MEC-based Smart Surveillance in 5G/6G Networks: Intrusion Detection and Loitering Monitoring

Deveshwar Singh and Samaresh Bera, *Senior Member, IEEE*
Dept. of Computer Science and Engineering
Indian Institute of Technology Jammu, India
Email: s.bera.1989@ieee.org

Ankur Bansal, *Senior Member, IEEE*
Dept. of Electrical Engineering
Indian Institute of Technology Jammu, India
Email: bansal.ankur143@gmail.com

***Abstract*—Mobile Edge Computing (MEC), empowered by deep learning techniques, plays a pivotal role in enabling real-time service delivery for applications with stringent latency requirements in 5G and beyond-5G (6G) networks. Although numerous studies have proposed resource allocation and model placement schemes for MEC environments, most of these efforts remain theoretical and are validated primarily through simulations or emulations.**

In this paper, we present a practical implementation of an MEC-based smart surveillance system deployed and tested on a 5G testbed. The system aims to detect human presence, identify intrusions, and monitor loitering behavior in real time. To achieve this, we employ two state-of-the-art deep learning models – `InsightFace` for intrusion detection and `YOLOv8-Pose` for loitering analysis. The models are deployed at the MEC of the 5G testbed to minimize latency with optimize inference efficiency. Experimental results demonstrate that the proposed system can accurately detect and monitor individuals within a few hundred milliseconds, validating the effectiveness of MEC-based deep learning deployment for real-time surveillance. Furthermore, we discuss key challenges, limitations, and lessons learned from our testbed-based implementation.

***Index Terms*—Smart surveillance, 5G/6G-based Internet-of-Things, Intrusion Detection, Face Recognition, Deep Learning**

I. Introduction

The evolution of cellular networks into fifth- and sixth-generation (5G/6G) systems has achieved remarkable success in supporting a wide range of emerging applications, including autonomous vehicles, smart surveillance, and smart healthcare [1]. These real-time applications impose stringent quality-of-service (QoS) demands in terms of latency, reliability, and bandwidth. Mobile Edge Computing (MEC) integrated with 5G/6G networks has emerged as a promising paradigm to meet these latency-sensitive requirements by placing the services near to the end-users [1]–[3].

Recent research has explored the use of 5G-enabled smart surveillance systems for crowd monitoring and object detection [4]–[6]. Ahmed et al. [7] investigated edge-based person detection using transfer learning, employing the CenterNet deep learning model to identify individuals from smart camera feeds. Similarly, Wei et al. [8] proposed a port surveillance framework for detecting human and vehicular movements. However, beyond general person detection, there remains a critical need to address more complex surveillance tasks such as intrusion detection and loitering behavior analysis.

Several subsequent studies [8]–[12] have proposed MEC-assisted surveillance architectures, though most have remained theoretical, focusing on analytical or simulation-based evaluations. To bridge this gap, this paper presents the development of a practical MEC-based real-time smart surveillance system empowered by artificial intelligence and machine learning (AI/ML) techniques within 5G/6G environments. The proposed system integrates live Real-Time Streaming Protocol (RTSP)-based video feeds from cameras operating over 5G/6G networks and applies AI/ML models for face detection, anomaly recognition, and loitering monitoring—extending beyond conventional video recording capabilities.

Leveraging 5G/6G connectivity and MEC-based computation, the proposed system enables the transmission of high-quality video streams with minimal latency, making it particularly suitable for large-scale or remote surveillance environments. By deploying AI/ML processing units at the network edge, detection models are executed locally, significantly reducing network traffic and improving real-time responsiveness. The system's modular design allows operators to selectively activate functionalities such as face recognition, restricted zone monitoring, and behavioral analysis based on specific application requirements.

