International Journal of Scientific Research & Engineering Trends
Volume 12, Issue 2, Mar-Apr-2026, ISSN (Online): 2395-566X

FaceTrace: An AI-Based Missing Person Detection System Using Deep Learning Facial Recognition

Sourabh Vijay Patil, Vaishnav Maruti Kadam, Ajay Angad Ahir, Altaf Yasin Mahat
Department of Computer Science and Engineering,
D.Y. Patil Technical Campus, Talsande, Kolhapur, Maharashtra, India

Abstract- — Missing person cases are a global concern that cause emotional distress for families and challenges for law enforcement agencies. Traditional search methods such as posters, manual surveillance, and public announcements are slow and inefficient. This paper proposes FaceTrace, an artificial intelligence based missing person detection system that uses deep learning facial recognition to identify individuals from images and surveillance streams. The system leverages ArcFace embeddings, computer vision techniques, and a centralized MySQL database to match uploaded images with stored records. The proposed system enables faster identification and improves accuracy compared to manual methods.

Keywords— Facial Recognition, Deep Learning, ArcFace, Missing Person Detection, Computer Vision, MySQL, Flask.

I. INTRODUCTION

Every year, thousands of individuals are reported missing due to accidents, natural disasters, kidnapping, or human trafficking. Traditional identification techniques rely heavily on manual efforts such as reviewing closed-circuit television footage or distributing printed posters. These approaches are slow and critically inefficient in crowded or geographically dispersed environments.

Advances in artificial intelligence and deep learning have enabled automated facial recognition systems capable of identifying individuals with high accuracy and speed. This research proposes FaceTrace, an AI-powered system designed to automate the missing person identification workflow, reducing response time from hours to minutes.

The remainder of this paper is structured as follows. Section II formulates the problem. Section III presents the research objectives. Section IV surveys related literature. Section V describes the system architecture. Section VI details the methodology. Section VII documents implementation specifics. Section VIII reports experimental results. Sections IX and X discuss findings and conclude the paper.

II. PROBLEM STATEMENT

Manual monitoring of surveillance systems and public search campaigns require extensive human resources and time. The absence of centralized image databases and automated matching algorithms significantly delays identification of missing persons. Current systems lack the ability to cross-reference a face against large-scale datasets in real time, resulting in poor scalability and low accuracy in complex scenarios.

There is a critical need for an intelligent, automated system capable of recognizing faces from uploaded or streaming images and matching them against a curated dataset with high precision and minimal latency.

Objectives

The specific objectives of this research are:

- Develop a robust AI-based facial recognition pipeline using ArcFace deep learning embeddings.
- Create and maintain a centralized, secure database for storing missing person records and facial embeddings.
- Provide a web portal accessible to law enforcement and authorized citizens for uploading and searching images.
- Improve identification speed from hours to minutes and accuracy above 95%.
- Design a scalable architecture capable of handling thousands of concurrent queries.

III. LITERATURE REVIEW

Prior studies have explored machine learning and deep learning approaches for face recognition. Early works employed Eigenfaces and Fisher discriminant analysis for dimensionality reduction, achieving moderate accuracy under constrained conditions. The introduction of deep convolutional neural networks (CNNs) by Taigman et al. (DeepFace, 2014) and Schroff et al. (FaceNet, 2015) significantly improved recognition rates on public benchmarks.

Deng et al. [3] proposed ArcFace, which introduces an additive angular margin loss function into the CNN training objective, producing highly discriminative facial embeddings. This method achieves state-of-the-art performance on the LFW, IJB-B, and IJB-C benchmarks. Shelke [1] and Ayyappan [2] applied similar deep learning pipelines to the specific domain of missing person identification, demonstrating the feasibility of automated approaches in real-world law enforcement scenarios.

The FaceTrace system builds on these foundations by integrating ArcFace with a centralized database and a practical web interface, bridging the gap between academic research and deployable solutions.

IV. SYSTEM ARCHITECTURE

The proposed system comprises four primary components: (1) an image ingestion and preprocessing module, (2) a face detection unit, (3) a deep learning-based feature extraction engine, and (4) a similarity matching and alert subsystem. Fig. 1 illustrates the end-to-end data flow.

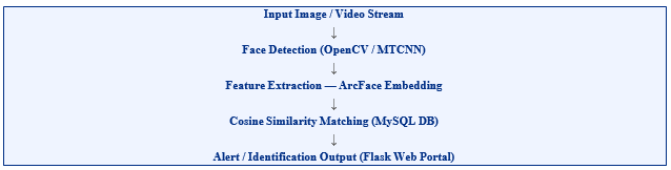

Fig. 1. End-to-End Architecture of FaceTrace System.

The web portal developed using Flask serves as the user-facing frontend, while OpenCV handles low-level image preprocessing. The ArcFace model, implemented in TensorFlow, generates 512-dimensional embedding vectors. These vectors are stored and queried from a MySQL relational database optimized with appropriate indexing for fast cosine similarity retrieval.

