AI-Based Missing Person Detection System

Project Mentor: Shubham Upadhyay
Department of Computer Science and Engineering,
Parul Institute of Engineering and Technology, Vadodara, Gujarat, India 391760
Dishant Patel
Team Lead , python developer and Security Specialist
Anas solanki
Backend Developer and Database Administrator
Jaimin magajwala
Research , Development and logic building
Harsh Sutariya
Frontend Designer and Documentation Lead

***Abstract*—Abstract— The rise in urbanization and population growth has resulted in an alarming increase in missing person cases worldwide. Conventional methods of searching, which rely on manual reporting and field investigation, often lead to delayed responses and low success rates. Artificial Intelligence (AI), particularly computer vision and deep learning, provides powerful tools for automating detection, identification, and tracking of missing individuals. This paper proposes an AI-based missing person detection system that leverages image recognition, face matching, and surveillance integration to assist authorities and families in locating missing persons efficiently. The system utilizes Convolutional Neural Networks (CNNs), transfer learning models such as VGGFace and FaceNet, and integrates with existing CCTV networks to provide real-time alerts. Our implementation is built using Python with Flask as the backend framework and a web interface designed with HTML, CSS, and JavaScript. The system supports real-time notifications, database integration, and user-friendly access through a web app, ensuring practicality and scalability. In addition, the system is designed to allow flexible data management where authorities can continuously update records, thereby improving accuracy over time. It ensures compatibility with multiple camera sources, making it adaptable for both small-scale and citywide deployments. The integration of cloud services enhances scalability and allows storage of large datasets without compromising speed. Moreover, the system incorporates preprocessing techniques to handle varying image qualities, ensuring robustness in diverse environments. Ethical considerations, including data privacy and responsible AI practices, are also discussed to guarantee safe usage. Through experimental results and analysis, this research demonstrates that an AI-assisted framework can drastically reduce the search window for missing individuals compared to manual approaches. The combination of automated recognition, easy accessibility, and scalable architecture ensures that the proposed solution has the potential to be adopted in real-world scenarios and assist both local authorities and large organizations working in public safety.Terms— Artificial Intelligence, Computer Vision, Deep Learning, Missing Person Detection, Facial Recognition, Surveillance Systems, Flask, Python, Web Application, Cloud Computing, Ethical AI, Real-Time Processing**

***Index Terms*—Artificial Intelligence, Computer Vision, Deep Learning, Missing Person Detection, Facial Recognition, Surveillance Systems, Flask, Python, Web Application**

I. Introduction

Missing persons pose a major challenge for law enforcement and society, with thousands of cases reported annually across the globe. Traditional approaches rely on manual witness reports, newspaper postings, and police investigations, which are time-consuming and prone to human error. With the advent of Artificial Intelligence and advancements in image recognition, there is a strong opportunity to build automated systems that can assist in faster identification and recovery of missing individuals.

This research paper presents the design and development of an AI-based missing person detection system. The system employs face recognition algorithms trained on large datasets to identify individuals in real-time through surveillance cameras and uploaded photographs. The solution leverages deep learning models, scalable cloud infrastructure, and an intuitive web interface to bridge the gap between manual reporting and automated detection. The proposed framework is built to function as a centralized platform where data from families, police, and surveillance networks can be integrated into one system. Unlike traditional approaches, the AI-driven recognition engine can continuously learn and adapt as new cases are added, improving its accuracy over time. The system is particularly effective in crowded public areas such as bus stations, airports, and shopping malls, where manual tracking becomes nearly impossible. It also supports cross-verification of information through multiple sources, thereby reducing false positives and improving trustworthiness. Furthermore, the integration of live camera feeds and instant notification alerts ensures that responses can be taken in near real-time, potentially saving lives. By combining robust AI models with practical web-based tools, this project aims to demonstrate a scalable and socially impactful solution that can complement and strengthen current law enforcement practices.

II. Literature Review

Numerous studies have explored the application of AI in surveillance and facial recognition. Early works focused on traditional computer vision methods such as Haar cascades and Histogram of Oriented Gradients (HOG). More recent research emphasizes deep learning, particularly Convolutional Neural Networks (CNNs), which significantly outperform earlier methods in accuracy and robustness.

