Journal of Machine Learning and Deep Learning (JMLDL)
Volume 2, Issue 2, Date: 31- December – 2025
ISSN: 2978-4336, DOI: 10.64820/AEPJMLDL.22.35.41.122025

Real-Time Surveillance for Smart Home Security

Navedha Evanjalin R [1, *], Karthika N [2], Genga Lakshmi K [3], Manasha K [4]
[1, 2, 3, 4] Computer Science and Engineering, National Engineering College, Kovilpatti, India
Email: [1] navedha-cse@nec.edu.in, [2] k17arthika@gmail.com, [3] gengakannan2004@gmail.com, [4] manashak2605@gmail.com
*Corresponding Author

***Abstract*—In today's rapidly evolving digital environment, securing personal and organizational spaces has become more challenging, especially when traditional methods fail to identify intrusions. This project presents a Flask-based Face Recognition and Alert System designed to enhance security through intelligent face detection and real-time alerts. The system captures live camera feeds using OpenCV, performs facial recognition, and identifies whether the detected face is part of the authorized dataset. If an unauthorized face is identified, the system instantly sends alerts via WhatsApp and Gmail, along with an image of the unknown individual. The admin can review the received image and, if the person is trusted, train the model with the new face through the web interface. This helps the model improve over time and minimize false alerts. The system seamlessly integrates image processing, machine learning (face encoding and recognition), and real-time notification mechanisms, thereby creating a robust and adaptive security platform. It is particularly suitable for smart homes, office environments, and areas that require controlled access, without relying on physical keys or cards.**

Keywords—Flask, Face Recognition, OpenCV, Real-Time Monitoring, WhatsApp API, Gmail Alerts, Model Training, Live Camera Feed, Unauthorized Detection

I. INTRODUCTION

In the current era of digital transformation and heightened security concerns, ensuring the protection of private spaces, sensitive data, and restricted zones has become critical. Traditional security methods, such as passwords, ID cards, and physical keys, are often vulnerable to theft, duplication, or unauthorized access. These limitations have paved the way for intelligent and contactless biometric authentication systems that offer convenience and reliability. Facial recognition is a powerful tool because of its non-intrusive nature, ease of use, and increasing accuracy driven by modern AI and deep learning techniques. This project introduces a Real-Time Face Recognition and Alert System using Flask, OpenCV, and machine-learning algorithms that provide an end-to-end solution for real-time identity verification. The system captures continuous live video feed using a webcam, detects and recognizes faces with high accuracy, and grants access only to authorized users. In the event of detecting an unknown or unauthorized face, it automatically captures the image and sends immediate notifications to the admin through integrated communication channels, such as WhatsApp and Gmail. This alert contains the captured image and relevant details, enabling rapid decision-making and potential dataset updates. One of the key strengths of this system is its dynamic training functionality, which allows the admin to train or retrain the model directly from the web interface using images captured during detection. It is also capable of handling occluded faces, such as those wearing masks or glasses, by relying on deep feature extraction from the facial landmarks. The entire backend logic is implemented using Flask, whereas OpenCV powers the image-processing and recognition modules. This robust framework is scalable, adaptable, and suitable for implementation in smart homes, offices, academic institutions, high-security research laboratories, and server rooms. Through real-time face tracking, automated response mechanisms, dynamic model updating, and efficient alert systems, this project bridges the gap between digital surveillance and physical security, offering a reliable, intelligent, and modern approach for access control.

