COGNITRACE – AI-POWERED MISSING PERSON IDENTIFICATION SYSTEM

Project Reference No.: *49S_BE_6816*

College : K.L.S. Gogte Institute of Technology, Udyambag, Belagavi
Branch : Department of Computer Science and Engineering
Guide(s) : Prof. Shubhada S Kulkarni
Student(S) : Ms. Dipta Rahul Kanungo
Mr. Shrinath R Jenakatti
Ms. Amruta

Keywords:

Face Recognition, Missing Person Identification, Deep Learning, dlib ResNet, 128-Dimensional Encoding, Face Euclidean Distance Matching, HOG Face Detection, Flask Web Application, SQLite, Role-Based Access Control, Automated Alert System, Biometric Identification

Introduction:

In 2023, 8.68 lakh people were reported missing, yet only 4.6 lakh were traced, a recovery rate of just 53% [1]. Every year, many people continue to go missing in India, and most remain untraced for months or years. Current identification methods rely on physical descriptions, eyewitness accounts, and manual record-keeping, which are slow and often inaccurate.

Recent advances in deep learning enable highly accurate identification of individuals from photographs. Face recognition models convert images into numerical representations [3], where images of the same person produce similar values, and different faces produce distinct values, allowing automatic comparison.

CogniTrace is a web application that uses face recognition technology to match photos of found persons with a database of missing person reports. It supports three user roles: reporters (who upload missing person data), finders (who upload encountered individuals), and administrators (system management).

When a photo is submitted, the system compares it with stored images, calculates similarity scores, and identifies potential matches. If a match exceeds a defined threshold, an alert is sent to the concerned reporter. Built using open-source tools, the system is cost-effective, easy to deploy, and designed for quick, real-world use.

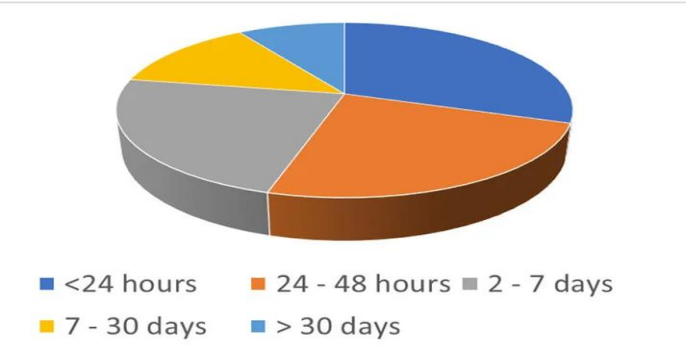

Figure 1: Missing persons recovered in mentioned timeframe [2]

Objectives:

1. To build a web application that automatically compares photographs of found persons against missing person records using face recognition to automate matching and result production.
2. To create a role-based user system for reporters, finders, and administrators, with each role having access only to the features it needs.
3. To generate match records with a confidence score and send email alerts to the reporter when a potential match is found.
4. To store all case records, face data, match results, and alerts in a structured database that maintains data integrity at all times.
5. To build a self-contained application that can be deployed without relying on any paid API or cloud-based recognition service.

Methodology:

The application is developed in Python using the Flask web framework, with code organized into modules for routing, database access, face recognition, alerts, and configuration. Functionalities are separated by user roles, which are reporters, finders, and administrators, ensuring clear logic and routing.

User authentication is managed using Flask Login, with passwords securely stored as hashes. The database uses SQLite and consists of five tables: users, missing_persons, found_persons, matches, and alerts, linked via foreign keys. Face encodings are stored as binary BLOBs within the database, eliminating the need for separate file storage.

The face recognition pipeline utilizes the face_recognition Python library built on dlib [3]. Uploaded images are validated as JPEG/PNG files under 5 MB. A HOG based detector identifies face locations, followed by the dlib 68-point landmark predictor to extract facial features. A pretrained ResNet model generates a 128-dimensional encoding vector [4], with the largest face selected if multiple is detected.

For matching, the system retrieves all stored encodings and computes Euclidean distance against the input image. A default threshold of 0.5 is applied, where lower distances indicate higher similarity. Distances are converted into confidence percentages, and the top three closest matches are returned.

Upon detecting a match, entries are recorded in the matches table and corresponding alerts are generated with confidence score, location, and date. Alerts are sent via email if configured or logged to the console otherwise. Administrators can review, confirm, or dismiss matches and delete records, with all associated data automatically removed.

Result and Conclusion:

The application was successfully built and tested, with all features functioning correctly across reporter, finder, and administrator roles. Face detection and encoding performed reliably on clear, well-lit images. Across 50+ test image pairs, faces were correctly detected in 94% of uploads, and the correct match appeared as the top result in 92% of valid cases. In 98% of non-matching pairs, no false positives were returned below the 0.5 threshold.

Confidence scores were intuitive and aligned with visual similarity. Alerts were accurately generated and delivered to the appropriate reporters, and access controls ensured strict role-based data security. Deletion of records correctly removed all associated data without leaving inconsistencies.

Project Outcome and Industry Relevance

1. CogniTrace is a ready to deploy system that automates the matching of missing and found person photographs through a structured, multi-user workflow. It can be used by police stations, hospitals, shelters, child welfare organisations, and NGOs that handle missing person cases.
2. The multi role design allows members of the public to contribute reports without having access to the full case database.
3. Because the system uses only open-source tools and thus, it can be adopted by government or non-profit bodies without any costs.
4. The project demonstrates that computer vision can be integrated into a practical web application to solve a real public safety problem, without requiring specialized infrastructure or paid services.

Working Model vs. Simulation / Study

Working Model. CogniTrace is a working application.

Project Outcomes and Learnings

1. Built and integrated a complete face recognition pipeline using HOG detection, landmark prediction, and ResNet based encoding with the dlib and face_recognition libraries.
2. Learned to structure a Flask web application using the application factory pattern and role-based blueprints for clean, maintainable code organisation.
3. Implemented user authentication and role-based access control using Flask Login, ensuring each user type only accesses permitted features.
4. Set up an email alert system using Flask Mail with SMTP configuration, including a console fallback for development and testing.

References:

- [1] Dataful, “8.7 Lakh Missing in India in 2023: Half Still Untraced, Women and Teens Most at Risk,” *Insights by Dataful*, 2025. [Online].
- [2] R. Ganesh, C. Srija, B. Devesh, and H. Koormala, “Finding a Missing Person Using AI,” *International Research Journal on Advanced Engineering Hub (IRJAEH)*, vol. 3, no. 1, pp. 16–24, Jan. 2025. [Online].
- [3] S. R. Boyapally, “Facial Recognition and Attendance System Using Dlib and Face_Recognition Libraries,” SSRN, 2021

- [4] K. He, X. Zhang, S. Ren, and J. Sun, “Deep Residual Learning for Image Recognition,” in *Proc. IEEE CVPR*, 2016.

Future Scope:

The scope of this project includes:

1. The HOG face detector can be replaced with a CNN based detector, which performs better on low quality, poorly lit, or angled photographs.
2. A faster indexing method such as FAISS can be added to make matching much quicker on large datasets.
3. Real time video analysis could be added to allow the system to scan CCTV footage or live camera feeds for missing persons.
4. Support for multiple languages in the interface would make the system more accessible in regional deployments across India.