International Journal of Creative and Open Research in Engineering and Management
ISSN: **3108-1754 (Online)**
Volume 02 Issue 04 April-2026 | Impact Factor: 3.5

Nexus Forensics: A Real-Time AI Surveillance System for Multi-Camera Suspect Detection and Biometric Tracking

Mohan M, Manoj Kumar G, Muthu Prasath M, Subhash B
Department of Artificial Intelligence and Data Science
Chettinad College of Engineering and Technology, Anna University
Karur, Tamil Nadu, India

Ms. Anitha R
Assistant Professor, Dept. of AI and Data Science
Chettinad College of Engineering and Technology, Anna University
Karur, Tamil Nadu, India

***Abstract*—**Traditional forensic investigations rely heavily on passive CCTV monitoring and manual sketch identification, resulting in subjective, time-consuming, and delayed suspect tracking. This paper introduces the Nexus Forensics AI Sys-tem, an end-to-end, multi-tier active surveillance architecture designed to bridge the modality gap between static forensic records and live video feeds. The proposed system orchestrates a three-tier technology stack comprising a React-based interactive frontend, a Node.js/Express backend for data logistics, and a highly optimized Python/FastAPI AI engine. To ensure real-time efficiency across multiple IP and web cameras, the com-puter vision pipeline utilizes a two-stage detection mechanism: Ultralytics YOLOv8s for structural person detection, followed by RetinaFace for precise facial boundary extraction. Biometric matching is executed using the ArcFace model, calculating cosine-distance matrices against stored identities. Upon a confirmed match, the system automates evidence recording, captures live GPS telemetry, and dispatches immediate forensic alerts. This architecture demonstrates a scalable, proactive approach to modern law enforcement and smart city security.

Index Terms—Forensic Surveillance, Face Recognition, Deep Learning, YOLOv8, ArcFace, Real-time Tracking, Biometrics**.**

I. Introduction

Forensic identification plays a fundamental role in modern law enforcement. Historically, investigators have relied on eyewitness descriptions translated into composite sketches by trained artists. While foundational, this process is subjective and introduces a significant modality gap when attempting to match hand-drawn sketches or static photographs against real-world, low-resolution surveillance footage [1]. Furthermore, traditional CCTV systems act as passive recording devices, requiring hundreds of hours of manual review after an incident has occurred.

With the rapid advancement of computer vision, there is a critical need to transition from passive video recording to active, automated threat detection. The Nexus Forensics AI System addresses this by deploying a distributed software architecture capable of tracking known suspects across various live camera streams. By integrating lightweight structural detection models with heavyweight biometric extraction algo-rithms, the system minimizes computational overhead while maximizing real-time accuracy.

II. Related Work

Previous surveys in the domain of AI-based forensic sketch drawing and recognition systems [2] have highlighted the limitations of handcrafted features (such as Local Binary Patterns) and the shift toward deep learning models like Convolutional Neural Networks (CNNs). While transfer learn-ing models (VGG16, ResNet) and Generative Adversarial Networks (GANs) have successfully reduced the sketch-to-photo modality gap [6], deploying these heavily parameter-ized models on continuous, live multi-camera streams often results in severe latency and network bottlenecking. The Nexus system builds upon these theoretical frameworks by implementing an optimized, multi-threaded inference pipeline designed specifically for edge and cloud stream deployments.

III. Theoretical Foundations & Mathematical Models

To ensure real-time performance without sacrificing bio-metric accuracy, the Nexus Forensics system relies on a mathematical decoupling of structural detection and facial embedding.

A. Stage 1: Structural Detection via YOLOv8

Traditional sliding-window face detectors are computation-ally prohibitive for high-resolution IP streams. Nexus employs YOLOv8s (You Only Look Once, version 8 small) [5] to perform a single-pass regression to detect the broader *Person* class. YOLOv8 abandons anchor-box logic in favor of an anchor-free, center-based approach, optimizing the detection of human figures even when faces are partially occluded. The model optimizes a joint loss function combining bounding box

regression and classification loss. For bounding box precision, it utilizes Complete Intersection over Union (CIoU) loss:

$$L_{\text{CIoU}} = 1 - \text{IoU} + \frac{\rho^2(b,\, b^{\text{gt}})}{c^2} + \alpha v \qquad (1)$$

Where ρ is the Euclidean distance between the center points of the predicted box b and ground truth box b^{gt}, c is the diagonal length of the smallest enclosing box covering both, and v measures the consistency of the aspect ratio. This ensures the cropped regions passed to the next stage are tightly bounded around the suspect's body.