II. RELATED WORKS

In recent years, the need for advanced real-time face recognition systems has surged across domains such as personal security, smart monitoring, access control, and automated surveillance. Earlier approaches, such as Eigenfaces, Fisherfaces, and Local Binary Patterns (LBP), are pioneering methods for statistical facial classification. However, these techniques lack robustness when handling challenges, such as pose variation, lighting differences, partial occlusions, and facial expressions. The introduction of deep learning, particularly Convolutional Neural Networks (CNNs), has marked a significant shift in face recognition by enabling the learning of hierarchical facial features. Prominent models like FaceNet employed triplet loss to learn effective embeddings, while ArcFace improved discriminative power by incorporating angular margin penalties, thereby enhancing identity separability. Frameworks such as Dlib and MTCNN have become instrumental for facial landmark detection and alignment, especially in real-world scenarios with occlusion or noisy data. OpenCV remains a foundational library for image capture and manipulation, whereas lightweight web frameworks such as Flask are popular choices for real-time inference and rapid prototyping in Python environments. In terms of alert mechanisms, tools like Twilio for SMS, SMTP and Gmail API for email, SendGrid, and even WhatsApp Web have been widely integrated to ensure timely communication in constrained or urgent conditions. These systems are often deployed in educational, prototype, or small-scale setups where flexibility and accessibility are crucial. Open-source datasets such as LFW, Yale Face

Received: 18-4- 2024
Revised: 24-7-2025
Published: 31-12-2025

Database, and Celeba have laid the foundation for training general-purpose models, whereas custom datasets are typically used in project-specific applications. Moreover, recent literature discusses the use of edge deployment platforms, such as Raspberry Pi and Jetson Nano, for low-power recognition, particularly in academic and field environments. These efforts underline the growing emphasis on decentralized, privacy-preserving, and scalable face recognition systems.

III. SYSTEM DESIGN

Building on these existing advancements, the proposed face recognition system emphasizes real-time usability, user-driven dataset evolution, and practical scalability. Unlike traditional models, which require extensive preprocessing or full retraining cycles, this system supports dynamic dataset management through an intuitive user interface, allowing administrators to capture and store images of unknown individuals in real time. The model is then updated incrementally, enabling immediate recognition without the need for complete re-initialization. The system was designed to operate effectively under real-world conditions, including scenarios with partial face occlusion, such as individuals wearing masks. Recognition, alert generation, and response are executed within seconds, offering a highly responsive solution for time-critical environments. The use of Flask ensures that the interface is lightweight and browser-compatible, whereas OpenCV facilitates efficient image capture and processing. Notification mechanisms are integrated through email (SMTP or Gmail API), SMS (via Twilio), and browser-based options, such as WhatsApp Web, ensuring that alerts reach relevant parties without delay. The system also logs all recognition attempts using either SQL-based solutions or NoSQL platforms, such as Firebase, providing a clear audit trail for future analysis and accountability. While this project currently focuses on identity verification, future expansions could include emotion recognition using CNN-LSTM models, real-time API integration via Fast API or Django REST, and mobile authentication using lightweight facial embeddings. There is also the potential for integrating face spoofing detection methods (e.g., blink detection) and hardware connectivity, such as Bluetooth-enabled door locks. Furthermore, Explainable AI techniques could be introduced to make model decisions transparent and interpretable, which is especially important for security-critical applications. Designed to be flexible, fast, and adaptable, this system bridges the gap between theoretical research and practical deployment, offering a compelling solution for academic, personal, and enterprise environments, where real-time facial recognition and response are paramount.