The key contributions and features of the proposed system are summarized as follows:

- Real-time Face Recognition: Design and implementation of a robust, modular architecture capable of accurately identifying authorized personnel and distinguishing unknown individuals under diverse environmental conditions.
- Zone-based Intrusion Detection: Development of a flexible intrusion detection framework that supports dynamic definition of restricted areas through polygonal zone annotation, ensuring reliable detection of unauthorized access.
- Behavior and Loitering Analysis: Integration of advanced human-pose estimation methods—specifically leveraging YOLOv8-Pose—to effectively detect and analyze loitering and other suspicious behaviors.
- Edge-based Inference: Deployment of AI/ML models

on MEC nodes to perform on-site inference, thereby reducing communication overhead, improving system responsiveness, and supporting real-time decision-making.
- Experimental Evaluation: Implementation and validation on a 5G testbed to assess system feasibility in terms of responsiveness, latency, and detection accuracy of the proposed AI/ML techniques.

The remainder of this paper is organized as follows: Section II presents the underlying framework and algorithms employed in the surveillance system. Section III describes the experimental setup on the 5G testbed. Section IV reports the experimental results. Section V discusses the key challenges and limitations encountered, and Section VI concludes the paper.

II. Proposed System: Face Recognition, Intrusion Detection, and Loitering Monitoring

Figure 1 illustrates the architectural components of the proposed smart surveillance system. The SP/IP camera captures real-time video streams and transmits them to the MEC server via a 5G/6G communication network. At the MEC, two AI/ML models – `InsightFace` [13] and `YOLOv8-Pose` [14] – are deployed for intelligent video analytics. Specifically, `InsightFace` is employed for face recognition and feature embedding extraction, while `YOLOv8-Pose` is utilized for human pose estimation and loitering behavior analysis.

These models were selected based on the following considerations:

- High accuracy and efficiency: `InsightFace` delivers state-of-the-art recognition accuracy even under limited computational resources, ensuring robust performance across diverse environmental conditions.
- Lightweight yet effective design: `YOLOv8n-Pose` possesses a compact architecture (6.2 million parameters) while maintaining sufficient precision for real-time behavioral analysis.
- Real-time inference capability: Both models can achieve real-time performance (with 25 FPS) when deployed on GPUs or edge AI platforms such as the NVIDIA Jetson Xavier NX.

A detailed discussion of these models and their integration into the proposed system is provided in the subsequent sections.

A. Face Recognition using `InsightFace`

`InsightFace` provides high-performance face analysis, including recognition, alignment, detection, and feature extraction [13]. For precise face recognition, we utilize three core components within `InsightFace`:

- `SCRFD`: A lightweight yet accurate face detector that outputs bounding box coordinates for each detected face.
- `Landmark Regression`: Predicts key facial landmarks (eyes, nose, and mouth) to enable precise alignment.
- `ArcFace (ResNet100)`: Produces a 512-dimensional embedding that uniquely represents each face. These embeddings are compared with entries in the known-person database using cosine similarity:

$$\text{similarity}(A, B) = \frac{A \cdot B}{\|A\| \times \|B\|}.$$

A match is accepted if the similarity exceeds a predefined threshold (typically 0.6–0.8). Faces that do not meet this criterion are classified as unknown.

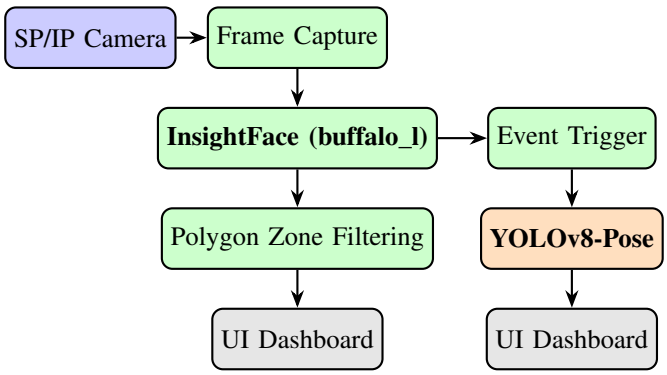


Fig. 1: Proposed pipeline integrating InsightFace for recognition and YOLOv8-Pose for loitering detection.

