

Multimodal Biometric Attendance System Using Convolutional Neural Networks with Face and Fingerprint Fusion

Priyanshu*, Dhawal, Shweta, Swarup, Abhishek, Vivek, Rahul Kawariya

Department of Computer Science and Engineering, G H Rasoni University, Amravati, Maharashtra, India.

ABSTRACT

Manual roll-call and single-trait attendance systems such as RFID cards or lone fingerprint scanners are slow and can be defeated through proxy attendance, in which one person marks attendance for an absent friend. This paper presents a Multimodal Biometric Attendance System that combines live face recognition with fingerprint recognition using Convolutional Neural Networks (CNN). Two CNN branches, one for face images and one for fingerprint images, are trained separately and then joined by a feature-level fusion model that makes a single combined identity decision. Face images were captured from team members through a webcam, and fingerprint samples were taken from the public SOCOFing dataset because no physical scanner was available. All images are converted to 100×100 grayscale and expanded with OpenCV-based data augmentation to reduce overfitting. The system is deployed as a Flask web application with a live camera interface, an SQLite attendance database, proof-photo capture, and a password-protected administrator dashboard with search, export, editing, analytics, and low-attendance alerts. A face-first decision strategy with a confidence and margin check was introduced after early versions misidentified users. The best fusion model reached about 99% validation accuracy. The framework provides a practical, end-to-end and extensible approach to secure automated attendance that can later accept a real fingerprint scanner.

Keywords: Multimodal Biometrics, Attendance System, Convolutional Neural Network, Face Recognition, Fingerprint Recognition, Feature-Level Fusion, Flask, Deep Learning

INTRODUCTION

Attendance recording is a routine but important activity in colleges and organizations. Most institutions still depend on manual roll calls, which consume lecture time and are prone to errors, or on single-trait automation such as RFID cards and a lone fingerprint scanner. Cards can be handed to a friend, and a single biometric trait can be spoofed or can fail because of dirt, injury, or poor sensing. A common consequence is proxy attendance, where a present student marks attendance on behalf of an absent one.

A multimodal biometric system addresses this weakness by requiring more than one trait to agree on the same identity. In the proposed work, the person's face, captured live from a camera, and the person's fingerprint are both used. Attendance is marked only when both modalities support the same identity with sufficient confidence. Convolutional Neural Networks are used for recognition because they learn hierarchical visual features directly from images and have become

a standard tool for face and fingerprint analysis [3], [4].

Beyond building a recognition model, this work treats the attendance system as a complete product. The trained models are embedded in a Flask web application that watches a live camera feed without any button press, records the result in an SQLite database, stores a proof photograph, and exposes the data through an administrator dashboard. The main contributions are: (i) a dual-branch CNN that fuses face and fingerprint features instead of running two unconnected systems; (ii) an OpenCV-based augmentation pipeline designed to counter the overfitting observed during testing; (iii) a face-first decision flow with confidence and margin checks that rejects uncertain predictions; and (iv) a working end-to-end application with analytics, built so that a physical fingerprint scanner can be plugged in later.

II. Related Work

Relevant conflicts of interest/financial disclosures: The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Biometric recognition has been studied extensively. Jain et al. gave a broad introduction to biometric recognition, describing the operating modes of identification systems and the limitations of single-trait systems such as noise in the sensed data, non-universality, and spoofing [2]. Ross and Jain examined information fusion in biometrics and described fusion at the sensor, feature, match-score, and decision levels, noting that combining evidence from multiple traits generally improves accuracy and makes spoofing harder [1]. Feature-level fusion preserves the richest information because the raw feature sets of each trait are combined before any decision is taken, which motivates the architecture used in this paper.

Deep learning has strongly influenced face recognition. LeCun et al. reviewed how deep convolutional networks learn multi-level representations from raw data [3]. DeepFace and FaceNet showed that CNNs trained on large face collections can reach very high verification accuracy under unconstrained conditions [5], [6]. For fingerprints, CNN-based methods have been applied to tasks such as liveness detection, where learned features outperformed hand-crafted descriptors [7]. Public datasets make such work reproducible;

the Sokoto Coventry Fingerprint Dataset (SOCOFing) offers more than 6,000 fingerprint images with labels and synthetically altered variants, and was used here as the fingerprint source [8].