FaceNet and VGGFace have achieved state-of-the-art results in facial verification tasks, making them reliable for real-world applications. Other works also discuss the integration of AI systems with law enforcement and public surveillance infrastructure. Despite these advances, challenges remain in low-light conditions, crowd detection, and ensuring privacy. Our approach builds on this literature by combining transfer learning, Flask-based web deployment, and a scalable architecture. In addition, prior research has highlighted the importance of maintaining high-quality datasets, since biased or incomplete training data can affect recognition accuracy. Several studies also propose hybrid frameworks that combine face recognition with other biometric traits such as gait and voice to improve robustness. Furthermore, the growing use of cloud-based services in similar applications suggests that scalable infrastructures are vital for real-time deployment. Ethical considerations have also been emphasized in multiple works, ensuring that AI-driven solutions respect user privacy and legal constraints. Collectively, these findings underline the relevance of our system and justify its design choices for practical implementation.

III. Proposed Solution

The proposed system consists of the following modules:

- **Data Entry and Management:** Families or authorities can upload photographs and personal details (such as name, age, gender, last seen location, clothing description) through a user-friendly web app. These details are stored in a secure database, which ensures that information is consistently updated and avoids duplication of records. The database also supports encrypted storage to protect sensitive data and provides search and filter options for faster retrieval of records.
- **Face Recognition Engine:** A CNN-based model, leveraging transfer learning from FaceNet, processes uploaded images and generates embeddings for efficient comparison. The engine is designed to improve accuracy by preprocessing images through normalization and noise reduction. It is also capable of handling large datasets by indexing embeddings, which allows faster query responses when matching incoming faces.
- **Live Camera Integration:** The system uses OpenCV to capture real-time video from CCTV or webcams. Faces detected in live streams are matched against the missing persons database. This module can handle multiple video inputs simultaneously, making it suitable for deployment across crowded locations such as airports and bus stations. To improve reliability, frame-skipping and motion detection techniques are implemented, which reduce unnecessary computation on static scenes.
- **Notification and Alerts:** If a match is detected, the system generates an alert through the web interface, highlighting the person, timestamp, and camera ID. The alert log is maintained for further investigation, and notifications can also be pushed to mobile devices or email accounts for quick response. This ensures that the system not only identifies the match but also helps trigger timely actions by the concerned authorities.
- **Web Application Interface:** Built with Flask (Python backend), HTML, CSS, and JavaScript, the system provides dashboards for authorities to update, delete, or review cases. The web application also integrates user authentication and role-based access control to ensure only authorized personnel can access or modify records. Future scalability is considered by enabling modular extensions such as integration with cloud storage and APIs for law enforcement databases.

IV. Software Requirements Specification (SRS)

A. *Functional Requirements*

- Secure database storage for missing person records.
- Real-time face detection and matching.
- Upload portal for families/authorities to submit cases.
- Alerts and logs for matched detections.
- Admin dashboard for managing records.
- Support for editing and updating existing cases to ensure data accuracy.
- Export functionality for generating reports that can be shared with law enforcement.
- Role-based access control for differentiating between admin, operator, and general users.
- Search and filter options to quickly retrieve specific cases from the database.
- Backup and recovery mechanism to prevent data loss during system failures.

B. *Non-Functional Requirements*

- Scalability to handle large video streams.
- Data privacy and secure authentication.
- Cross-platform accessibility via browsers.
- Near real-time processing of surveillance feeds.
- High system availability and fault tolerance to ensure continuous monitoring.
- Low latency communication between modules for efficient alert generation.
- Usability with an intuitive interface for non-technical users such as families.
- Compatibility with existing CCTV infrastructure without requiring major upgrades.
- Maintainability with modular code structure to simplify updates and bug fixes.
- Compliance with ethical and legal standards for data handling and AI deployment.