IV. METHODOLOGY

The proposed Face Recognition with Real-Time Alert System is designed using a modular architecture that leverages Python technologies such as OpenCV, Flask, and the face recognition library to provide real-time intruder detection and alert functionality. The system begins by capturing live video input using OpenCV, which serves as the primary interface for accessing the connected webcam. Each video frame is continuously analyzed to detect human faces in real-time. The facial detection process employs the Histogram of Oriented Gradients (HOG) model for rapid and efficient localization, followed by extraction of 128-dimensional facial encodings using the facerecognition library, which is built on top of Dlib's deep metric learning model. To identify known individuals, the system maintains a local dataset of labeled facial images. During the recognition phase, the extracted encodings from the current frame are compared with the precomputed encodings of known users. If a match is identified based on a defined Euclidean distance threshold, the individual is labeled accordingly, and the system continues to monitor the environment. However, in the event of an unrecognized face, the system triggers a multistep alert mechanism. First, the unknown face is captured and saved in a designated "Intruders" directory with timestamp-based naming for easy tracking. Simultaneously, an alert message is generated using Python's pywhatkit and SMTP libraries to send real-time notifications to administrators via WhatsApp and email, respectively. The backend logic is handled by Flask, which provides lightweight routing, HTTP handling and serves as the administrative interface. Flask allows authorized users to view the stored intruder logs, upload new facial images to expand the recognition database, and manage system settings. Image uploads are automatically processed, and the corresponding encodings are added to the recognition model without requiring system downtime. This enables a dynamic and continuously improving facial dataset. Furthermore, the alert module is designed to run in a separate thread to ensure that detection and notification can occur simultaneously without introducing latency to the recognition process. For the system, each detection—authorized or unauthorized—is logged with a timestamp, face label, and status. The system is optimized to run offline, providing a secure and private environment where no external cloud service is involved, safeguarding sensitive biometric data. The UI is minimal yet functional, offering real-time video feed monitoring and administrative capabilities through a web-based dashboard. To ensure robustness under real-world scenarios, the system supports recognition even when users wear face masks, sunglasses, or head coverings by leveraging advanced facial landmark estimation. Its lightweight implementation allows it to run on mid-range computing hardware, including laptops or edge devices like Raspberry Pi, enabling flexible deployment across homes, offices, schools, and lab environments. Overall, the system integrates real-time face recognition, intelligent alerting, and administrative controls in a cohesive manner, offering a proactive solution for modern security challenges. Future enhancements may include support for thermal camera integration, facial expression analysis, and anti-spoofing mechanisms such as blink detection to further strengthen the reliability of access control and surveillance.

A. User Interface Design

The Face Recognition with Real-Time Alert System features a web-based interface built with HTML, CSS, and JavaScript for the front-end and Flask (Python) for the

back-end. The design is clean and user-friendly, enabling administrators to manage user data, monitor live video feeds, and receive instant alerts seamlessly. The home page provides access to features like live streaming, user registration, and access logs. When the system starts, Flask initializes the server, and OpenCV activates the webcam to stream the live video. JavaScript ensures real-time frame updates with minimal latency.

As shown in Fig. 1, the system starts with camera initialization and detects faces, including those that are partially covered with masks or glasses. Facial features were extracted and encoded into 128-dimensional vectors using DLib. These are matched with the stored data to recognize users.