Training small custom datasets introduces the risk of overfitting. Batch normalization stabilizes and accelerates the training of deep networks [9], and data augmentation, which creates label-preserving variants such as rotated, flipped, and brightness-shifted images, is a widely used way to improve generalization [10]. However, many reported systems stop at a trained model evaluated offline. The present work differs by combining the two modalities in one fused network and by delivering a complete, live attendance application with decision logic designed from observed failure cases.

Table I compares the main attendance approaches in qualitative terms. Manual and card-based methods are cheap but easy to cheat, single-trait biometrics are harder to cheat but sensitive to sensor and environmental problems, and a multimodal system is the most robust at the cost of more training data and design effort.

Table I: Comparison of Attendance Methods

Method	Speed	Proxy risk	Weak point
Manual roll call	Slow	High	Time loss, errors
RFID card	Fast	High	Card can be lent
Single fingerprint	Fast	Low	Sensor wear, dirt
Face only	Fast	Low	Lighting, pose
Proposed multimodal	Fast	Very low	Needs good data

III. Proposed Methodology

The system is organized into sequential stages: environment setup, dataset collection, preprocessing and augmentation, training of the face CNN, the fingerprint CNN and the fusion model, face-first decision logic, web deployment, and administration. The implementation uses Python with TensorFlow/Keras for the models, OpenCV for camera access and face detection, NumPy and scikit-learn for data handling, Flask for the server, and SQLite for storage [11], [12] (Table II).

Table II: Tools and Technologies Used

Category	Tool and purpose
Language	Python, main implementation language

Category	Tool and purpose
Deep learning	TensorFlow / Keras, building and training CNNs
Image processing	OpenCV, webcam access and face detection
Data handling	NumPy, scikit-learn, array processing and train/test split
Charts	Matplotlib, Chart.js, accuracy graphs and attendance trends
Backend	Flask, web server and decision API
Database	SQLite, name, date, time and proof records
Frontend	HTML, CSS, JavaScript, camera page and dashboard
Face data	Own webcam photographs of team members
Fingerprint data	SOCOFing public dataset [8]

The system follows a three-layer design (Fig. 1). The presentation layer runs in the browser and contains the live camera page and the administrator dashboard. The application layer is a Flask server holding the REST routes, face detection, the CNN and fusion models, and the decision logic. The data layer contains the SQLite database, the stored proof photographs, and the datasets and saved models. The browser communicates with the server only through REST API calls.

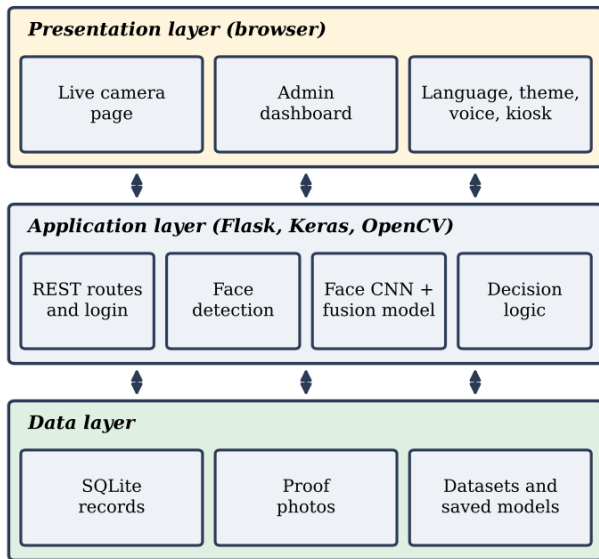


Figure 1: Layered Architecture of the Proposed System

A. Face Dataset Collection

A Python script (capture_faces.py) uses the webcam to capture many photographs of each enrolled team member automatically. Images of every person are stored in a separate folder named after that person. Around 100 to 250 photographs were collected per person, with slight changes in angle and expression so that the model learns the face rather than a single pose.

B. Fingerprint Dataset Collection

No physical fingerprint scanner was available, so fingerprint images were drawn from SOCOFing [8]. A fixed set of sample fingerprints was assigned to each enrolled member to act as that member's stored fingerprint template. This substitution lets the full multimodal pipeline be developed and tested, while the architecture remains ready for a real scanner.

C. Preprocessing and Data Augmentation

Face and fingerprint images are converted to grayscale and resized to 100×100 pixels, because the

CNNs require inputs of identical size and format. To counter overfitting, a custom OpenCV augmentation pipeline generates additional training images by small rotations, flips, and brightness changes [10]. The augmented set exposes the model to the variation present in live camera images, which a small captured dataset cannot supply by itself.