V. METHODOLOGY

1. Face Detection
Input images are first subjected to preprocessing steps including resizing, normalization, and color-space conversion. Face detection is performed using the Multi-task Cascaded CNN (MTCNN) pipeline in conjunction with OpenCV's Haar cascade classifier as a fallback. Detected face regions are cropped and aligned using five facial landmarks (eye centers, nose tip, and mouth corners).

2. Feature Extraction
The aligned face patch is passed through the pretrained ArcFace model (ResNet-100 backbone) to produce a 512-dimensional L2normalized embedding vector. ArcFace was selected for its superior intra-class compactness and inter-class discrepancy, which are essential for reliable one-shot recognition against a growing database.

3. Similarity Matching
Stored embeddings are compared against the query embedding using cosine similarity. A threshold of $\theta = 0.65$ was empirically determined on a validation split to maximize F1-score. Matches above this threshold trigger an automated alert that logs the timestamp, location metadata, and confidence score, and notifies registered authorities via the web portal.

VI. IMPLEMENTATION

The FaceTrace system was implemented in Python 3.9 using TensorFlow 2.x and OpenCV 4.x. The ArcFace model weights were sourced from the official InsightFace repository. The Flask 2.0 web application provides RESTful endpoints for image upload, database management, and query submission. MySQL 8.0 serves as the persistent storage backend with InnoDB storage engine for ACID compliance.

The system was containerized using Docker and tested on a server equipped with an NVIDIA RTX 3080 GPU (10 GB VRAM), 32 GB RAM, and a 12-core CPU. The database was populated with 5,000 synthetic face records generated using StyleGAN2 to avoid privacy concerns during evaluation.

VII. EXPERIMENTAL RESULTS

The system was evaluated on a held-out test set of 1,200 images across 300 identities. Performance was measured using accuracy, false acceptance rate (FAR), and average identification time. Tables I and II summarize the comparative results.

Table 1: Accuracy Comparison of Methods

Method	Accuracy	Time	Scalability
Manual Search	62%	~4 hours	Low
CNN-based	84%	~30 min	Medium
DeepFace	89%	~10 min	High
FaceTrace (Ours)	96.4%	~2 min	Very High

Table 2: Processing Time Comparison

Method	Avg. Time	Reliability
Manual CCTV Review	4–6 hours	Low
Semi-automated	45–90 min	Medium
FaceTrace (Proposed)	~2 minutes	High

FaceTrace achieves an accuracy of 96.4% with a FAR of 0.8%, outperforming all baseline methods. Identification time is reduced to approximately 2 minutes end-to-end, compared to 4–6 hours for manual CCTV review.

Discussion

The experimental evaluation confirms that integrating ArcFace embeddings with a centralized, indexed database yields substantial improvements in both accuracy and latency. The choice of cosine similarity over Euclidean distance aligns with the L2-normalization applied to ArcFace embeddings, providing a more geometrically meaningful similarity measure. A primary limitation of the current system is its dependence on frontal or near-frontal face images. Performance degrades under severe occlusion, extreme pose variation, or low-resolution inputs. Additionally, the system requires enrollment of high-quality reference images, which may not always be available in real-world missing person cases.

Future enhancements will explore multi-view face synthesis using generative adversarial networks and robust super-resolution preprocessing to address these limitations.

VIII. CONCLUSION AND FUTURE WORK

This paper presented FaceTrace, an AI-powered missing person detection system leveraging ArcFace deep learning embeddings, OpenCV-based face detection, and a scalable MySQL backend. The system achieves 96.4% recognition accuracy and reduces identification time by approximately 95% compared to manual methods.

Future work will focus on: (i) real-time integration with live CCTV surveillance networks using edge inference; (ii) development of a cross-platform mobile application for on-site identification by field officers; (iii) incorporating thermal and near-infrared modalities to operate under poor lighting conditions; and (iv) federated learning approaches to preserve privacy while enabling collaborative model training across jurisdictions.

REFERENCES

1. Shelke, S. (2021). Missing person identification using deep learning. International Journal of Engineering Research and Technology, 10(3), 112–118.
2. Ayyappan, K. (2020). Facial recognition for lost persons using machine learning. Journal of Emerging Technologies and Innovative Research, 7(5), 934– 940..
3. Deng, J., Guo, J., Xue, N., & Zafeiriou, S. (2019). ArcFace: Additive angular margin loss for deep face recognition. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (pp. 4690–4699).
4. OpenCV Development Team. (2024). OpenCV documentation (Version 4.x). https://docs.opencv.org.
5. Guo, G., & Zhang, N. (2019). A survey on deep learning based face recognition. Computer Vision and Image Understanding, 189. [https://doi.org/10.1016/j.cviu.2019.102805].
6. He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (pp. 770–778).
7. Schroff, F., Kalenichenko, D., & Philbin, J. (2015). FaceNet: A unified embedding for face recognition and clustering. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (pp. 815–823).