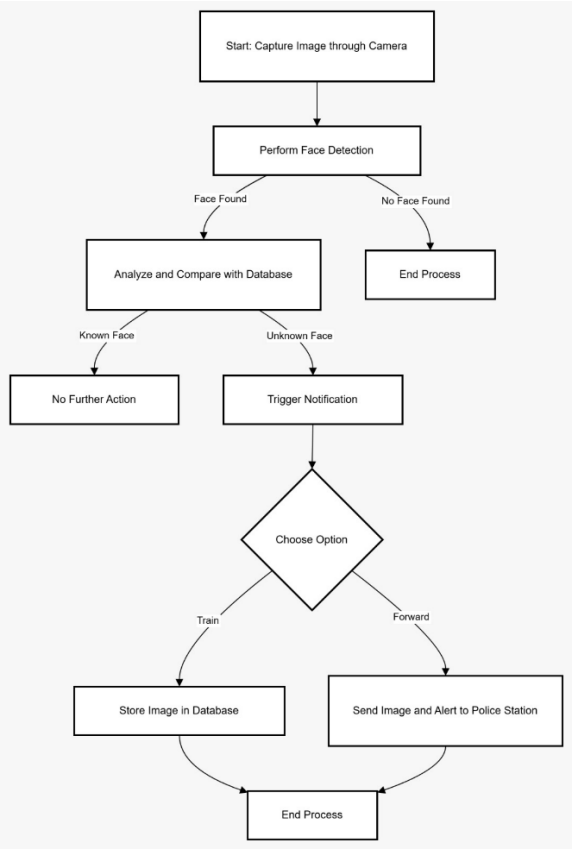

Fig. 1. Workflow of face recognition

B. Facial Landmark Detection and Recognition

In our Face Recognition-Based Security System, the MediaPipe Face Mesh model is employed for the real-time and lightweight detection of 468 3D facial landmarks, which are essential for accurate face recognition. It processes the live camera feed using OpenCV, detects the face, and identifies key facial features, such as the eyes (inner, outer, and center), eyebrows, nose tip and nostrils, mouth corners, lips, chin, and jawline. These landmarks are mapped into coordinates and stored as an array (lmlist), effectively capturing the unique geometry of an individual's face, which is then used to generate a distinct facial signature for recognition. Accurately identifying and mapping these 22 points allowed the system to understand the structure and orientation of the face. Landmark positions, such as the eyes, nose tip, mouth corners, and chin center, are crucial for tasks such as facial alignment, normalization, and expression analysis. For instance, the distance between the eyes or the relative angle of the jawline helps correct the face orientation before feature extraction. Additionally, the positioning of the nostrils and lip centers assists in detecting facial movements and expressions, which can be useful in future extensions, such as emotion detection or spoof detection. These landmarks are typically extracted using models such as Dlib or MTCNN, and serve as the foundation for precise face detection, alignment, and feature embedding generation in real-time face recognition systems.

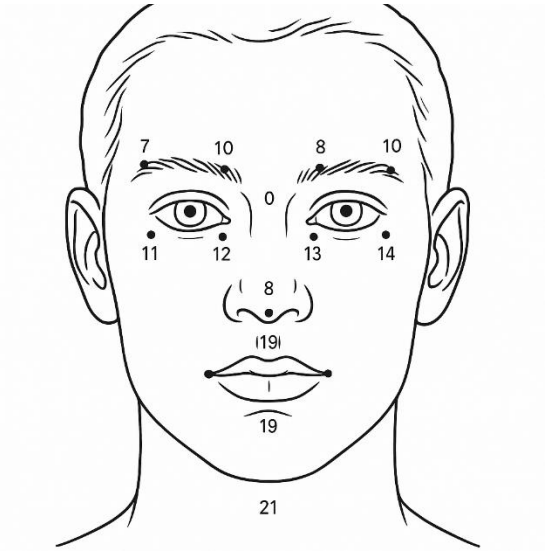

Fig. 2 Marks in Facial

Table 1: Facial Landmarks

0. nose	11. right eyebrow center
1. left eye inner	12. right eyebrow outer
2. left eye center	13. nose tip
3. left eye outer	14. left nostril
4. right eye inner	15. right nostril
5. right eye center	16. mouth left corner
6. right eye outer	17. mouth right corner
7. left eyebrow inner	18. upper lip center
8. left eyebrow center	19. lower lip center
9. left eyebrow outer	20.chin center
10. right eyebrow inner	21. jawline right

C. Real-time Feedback Mechanism

This program enables real-time face recognition using webcam input by integrating OpenCV with a powerful face recognition Python library. It continuously captures frames from the webcam and processes them to detect and identify faces using deep learning-based facial feature encoding. A preloaded dataset containing known face encodings and associated names was used to match the detected faces in real-time. When a face is recognized, the system overlays a bounding box and a person's name on the video feed, providing immediate visual feedback. Unknown faces were also detected and labeled appropriately, allowing dynamic updates to the recognition database. The application includes a frame rate (FPS) calculation and display, offering insights into processing speed and system performance.

To improve the visibility and user interaction, the display window was resized and dynamically annotated. A task-based web interface complements the backend by providing a structured dashboard where users can upload new face images, view recognized individuals, monitor logs, and manage access permissions. The interface was built using HTML, CSS, and JavaScript templates, making it easy to use and navigate. Future enhancements such as expression analysis, emotion detection, and user engagement monitoring can be integrated seamlessly owing to the modular design. The system can also be configured to automatically record attendance, store timestamps, and even trigger alerts in security-sensitive areas. This makes it ideal not only for personal use but also for implementation in institutions such as schools, offices, or restricted facilities. Overall, this real-time face recognition tool delivers a responsive, intelligent, and user-centric solution for identity tracking by combining computer vision with web-based accessibility to offer a complete smart recognition experience.