D. Dual-Branch CNN and Fusion Architecture

Two CNN branches were built, one for faces and one for fingerprints. Each branch stacks Conv2D, Batch Normalization [9] and Max Pooling layers that progressively extract more abstract visual features and produce a compact feature vector. The face and fingerprint CNNs were first trained separately to confirm that each works on its own.

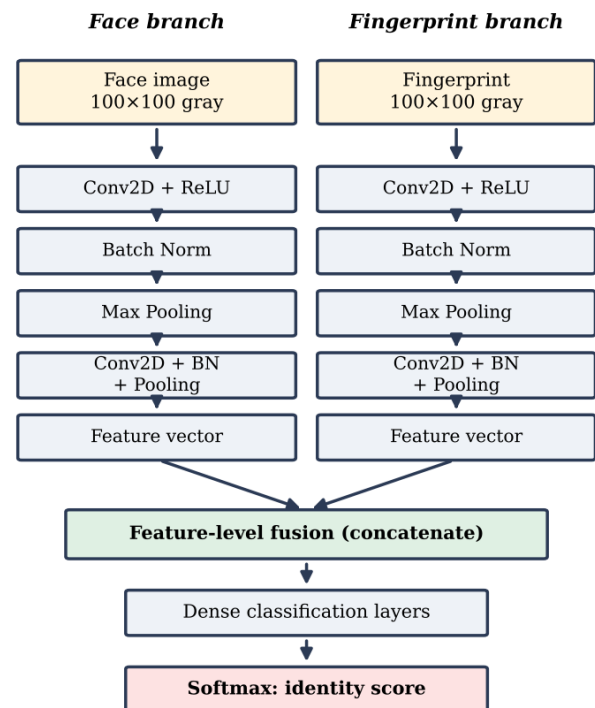


Figure 2: Dual-branch CNN with Feature-Level Fusion for the Multimodal Attendance System

A third model, the fusion model, is the multimodal core of the system. It takes the feature vectors learned by both branches, concatenates them (feature-level fusion [1]), and passes the combined representation through classification layers that output one final identity score. Because the decision is made on the joint representation, an impostor must be consistent in both modalities to be accepted.

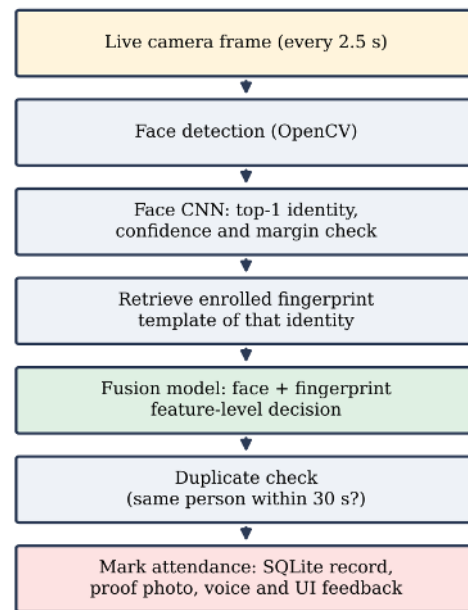
E. Face-First Decision Logic

An early fusion version relied too heavily on the fingerprint sample and repeatedly guessed the same person regardless of the live face. The logic was therefore redesigned so that the live face is the primary real-time signal. The face CNN first proposes an identity; only then is that identity's stored fingerprint template retrieved and fused with the live face features for confirmation.

The acceptance rule does not rely only on the top confidence score. It also examines the margin between the best and second-best class probabilities. A large margin indicates that the model is truly sure, whereas a small margin indicates confusion between identities. Predictions with insufficient confidence or margin are rejected instead of being forced into a possibly wrong match.

F. Web Application and Attendance Marking

The Flask application opens the camera in the browser and runs continuously without any button. Every 2.5 seconds a frame is captured and checked for a face using OpenCV. If a face is found, the decision logic of Section III-E is applied. When both modalities agree with high confidence, the name, date, and time are written to the SQLite database, a proof photograph is saved as evidence, a success message is shown, the name is announced by voice, and a short confetti animation is displayed. If the same person was already marked within the previous 30 seconds, the system reports "already marked" and does not create a duplicate record. The complete flow is shown in Fig. 3.