B. Stage 2: Biometric Extraction via ArcFace

Once a person is cropped, RetinaFace [4] extracts the precise facial landmarks. The facial crop is then processed by the ArcFace (Additive Angular Margin Loss) model [3]. Unlike traditional Softmax loss, which only maximizes inter-class variance, ArcFace introduces an additive angular margin to the hypersphere, forcing the model to learn highly discriminative facial features. This is critical for matching low-quality CCTV frames with high-quality database images or sketches.

The ArcFace loss function is defined as:

$$L_{\text{ArcFace}} = \frac{1}{N}\sum_{i=1}^{N} \log \frac{e^{s(\cos(\theta_{y_i}+m))}}{e^{s(\cos(\theta_{y_i}+m))} + \sum_{j=1,\, j\neq y_i}^{n} e^{s\cos\theta_j}} \qquad (2)$$

Where θ_{y_i} is the angle between the weight vector and the feature vector, s is the hypersphere radius (feature scale), and m is the additive angular margin penalty. By projecting embeddings into this highly structured hypersphere, the Nexus system can confidently utilize Cosine Distance to measure the similarity between the live CCTV face and the stored suspect embedding.

IV. System Architecture

The Nexus Forensics AI System is structured as an end-to-end multi-tier application. As illustrated in Fig. 1, the system is divided into three distinct operational tiers handling the client interface, data logistics, and AI inference simultaneously.

A. Tier 1: Frontend (Client Interface)

The user interface is engineered using React and TypeScript via Vite, styled with TailwindCSS. Hosted on port 5173, the dashboard provides a centralized command center for security operators, featuring three core modules:

- **Upload & Generate:** Facilitates the ingestion of original suspect photographs or AI-generated sketches into the system's active memory.
- **Monitor:** Renders real-time video feeds from active camera sources, overlaid with dynamic YOLO bounding boxes and immediate alert flags.
- **Results:** Provides a historical ledger of detections queried from the database, offering direct playback of stored video evidence.

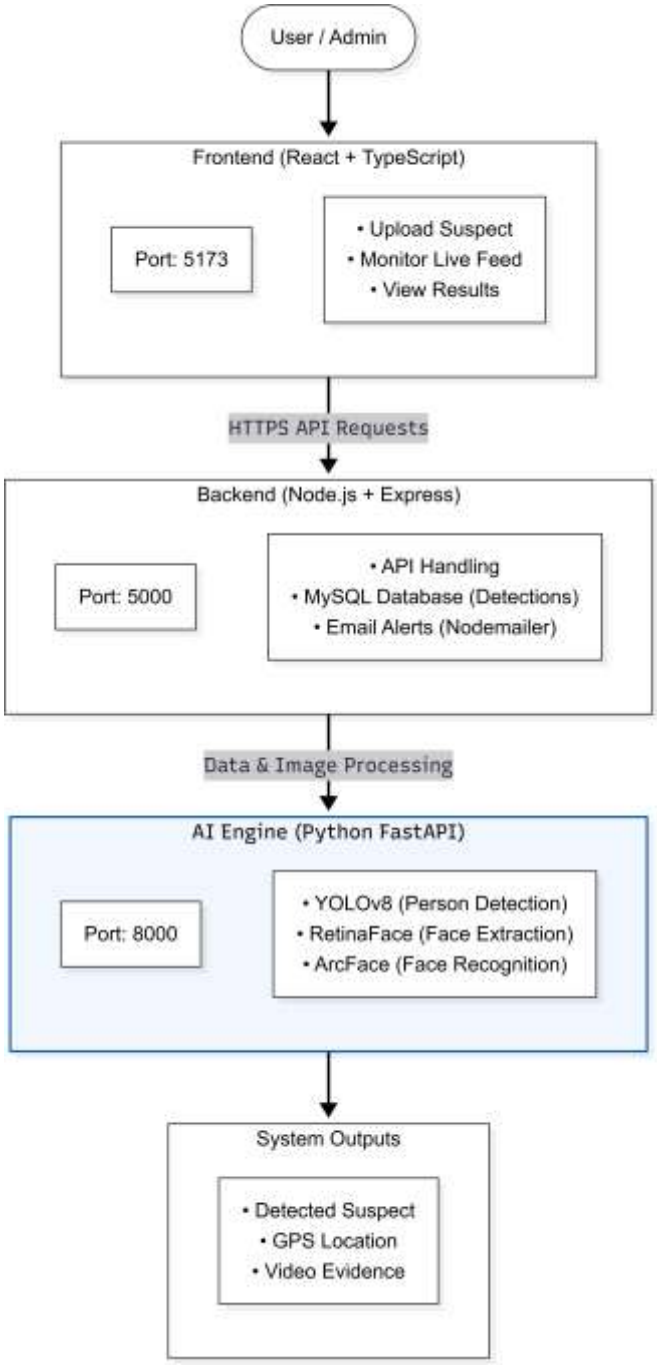
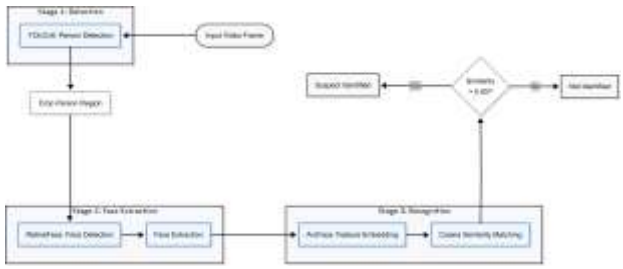