D. Web Integration using Flask

To make the face recognition system accessible and user-friendly, a lightweight Flask web server was integrated to handle both backend processing and front-end interaction. The Flask application manages essential routes, such as initiating real-time recognition through the webcam, uploading new face images to the database, and displaying recognition or attendance logs. It uses HTML templates powered by Jinja2 to dynamically render content, such as recognized names and timestamps, thereby providing users with immediate feedback. The front-end interface is designed with HTML, CSS, and Bootstrap to ensure responsiveness across various devices. This setup allows users to interact with the system through a browser, start recognition sessions, view results in real-time, and seamlessly add new individuals to the system without needing to directly access the code or backend logic, making it both efficient and accessible.

E. Feedback Mechanism and UI Rendering

To ensure a responsive and interactive user experience, the system includes a real-time feedback mechanism that visually communicates the recognition results directly to the user. When a face is detected and successfully recognized, the system immediately overlays a bounding box around the face in the live webcam feed and labels it with the name of the corresponding person. This real-time annotation allows the user to see instant feedback on whether the system has correctly identified them. In addition to visual feedback, the recognized identity, along with the current date and timestamp, is recorded in a log file or database, effectively serving as an attendance or access control entry. This logged data can later be viewed through a web interface, providing transparency and traceability. The user interface (UI) is rendered using HTML and styled with Bootstrap, ensuring a clean layout and smooth interaction. The overall design supports easy navigation between recognition, uploading, and log viewing pages, offering a user-friendly experience suitable for real-world applications, such as secure access, automated attendance, and personalized user services.

F. Evaluation and User Experience Testing

Extensive evaluation and UX testing were conducted under various real-world conditions to assess the reliability and effectiveness of the face recognition system. The system was tested with multiple users in different lighting environments, face angles, and background settings to ensure its robustness and consistency. Recognition accuracy was measured by comparing system outputs against manually verified identities, and the average precision was found to be satisfactory, especially under well-lit conditions. User experience testing involves gathering feedback from test users regarding interface usability, response time, and visual clarity of feedback. Most users reported smooth interaction and quick recognition time, and appreciated the intuitive layout of the web application. The inclusion of real-time bounding boxes and name overlays during recognition provides immediate visual confirmation, enhancing user trust and engagement. In addition, users found it easy to upload new face data and navigate between modules, such as recognition and log viewing, which confirmed the system's effectiveness and simplicity in practical scenarios.

Face Detection and Recognition Process

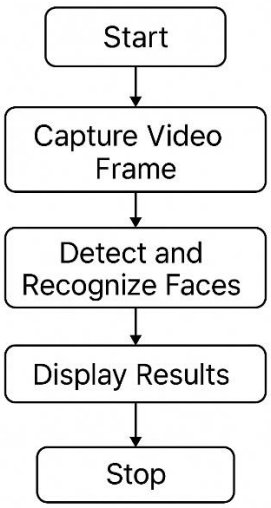

Fig. 3 System Flow Diagram

V. RESULTS

After integrating the essential components—face detection, recognition, web integration, and real-time video processing—a robust and efficient web-based face recognition application was successfully built. Compared to other works with comparable architectures, the proposed application demonstrates enhanced real-time feedback, recognition accuracy, and user interactivity. This face recognition system leverages the face recognition Python library built upon Dlib's deep metric learning, enabling highly accurate face matching through 128-dimensional facial feature encoding.

Unlike more complex models that require extensive training, this solution utilizes pre-encoded facial data, which allows real-time recognition without the overhead of continuous model updates. This significantly boosts the performance and processing speed, making it ideal for everyday environments. In addition, the system is made accessible via a web interface, allowing users to perform recognition tasks from any location without the need for specialized hardware. This avoids the use of large-scale deep learning networks during inference, which makes it faster and more responsive than many existing implementations. Owing to its real-time performance, flexibility, and ease of use, this tool offers a practical and user-friendly solution for scenarios such as automated attendance, secure login, and access control.