Uncertain or rejected cases: no attendance is marked

Figure 3: Real-time Attendance Workflow of the Proposed System

G. Administrator Panel and Interface

A separate password-protected administrator panel lets a teacher monitor and correct the data. It shows the total number of enrolled students, who is present today, day-wise summaries, and a 30-day attendance trend chart. Administrators can export attendance as a CSV file, search and filter records by name or date, edit or delete incorrect entries, view proof photographs, and add manual entries. A warning is raised when a student's attendance percentage falls below 75%.

The frontend was redesigned several times for usability. It offers a live clock, Hindi and English language switching, dark and light modes, a full-screen kiosk mode for attendance desks, a scanning animation, and a bounding box drawn around the detected face. A consistent gold-and-emerald theme is used on both the camera page and the dashboard.

H. Implementation Details

The development environment was created in Anaconda as a separate environment (biometric_env) so that the project libraries do not conflict with other software, and Visual Studio Code was used as the editor. Early errors caused by Python version conflicts, missing libraries such as dlib, and a wrong

virtual environment were resolved by rebuilding the environment with a compatible Python version. The browser captures a frame every 2.5 seconds and sends it to the Flask server, which returns either a recognized identity or a rejection. The interface offers a live clock, Hindi and English switching, dark and light themes, a kiosk mode for attendance desks, a scanning animation and a box around the detected face. On a successful match the name is announced by voice and a short confetti animation is shown.

I. Security and Privacy Considerations

Biometric data is sensitive, so the design limits what is exposed. The administrator panel is password protected, while the student-facing camera page shows only the result of the current recognition and no stored records. Face photographs and fingerprint templates are kept on the server and are used only for recognition and as attendance evidence. Duplicate marking is blocked for 30 seconds to prevent repeated entries, and every attendance record has a proof photograph so that disputed entries can be reviewed. For a campus deployment, stored templates should additionally be encrypted, and students should be informed about how their biometric data is stored and used.

IV. Experimental Results and Discussion

The system was developed and tested in an Anaconda environment (biometric_env) using Visual Studio Code, with Python, TensorFlow/Keras, OpenCV, NumPy, scikit-learn, Matplotlib, and Flask. Face data comprised 100 to 250 self-captured images per team

member, and fingerprint data came from SOCOFing. Table III summarizes the main configuration and outcome.

A. Experimental Setup and Overall Performance

Table III: System Configuration and Outcome

Parameter	Value
Input modalities	Face (live camera) + fingerprint
Input size	100 × 100, grayscale
Face images per person	About 100–250
Fingerprint source	SOCOFing (public dataset)
Fusion type	Feature-level (concatenation)
Frame capture interval	2.5 seconds
Duplicate-marking window	30 seconds
Low-attendance alert	Below 75%
Best fusion validation accuracy	About 99%

The best fusion model reached about 99% accuracy on validation data, meaning that it identified the correct enrolled person in almost every validation sample. The individual face and fingerprint CNNs were also trained and verified separately before fusion. In live operation, the system marks attendance automatically when a face is recognized, records evidence, and prevents duplicate entries.

B. Problems Encountered and Solutions

Several real problems were found during testing, and the resulting fixes are an important part of the contribution (Table IV).

Table IV: Problems Faced and Solutions

Problem	Cause	Solution
Environment errors	Python version conflicts, missing libraries such as dlib, wrong virtual environment	Rebuilt the environment with a compatible Python and reinstalled all libraries
Overfitting	Face model confident on training photos but as low as 20% accurate on new live images	Custom augmentation pipeline and more photographs per person
Wrong person announced	Some folders were empty because images were saved in the wrong project folder	Fixed folder structure and re-captured complete sets for everyone
Fusion bias	Model relied on fingerprint sample and always guessed one person	Face-first identification, then fingerprint confirmation
False matches	Top-1 confidence alone accepted confused predictions	Added margin check between best and second-best scores

The overfitting case is notable. The face model scored very well on training photographs yet reached as little as 20% accuracy on new camera images, which shows

that offline accuracy alone is a poor indicator of live behavior. Augmentation and a larger set of captured images corrected this. The misidentification case was

traced not to the model design but to inconsistent training data, which confirms that data hygiene is as important as architecture.

C. Live System Behavior

In live operation the camera page stays active without any button press. Each captured frame is checked for a face, and frames without a face are ignored. A recognized person receives immediate visual, spoken and animated confirmation, while an unrecognized or uncertain face receives no entry. If the same person remains in front of the camera, the 30-second window prevents duplicate records and the system reports that attendance is already marked. All entries appear in the administrator panel as soon as they are written, where the teacher can search by name or date, export a CSV file, correct a wrong entry, add a manual entry, or open the proof photograph. Students whose attendance falls below 75% are flagged automatically, which saves the teacher the effort of calculating percentages at the end of a term.