Fig. 1. The end-to-end multi-tier architecture of the Nexus Forensics AI Surveillance System, detailing the React frontend, Node.js backend, and FastAPI inference pipeline.



B. Tier 2: Backend (API & Data Management)

Operating on a Node.js runtime with an Express framework (Port 5000), the backend serves as the logistical bridge. It utilizes Multer for multi-camera file handling and Nodemailer for automated email dispatch. Persistent data storage is managed via a MySQL relational database (*forensic_db*), utilizing connection pooling to log suspect bounding coordinates, GPS location statistics, and temporal data within a dedicated detections table.

C. Tier 3: AI Engine (Computer Vision & Inference)

The core analytical brain is a Python-based FastAPI service (Port 8000). OpenCV handles real-time video stream buffering, intelligently flushing buffers to mitigate MJPEG IP network lag. The inference engine is powered by three distinct machine learning models:

- **Ultralytics YOLOv8s:** Deployed for the swift detection of broader classes (e.g., Persons).
- **RetinaFace:** Executed only on the localized crops generated by YOLO to extract precise facial boundaries.
- **ArcFace:** A heavyweight biometric model used to generate embedding matrices for identity matching.

D. Multi-Threading and Asynchronous Processing Optimization

A primary bottleneck in real-time surveillance is the I/O blocking caused by decoding RTSP/IP camera streams over a network. If the AI inference takes longer than the frame arrival rate, the stream will buffer, leading to extreme latency.

The Nexus AI Engine solves this using the Python *ThreadPoolExecutor* and a strictly throttled asynchronous pipeline.

1) **Frame Ingestion Thread:** A dedicated daemon thread continuously reads the OpenCV cv2.VideoCapture buffer. It immediately flushes outdated frames, keeping only the most recent frame in memory.
2) **Inference Throttling:** To preserve CPU/GPU cycles, the AI pipeline does not process 30 frames per second (FPS). It utilizes a temporal throttle, evaluating exactly 1 frame per second per camera.
3) **Temporal Confirmation Logic:** To drastically reduce False Positives, the matching engine implements a strict Consecutive Frame Rule. A suspect is only flagged if the Cosine Similarity exceeds the minimum threshold (e.g., > 0.60 for photos, > 0.45 for sketches) for N consecutive evaluated frames (where $N = 3$).

V. Methodology & Execution Workflow

The operational pipeline is divided into four chronological phases designed to ensure accuracy and reduce false positives, as mapped out in Fig. 2.

A. Phase 1: Target Registration

The workflow initiates when a security operator submits a suspect target via the React frontend. The Node.js backend intercepts the payload, transmitting it via FormDatato the FastAPI */upload-suspect* endpoint. The image undergoes preprocessing via Contrast Limited Adaptive Histogram Equalization (CLAHE). ArcFace then encodes a strict biometric embedding matrix, cached in memory. User-uploaded sketches are dynamically enhanced via Gaussian Blur and inverted division.

Fig. 2. The AI execution pipeline, demonstrating the data flow from multi-camera input, through YOLOv8 and ArcFace processing, to final decision matching.

B. Phase 2: Live Inference & Face Matching

The system begins concurrent video capture polling across multiple sources. A background thread fetches live GPS telemetry from the physical device locations. First, YOLOv8 validates the structural presence of a person. RetinaFace is then applied exclusively to those bounding boxes. Once a face is captured, ArcFace generates an active embedding vector **A**. This is compared against the stored suspect's matrix **B** using Cosine Similarity:

$$\text{Cosine Similarity} = \frac{\mathbf{A} \cdot \mathbf{B}}{|\mathbf{A}||\mathbf{B}|} \tag{3}$$

C. Phase 3: Evidence Recording

Upon a confirmed match, an isolated forensic recording thread is triggered. The active stream is written locally into a dynamically generated directory (*dataset/evidence/{camera_type}/*) as an MP4 or WebM loop, capturing approximately five seconds of 20 FPS footage.