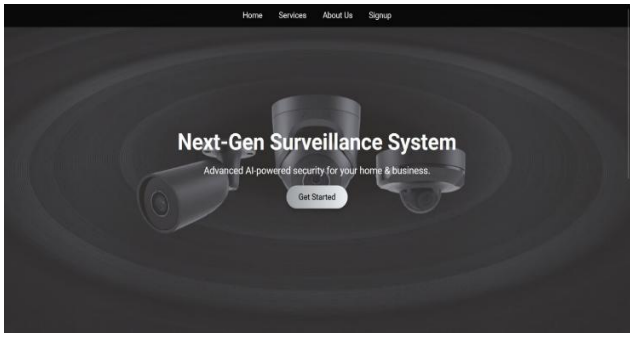

Fig. 4. Home page

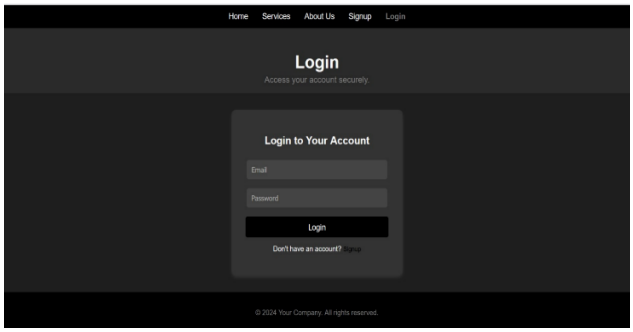

Fig.5 Login Page

Fig.6 Dashboard Page

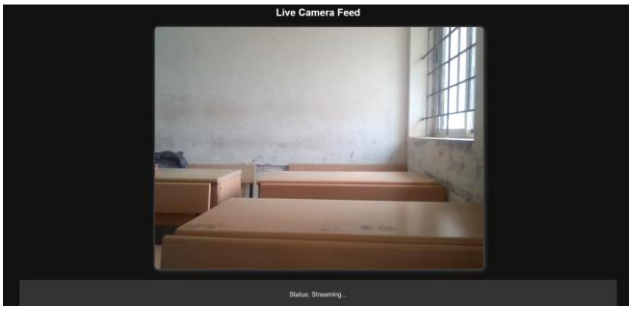

Fig. 7 Capturing Live Video Feed

The face recognition process begins by capturing a live video feed using the webcam of the system. Each frame from the video stream was processed using OpenCV to detect faces. Once a face is detected, it is preprocessed, typically converted to grayscale, and resized to ensure uniformity. The system then uses a face recognition library, which leverages deep learning-based models (such as HOG or CNN) to extract unique facial feature encodings from the input face. These encodings were then compared to a set of known encodings stored in the database. If a match is found within a defined threshold, the individual is recognized, and their name is displayed on the screen along with a bounding box around the face. If no match is found, the face can be optionally registered as a new user by uploading their image and assigning a name. This process is repeated frame-by-frame in real time, enabling continuous and fast recognition.