V. Design

The system design prioritizes modularity and usability. The architecture includes:

- **Frontend:** HTML, CSS, and JavaScript-based user interface for data entry and results. The design focuses on responsiveness so that users can access the system on desktops, tablets, and mobile devices with equal ease. Interactive dashboards provide real-time updates, and data visualization elements such as tables and graphs allow quick interpretation of results.
- **Backend:** Flask framework handling requests, running face recognition, and managing the database. The backend is structured with modular routes and APIs, ensuring that additional services can be integrated in the future without disrupting the core functionality. Security layers such as token-based authentication are applied to prevent unauthorized access.
- **Database:** SQL-based storage for person profiles, photographs, and detection logs. The database is normalized to eliminate redundancy and supports efficient indexing for faster queries. Backup mechanisms are implemented to ensure data safety, and scalability is maintained by allowing migration to cloud-based storage systems if required.
- **Recognition Module:** Python scripts with TensorFlow/Keras for deep learning inference. This module generates embeddings for facial features and compares them against stored records. Optimization techniques such as batch processing and GPU acceleration are used to improve performance. In addition, preprocessing filters like image resizing, contrast adjustment, and noise reduction enhance recognition accuracy.

The architecture is deliberately layered so that failures in one module do not affect the functioning of the entire system. Communication between modules is lightweight and efficient, following RESTful API principles. This design approach ensures robustness, maintainability, and easy future extension of the system.

VI. Methodology

The methodology includes:

1) Collect photographs and metadata of missing persons. This step involves gathering images from families, police records, and public sources while ensuring proper data labeling for training purposes.
2) Preprocess data using OpenCV (cropping, resizing, normalization). Preprocessing also includes face alignment, grayscale conversion, and noise reduction to improve recognition accuracy in diverse lighting conditions.
3) Train/finetune CNN models with embeddings generation. Transfer learning from pre-trained models such as FaceNet or VGGFace is applied to reduce training time and increase precision in real-world scenarios.
4) Deploy Flask backend connected to database and recognition engine. The backend is responsible for handling user requests, managing sessions, and ensuring secure interaction between the web interface and the AI model.
5) Stream live video feeds through web app and detect matches. OpenCV's VideoCapture is used to handle real-time input, and the system supports parallel processing of multiple streams for scalability.
6) Notify authorities through alerts. Notifications include timestamp, camera ID, and a highlighted bounding box around the detected person. A historical log of detections is also stored for auditing and further investigation.
7) Integrate a secure authentication mechanism so that only authorized users can upload or access sensitive data. This prevents misuse of the system and ensures compliance with privacy regulations.
8) Validate results through cross-verification, comparing detected matches with multiple embeddings to reduce false positives and improve confidence scores.
9) Implement a continuous learning mechanism where new confirmed detections can be re-added to the training dataset, enhancing the accuracy of the recognition model over time.
10) Conduct performance evaluation based on metrics such as accuracy, precision, recall, and F1-score, ensuring that the system is tested under real-world constraints like crowded environments and low-quality video feeds.

VII. Implementation

The system is implemented using Python, TensorFlow/Keras, and OpenCV for AI tasks. Flask serves as the backend, connecting with a MySQL database. The web interface is developed using HTML, CSS, and JavaScript, ensuring cross-device compatibility. Real-time face detection is achieved with Haar cascades and DNN-based face detectors, while embedding comparison is done with FaceNet. Notifications are displayed directly on the web app for ease of use.

The database schema is designed to handle large volumes of images and associated metadata, ensuring that queries and comparisons are performed efficiently. Preprocessing techniques such as image normalization, resizing, and face alignment are applied before embeddings are generated to improve recognition accuracy. The Flask backend follows a modular structure, with separate routes for image upload, live camera streaming, and alert generation, making it easier to maintain and extend. OpenCV is integrated with the Flask server through video streaming APIs to enable real-time monitoring from multiple sources simultaneously.

For security, the system incorporates role-based authentication, ensuring that only verified authorities can add, modify, or delete records. The implementation also uses JSON-based communication between the frontend and backend, providing a lightweight and fast mechanism for data transfer. To optimize system performance, embeddings are cached for recently processed faces, reducing redundant computations during continuous video feeds. The deployment is carried out on a local server environment but is designed to be easily migrated to cloud infrastructure for larger-scale deployment. This layered implementation approach ensures the system remains scalable, secure, and responsive under varying operational conditions.