D. Phase 4: Alerting & Review Logistics

The backend commits the timestamped detection footprint to the MySQL database. Concurrently, an alert pipeline constructs a stylized HTML notification containing GPS coordinates, a Google Maps lookup, suspect metadata, and the video evidence frame.

VI. Implementation Advantages

By decoupling the structural person detection from intensive facial extraction, the Nexus architecture prevents hardware throttling during crowded scenes. The integration of intelligent buffer flushing ensures that IP camera streams do not fall out of sync with real-time events.

VII. Experimental Setup & Evaluation Metrics

To validate the Nexus Forensics system, experiments evaluate both biometric accuracy and logistical latency. The frontend output of these evaluations is captured in Fig. 3.

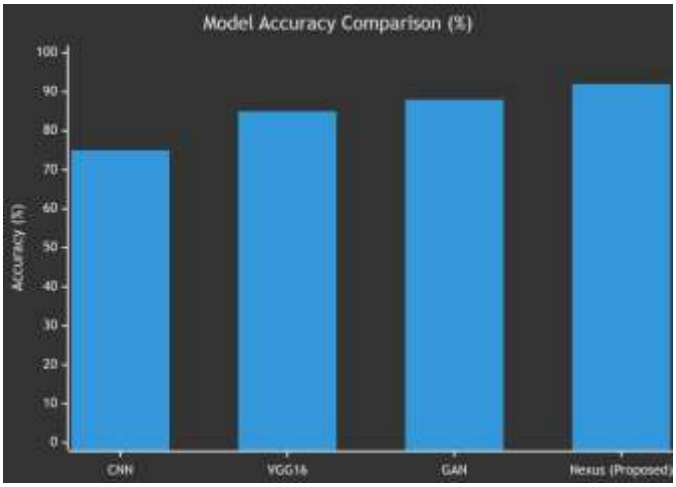


Fig. 3. The React-based Results Dashboard, illustrating the real-time detection logs and video playback capabilities generated post-inference.

A. Hardware and Environment Specifications

The backend Node.js server and Python FastAPI engine were hosted on a machine utilizing an NVIDIA RTX series GPU with CUDA and cuDNN optimization layers. Video ingestion was tested using a 1080p webcam and a simulated MJPEG mobile IP camera stream.

B. Dataset Integration

The CUHK Face Sketch Database (CUFS) is utilized to determine optimal Cosine Distance degradation for forensic sketches. For live CCTV validation, the WIDER FACE dataset parameters are used to test detection capabilities under extreme variations in scale and pose.

C. Evaluation Metrics

The system's performance is quantified using the following standard metrics:

- **Precision and Recall:**

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{4}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{5}$$

- **F1-Score:** The harmonic mean of precision and recall.
- **End-to-End Latency:** Measured in milliseconds (ms).

VIII. Ethical Considerations & Data Privacy

The Nexus Forensics system is architected with "Privacy by Design" principles. Unlike mass-surveillance, it operates on targeted watchlist logic.

- **Ephemeral Data:** Video frames are held in volatile RAM. If no match is found, the frame is immediately dropped.
- **Encrypted Storage:** Only confirmed forensic evidence is written to the persistent database, ensuring the biometric data of innocent bystanders is never stored.

IX. Conclusion

The Nexus Forensics AI System bridges the gap between static forensic inputs and active, multi-camera surveillance. The integration of automated recording and GPS-tagged alerting provides a scalable solution for modern law enforcement. Future work will focus on expanding tolerance for severe occlusion.

Acknowledgment

The authors express their sincere gratitude to Ms. Anitha R, Assistant Professor, for her invaluable guidance and continuous support throughout this research.

References

[1] A. Firdouse, et al., "Enhancing Forensic Face Construction with Real-Time Recognition," *2025 IEEE ICWITE*, pp. 1-6, 2025.
[2] M. Mohan, et al., "AI-Based Forensic Sketch Drawing and Recognition Systems," *IJCREM*, vol. 2, no. 3, 2026.
[3] J. Deng, et al., "ArcFace: Additive Angular Margin Loss for Deep Face Recognition," *CVPR*, 2019, pp. 4690-4699.
[4] J. Deng, et al., "RetinaFace: Single-Shot Multi-Level Face Localisation in the Wild," *CVPR*, 2020, pp. 5203-5212.
[5] G. Jocher, et al., "Ultralytics YOLO," 2023. [Online].
[6] C. Galea and R. A. Farrugia, "Matching Software-Generated Sketches to Face Photographs," *IEEE T-IFS*, vol. 13, no. 6, pp. 1421-1431, 2018.