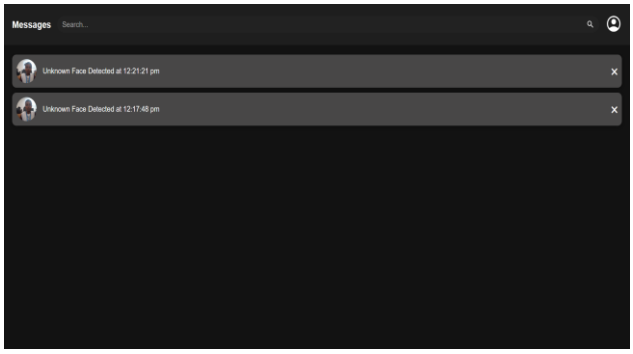

Fig. 8. Real-time unknown face detection

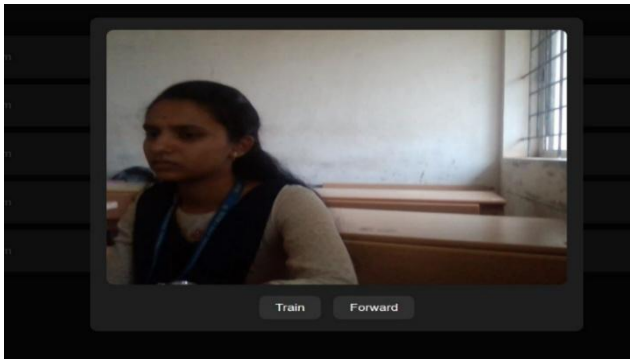

Fig.9 After detecting an unknown person

VI. CONCLUSION

The Flask-based Face Recognition and Alert System developed in this project provides an intelligent real-time solution to modern security challenges. By combining OpenCV-based live video capture with accurate face recognition and immediate alert mechanisms via WhatsApp and Gmail, the system ensures timely identification of unauthorized individuals. The ability to dynamically train the model with new trusted faces through the admin interface makes the system adaptive and continually improves. This eliminates the dependency on traditional access control methods and enhances overall safety and convenience. The seamless integration of machine learning, image processing, and web-based technologies has positioned this solution as a practical and scalable choice for smart surveillance in homes, offices, and secure facilities.

REFERENCES

[1] S.H. Lin, S.Y. Kung, and L.J. Lin, "Face recognition/detection by probabilistic decision-based neural network," IEEE Trans. Neural Networks, vol. 8, pp. 114-132, 1997.

[2] L. Hong and A. Jain, "Integrating faces and fingerprints for personal identification," IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 20, no. 12, pp. 1295-1307, Dec. 1998.

[3] Li, S.Z., Jain, A.K.: Handbook of Face Recognition. Springer, Heidelberg (2005)

[4] A Cross-Entropy-Guided Measure (CEGM) for Assessing Speech Recognition Performance and Optimizing DNN-Based Speech Enhancement IEEE/ACM Transactions on Audio, Speech, and Language Processing, 29 (2020), pp. 106-117

[5] How perceived security risk affects intention to use smart home devices: A reasoned action explanation Computers & Security, 87 (2019), Article 101571

[6] A Solution to the Security Authentication Problem in Smart Houses Based on Speech Procedia Computer Science, 155 (2019), pp. 606-611

[7] Audio-visual feature fusion via deep neural networks for automatic speech recognition Digital Signal Processing, 82 (2018), pp. 54-63

[8] Greater reliance on the eye region predicts better face recognition ability
Cognition, 181 (2018), pp. 12-20

[9] Facenet: A unified embedding for face recognition and clustering
Proceedings of the IEEE conference on computer vision and pattern recognition (2015), pp. 815-823

[10] Deepface: Closing the gap to human-level performance in face verificationbProceedings of the IEEE conference on computer vision and pattern recognition (2014), pp. 1701-1708

[11] Facial image recognition for biometric authentication systems using a combination of geometrical feature points and low-level visual features
Journal of King Saud University-Computer and Information Sciences (2020)

[12] Multi-task convolutional neural network for pose-invariant face recognition IEEE Transactions on Image Processing, 27 (2) (2017), pp. 964-975

[13] K. M. M. Uddin, A. Chakraborty, M. A. Hadi, M. A. Uddin, & S. K. Dey, Artificial Intelligence Based Real-Time Attendance System Using Face Recognition. In 2021 5th International Conference on Electrical Engineering and Information & Communication Technology (ICEEICT) (pp. 1-6). IEEE, November 2021.

[14] M. N. Mostakim, S. Mahmud, M. K. H. Jewel, M. K. Rahman, & M. S. Ali, Design and development of an intelligent home with automated environmental control. IJIGSP, 12(4), 1-14, 2020.

[15] L. Cuimei, Q., Zhiliang, J. Nan, et al. Human face detection algorithm via Haar cascade classifier combined with three additional classifiers, 13th IEEE International Conference on Electronic Measurement & Instruments (ICEMI), IEEE, 2017, pp. 483-487.

