on the live video frame, together with the bounding rectangle drawn during detection.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 4, July 2026
Impact Factor: 8.2

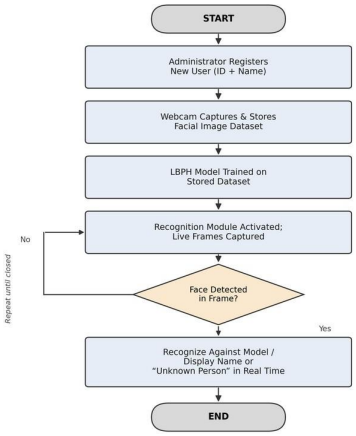

Fig. 2. End-to-end operational flow, from user registration through continuous real-time recognition and result display.

G. End-to-End Flow

The complete operational sequence proceeds as follows: (1) the administrator launches the application and registers a new user with a unique ID and name; (2) the webcam captures a series of facial images for that user, which are stored in the dataset directory; (3) once sufficient users are registered, the administrator triggers training, which extracts LBPH features from the dataset and saves a trained model file; (4) the recognition module is activated, continuously capturing webcam frames; (5) each frame is passed through face detection and, if a face is found, through recognition against the trained model; and (6) the result—either a recognized user's name or an unknown-person label—is displayed on screen in real time, with the cycle repeating until the application is closed.

VI. IMPLEMENTATION DETAILS

A. Technology Stack

- Programming Language: Python 3.x
- Computer Vision: OpenCV (image processing, face detection, webcam access)
- Numerical Computation: NumPy (image array manipulation)
- Image Handling: Pillow (image reading and processing)
- Face Detection: Haar Cascade frontal-face classifier (pre-trained XML model)
- Face Recognition: OpenCV's LBPH face recognizer
- Development Environment: Visual Studio Code on Windows 10/11

B. Project Structure

The codebase is organised into clearly separated directories so that datasets, the trained model, and the pre-trained cascade classifier remain distinct from the application's source files: a dataset directory holding one sub-folder of facial images per registered user; a trainer directory holding the trained LBPH model file; a haarcascade directory holding the pre-trained frontal-face classifier; and three top-level scripts—one for registration, one for training, and one for recognition—alongside a requirements file listing the project's Python dependencies. This layout keeps data, model artefacts, and code independent of one another, which simplifies both debugging and future extension.

C. Hardware and Software Requirements

The application has modest requirements: a processor equivalent to an Intel Core i3 or better, at least 4 GB of RAM (8 GB recommended for a comfortable runtime), roughly 500 MB of free storage, a standard integrated or external webcam, and a stable environment for running Python 3.x with OpenCV, NumPy, and Pillow installed. No specialised graphics hardware or cloud connectivity is required, which keeps the system accessible on ordinary consumer-grade laptops and desktops and economically viable for educational institutions and small businesses.

DOI: 10.48175/IJARSCT-37908

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 4, July 2026
Impact Factor: 8.2

VII. RESULTS AND DISCUSSION

The system was implemented and tested across its constituent modules—user registration, dataset creation, face detection, model training, face recognition, and unknown-person handling—using unit, integration, system, performance, and informal user-acceptance testing. Every module passed its corresponding test case: registration successfully captured facial images for a new user, the dataset was created and stored correctly, the Haar Cascade classifier reliably detected faces in the live video stream, the LBPH recognizer trained without error, registered users were correctly identified, and faces that did not match any registered user were consistently reported as unknown.

A. Recognition Performance

Under adequate lighting and with a reasonably sized set of training images per user, the system achieved an estimated recognition accuracy of approximately 85–90 percent, a precision of around 94 percent (it rarely mislabels an unknown individual as a registered user), and a recall of approximately 85 percent (most registered users are correctly recognised). Webcam initialisation was fast, recognition results were displayed with minimal perceptible delay, and CPU and memory usage remained within the range typical of an ordinary personal computer throughout continuous operation.

B. Limitations Observed

- Recognition accuracy decreased under inadequate or uneven lighting.
- Side-profile faces were harder to distinguish reliably than frontal faces.
- Sunglasses or face masks occasionally interfered with correct identification.
- Each registered user required a sufficient number of training images for reliable recognition.
- Overall recognition quality was sensitive to the resolution of the webcam used.

These limitations are broadly consistent with the well-documented sensitivities of Haar Cascade and LBPH to illumination and pose, and none of them prevented the system from performing reliably within the conditions it was designed for—typical indoor lighting and largely frontal facial presentation.

VIII. CONCLUSION AND FUTURE SCOPE

This paper presented the design, implementation, and testing of a real-time facial recognition system built on the Haar Cascade classifier for detection and the LBPH algorithm for recognition, implemented in Python with OpenCV, NumPy, and Pillow. Testing across all functional modules confirmed that the system reliably detects faces, trains a recognition model from a locally collected dataset, correctly identifies registered users, and correctly flags individuals outside the registered set as unknown, all while running on ordinary consumer hardware without requiring a GPU or cloud infrastructure. The results support the broader claim that lightweight, classical computer-vision algorithms remain a practical and cost-effective choice for academic projects and small-scale deployments where high-end hardware is neither available nor justified.

Several directions for future work follow naturally from the limitations observed during testing. These include exploring deep-learning-based recognition to improve accuracy and scalability; incorporating liveness detection to guard against spoofing via printed photographs or displayed images; migrating from local file storage to a cloud-based database to support larger datasets and remote access; extending the system to mobile platforms; combining facial recognition with a second authentication factor such as a PIN or OTP for higher-security contexts; integrating the recognizer with attendance-management software; and improving robustness under poor lighting, partial occlusion, and extreme pose. Taken together, these enhancements would extend the system from a functional academic prototype toward a deployment-ready platform for smart-campus and small-business identity verification.

We would like to express our sincere gratitude to our project guide for the valuable guidance, support, and encouragement provided throughout the course of this project. We convey our thanks to the faculty members of the Department of Master of Computer Applications, Maharaja Institute of Technology Mysore, for their continued support and encouragement. We also extend our thanks to the management of the institute for providing the necessary facilities

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 4, July 2026
Impact Factor: 8.2

and resources for the completion of this work. Finally, we thank our classmates and friends for their cooperation and support throughout this project.

REFERENCES

[1] P. Viola, M. Jones, "Rapid Object Detection using a Boosted Cascade of Simple Features", Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2001, pp. 511-518.
[2] T. Ahonen, A. Hadid, M. Pietikäinen, "Face Recognition with Local Binary Patterns", Proceedings of the European Conference on Computer Vision (ECCV), 2004, pp. 469-481.
[3] R. C. Gonzalez, R. E. Woods, "Digital Image Processing", 4th ed., Pearson Education, 2018.
[4] R. Szeliski, "Computer Vision: Algorithms and Applications", 2nd ed., Springer, 2022.
[5] S. Z. Li, A. K. Jain, Eds., "Handbook of Face Recognition", 2nd ed., Springer, 2011.
[6] I. Goodfellow, Y. Bengio, A. Courville, "Deep Learning", MIT Press, Cambridge, MA, USA, 2016.
[7] OpenCV Development Team, "OpenCV Documentation", [Online]. Available: https://docs.opencv.org/.
[8] Python Software Foundation, "Python 3 Documentation", [Online]. Available: https://docs.python.org/3/.
[9] NumPy Developers, "NumPy User Guide", [Online]. Available: https://numpy.org/doc/.
[10] Pillow Developers, "Pillow Documentation", [Online]. Available: https://pillow.readthedocs.io/.