A practical observation from testing is that results on stored images and results on live images can differ widely. The face model that looked strong on training photographs fell as low as 20% on new camera images, and this gap was found only because the system was tested live. This supports testing biometric systems under real conditions of lighting, pose, and distance rather than relying on offline accuracy alone.

D. Discussion and Limitations

The results show that feature-level fusion of two CNN branches, controlled by a face-first rule and a margin-based rejection test, gives reliable identification on the enrolled group. Rejecting uncertain predictions reduces wrongly announced names at the cost of occasionally asking the person to look at the camera again, which is the preferable failure for attendance.

Several limitations should be stated. First, the group of enrolled users is small, so the reported validation accuracy should not be read as population-level performance. Second, fingerprints are taken from a public dataset and assigned to team members, so the fingerprint modality does not yet reflect true per-person biometric capture. Third, the roughly 99% figure is validation accuracy, and a fully separate unseen test set with precision, recall, F1-score, and a confusion matrix should be reported in future evaluation. Fourth, the system has no dedicated liveness detection, so a

photograph shown to the camera is a possible spoofing risk.

CONCLUSION AND FUTURE SCOPE

This paper presented a Multimodal Biometric Attendance System that fuses face and fingerprint features using a dual-branch CNN and delivers the result as a complete web application. The work covered environment setup, dataset collection, preprocessing and augmentation, training of face, fingerprint and fusion models, a face-first decision flow with confidence and margin checks, an SQLite-backed live attendance interface, and an administrator dashboard with export, search, editing, analytics, and low-attendance alerts. The best fusion model achieved about 99% validation accuracy, and the problems encountered during development, particularly overfitting and misidentification, were solved through augmentation, data correction, and improved decision logic.

Future work includes integrating a real fingerprint scanner using the existing template-based design, enrolling a larger and more diverse set of users, and evaluating on a separate unseen test set with full class-wise metrics. Liveness detection and anti-spoofing for the face stream, such as blink or depth cues, would strengthen security. Further directions include score-level and decision-level fusion comparisons, lightweight models for low-cost edge devices, cloud or multi-classroom deployment, and automatic integration with institutional student information systems.

REFERENCES

1. A. Ross and A. K. Jain, "Information fusion in biometrics," *Pattern Recognition Letters*, vol. 24, no. 13, pp. 2115–2125, 2003.
2. A. K. Jain, A. Ross, and S. Prabhakar, "An introduction to biometric recognition," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 14, no. 1, pp. 4–20, 2004.
3. Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.
4. A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," *Communications of the ACM*, vol. 60, no. 6, pp. 84–90, 2017.

5. Y. Taigman, M. Yang, M. Ranzato, and L. Wolf, "DeepFace: Closing the gap to human-level performance in face verification," in *Proc. IEEE CVPR*, 2014, pp. 1701–1708.
6. F. Schroff, D. Kalenichenko, and J. Philbin, "FaceNet: A unified embedding for face recognition and clustering," in *Proc. IEEE CVPR*, 2015, pp. 815–823.
7. D. Menotti et al., "Deep representations for iris, face, and fingerprint spoofing detection," *IEEE Transactions on Information Forensics and Security*, vol. 10, no. 4, pp. 864–879, 2015.
8. Y. I. Shehu, A. Ruiz-Garcia, V. Palade, and A. James, "Sokoto Coventry Fingerprint Dataset," *arXiv:1807.10609*, 2018.
9. S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proc. ICML*, 2015, pp. 448–456.
10. C. Shorten and T. M. Khoshgoftaar, "A survey on image data augmentation for deep learning," *Journal of Big Data*, vol. 6, art. 60, 2019.
11. M. Abadi et al., "TensorFlow: A system for large-scale machine learning," in *Proc. USENIX OSDI*, 2016, pp. 265–283.
12. G. Bradski, "The OpenCV library," *Dr. Dobb's Journal of Software Tools*, 2000.

HOW TO CITE: Priyanshu, Dhawal, Shweta, Swarup, Abhishek, Vivek, Rahul Kawariya, Multimodal Biometric Attendance System Using Convolutional Neural Networks with Face and Fingerprint Fusion, *Int. J. Sci. R. Tech.*, 2026, 3 (10), 1397-1403. <https://doi.org/10.5281/zenodo.23761291>