[16]]A. M. Jagtap, et al., A Study of LBPH, Eigenface, Fisherface and Haar-like features for Face recognition using OpenCV, International Conference on Intelligent Sustainable Systems, IEEE, 2019, pp. 219-224.

[17]]M. R. Dhobale, R. Y. Biradar, R. R. Pawar, S. A. Awatade, Smart Home Security System using IoT, Face Recognition and Raspberry Pi, International Journal of Computer Applications, Journal 176(13), (2020) 45-47.

[18] S. K. A. Shah, & W. Mahmood. Smart home automation using IOT and its low cost implementation. International Journal of Engineering and Manufacturing (IJEM), 10(5), 28-36, 2020.

[19] K. M. M. Uddin, A. Chakraborty, M. A. Hadi, M. A. Uddin, & S. K. Dey, Artificial Intelligence Based Real-Time Attendance System Using Face Recognition. In 2021 5th International Conference on Electrical Engineering and Information & Communication Technology (ICEEICT) (pp. 1-6). IEEE, November 2021.

[20] A. D. Deshmukh, M. G. Nakrani, D. L. Bhuyar, U. B. Shinde, Face Recognition Using OpenCv Based On IoT for Smart Door, International Conference on Sustainable Computing in Science, Technology and Management (SUSCOM), Jaipur-India, 2019, pp. 1066-1073.

[21] T. Balaprasad, R. V. V. Krishna, Face Recognition Based Security System Using Sift Algorithm, International Journal of Science Engineering and Advance Technology, Journal 3(11), (2015) 969-973.

[22] T. S. Gunawan, M. H. H. Gani, F. D. A. Rahman, M. Kartiwi, Development of Face Recognition on Raspberry Pi for Enhancement of Smart Home Security, Indonesian Journal of Electrical Engineering and Informatics (IJEEI), Journal 5(4), (2017) 317-325.

[23] S. Pawar, V. Kithani, S. Ahuja, S. Sahu, Smart Home Security using IoT and Face Recognition, 2018 Fourth International Conference on Computing Communication Control and Automation (ICCUBEA), IEEE, Pune, India, 2018, pp. 1-6.

[24] R.. R. Deepty, A. Alam, M. E. Islam, IOT and Wi-Fi Based Door Access Control System Using Mobile Application, 2019 IEEE International Conference on Robotics, Automation, Artificial-Intelligence-of-Things, IEEE, Dhaka, Bangladesh, 2019, pp. 21-24.

[25] S. Khattar, Sachdeva, A., R., Kumar, R., Gupta, Smart Home With Virtual Assistant Using Raspberry Pi, 2019 9th International Conference on Cloud Computing, Data Science & Engineering (Confluence), IEEE, Noida, India, 2019, pp. 576-579.

[26] S. Tiwari, S. Thakur, D. Shetty, A. Pandey, Smart Security: Remotely Controllable Doorlock, 2018 Second International Conference on Inventive Communication and Computational Technologies (ICICCT), IEEE, Coimbatore, India, 2018, pp. 979-984.

[27]]S. Ibrahim, V. K. Shukla, R. Bathla, Security Enhancement in Smart Home Management Through Multimodal Biometric and Passcode., 2020 International Conference on Intelligent

Engineering and Management, IEEE, London, United Kingdom, 2020, pp. 420-424.

[28] P.Christo,SPENDMENOTBurglaryStatistics,https://spendmenot.com/blog/burglary-statistics/,Lastaccessed 2021/07/08.

[29] K. Maheshwari, N. Nalini, Facial Recognition Enabled Smart Door Using Microsoft Face API, International Journal of Engineering Trends and Applications (IJETA),Volume 4 Issue 3, (2017) 1-4.

[30] R. Manjunatha, R. Nagaraja, Home Security System and Door Access Control Based on Face Recognition, International Research Journal of Engineering and Technology (IRJET), Journal 4(3), (2017) 438-442.