VIII. Results and Discussion

Preliminary testing shows accuracy above 93% in controlled conditions. Performance drops in low-light and heavily crowded environments, though preprocessing techniques improve detection quality. The Flask application demonstrates real-time alerts with minimal latency, and the web-based database management ensures easy record updates. Overall, the system reduces manual effort while improving efficiency in detection.

Further testing with multiple datasets revealed that the recognition engine maintained consistent performance across different age groups and facial orientations, though occlusions such as masks or sunglasses affected results. The average response time for live camera detection was under two seconds, making the system viable for real-time applications. Compared with manual monitoring of CCTV feeds, the proposed solution reduces human workload significantly and provides more reliable outputs.

Another important observation was the system's scalability; it successfully handled simultaneous streams from multiple webcams without a noticeable drop in accuracy. The database structure also allowed smooth retrieval of records even as the dataset size grew. However, the system still requires improvements in handling poor-quality images and heavily compressed video feeds, which remain common in older CCTV networks. Additionally, ethical considerations such as safeguarding personal data and preventing misuse of surveillance remain critical for broader deployment. Overall, the experimental evaluation demonstrates that the system achieves a balance between technical feasibility, usability, and social impact, making it a promising tool for real-world adoption.

IX. Conclusion

This project demonstrates the feasibility of an AI-based missing person detection system deployed as a web application. By integrating CNNs, Flask, and real-time video analysis, the solution bridges the gap between manual searches and automated detection. While promising, further improvements are needed in multi-camera scaling, ethical considerations, and support for multimodal biometrics.

The implementation highlights how deep learning models, when combined with a lightweight web framework, can deliver real-time functionality without requiring high-end infrastructure. The modular nature of the design ensures that additional models or data sources can be incorporated with minimal changes to the system. Another key contribution of this work is the focus on usability, as the web interface was specifically developed to make the technology accessible to non-technical users, including families and local authorities.

Experimental results show that the system can perform effectively in controlled conditions, though performance decreases slightly in low-light and crowded environments. This reinforces the need for continual improvements in preprocessing and adaptive recognition models. The project also raises important questions about data security and privacy, which must be carefully addressed when scaling such solutions to real-world deployment. Overall, the system provides a solid foundation for future development, proving that AI-based automation can significantly reduce the time and effort required to locate missing individuals. It demonstrates not only the technical feasibility but also the potential societal impact of adopting AI-driven surveillance systems for public safety.

- Integration with national databases and smart city infrastructure.
- Gait and voice recognition modules for enhanced accuracy.
- AI-based enhancement of low-quality images.
- Mobile application support for crowdsourced reporting.
- Privacy-preserving techniques for compliance with regulations.

References

[1] Schroff, F., Kalenichenko, D., & Philbin, J. (2015). “FaceNet: A Unified Embedding for Face Recognition and Clustering.” CVPR.
[2] Parkhi, O. M., Vedaldi, A., & Zisserman, A. (2015). “Deep Face Recognition.” BMVC.
[3] Huang, G. B., Ramesh, M., Berg, T., & Learned-Miller, E. (2007). “Labeled Faces in the Wild: A Database for Studying Face Recognition.” UMass Amherst.
[4] Singh, A., & Sharma, R. (2020). “AI in Surveillance: Opportunities and Challenges.” IJCA.
[5] LeCun, Y., Bengio, Y., & Hinton, G. (2015). “Deep learning.” Nature, 521(7553).
[6] Tan, C., et al. (2018). “A Survey on Deep Transfer Learning.” ICMLC.
[7] Garvie, C., Bedoya, A., & Frankle, J. (2016). “The Perpetual Line-Up: Unregulated Police Face Recognition in America.” Georgetown Law.
[8] Raghavendra, R., et al. (2017). “Multimodal Biometrics: Trends and Applications.” Springer.
[9] Neirotti, P., et al. (2014). “Current Trends in Smart City Initiatives.” Cities Journal.
[10] Grinberg, M. (2018). “Flask Web Development: Developing Web Applications with Python.” O'Reilly Media.