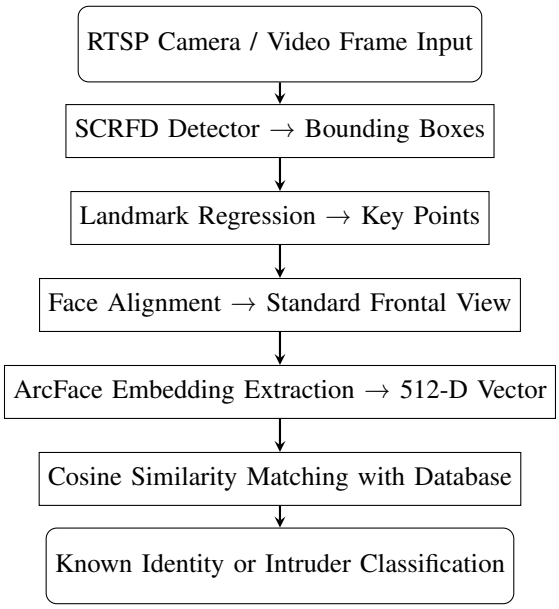


Fig. 2: Workflow of the `InsightFace`-based recognition system

The integration of these modules enables real-time operation while maintaining robustness against variations in lighting, occlusion, and head pose. Figure 2 illustrates the flowchart of face detection using `InsightFace`.

B. Loitering Detection using `YOLOv8-Pose`

In the proposed system, `YOLOv8-Pose` is employed for loitering detection. Unlike conventional object detectors that

simply localize people with bounding boxes, `YOLOv8-Pose` identifies key body points, effectively constructing a skeletal representation. This allows the system to track individuals accurately, even if the camera or the person moves.

Loitering is detected by monitoring the movement of these keypoints over time. If a person remains within a defined area beyond a preset duration, the system flags it as loitering. This approach reduces false alarms and ensures consistent tracking of the same individual, enhancing system reliability. Figure 3 presents the flowchart for loitering detection using `YOLOv8-Pose`.

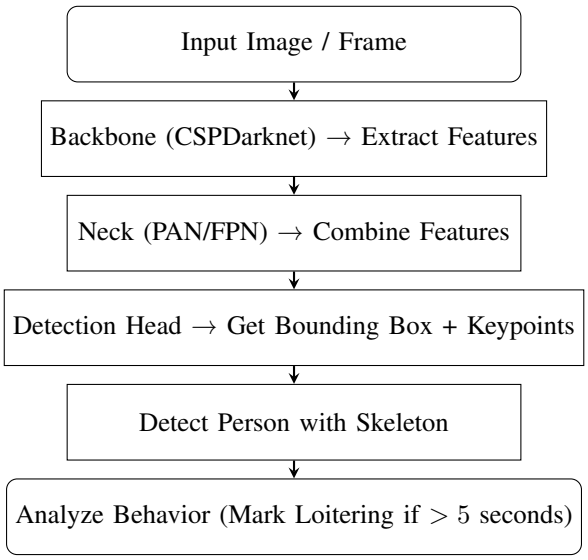


Fig. 3: Workflow from input image to behavior analysis using `YOLOv8-Pose`

C. System Toggles

The system supports configurable toggles via a custom script, including:

- `zone_logic_enabled`: Enables or disables polygon-based zone monitoring.
- `intrusion_logic_enabled`: Enables or disables marking intrusions in zones.
- `loitering_enabled`: Toggles the loitering detection module.
- `skeleton_view_enabled`: Displays keypoint skeletons in the heatmap stream.
- `image_saving_enabled`: Controls saving of cropped face images.

D. Stream Handling

- Both face recognition and pose/heatmap modules utilize the same RTSP video stream.
- The heatmap stream displays pose skeletons and loitering labels.
- The primary face stream displays bounding boxes, names, and intrusion zones.

III. Experimental Setup

We evaluate the performance of the proposed system using a 5G testbed at the Indian Institute of Technology Jammu, India. Figure 4 illustrates the testbed setup, which includes 5G/6G cameras, a gNB base station, the MEC server, and the 5G core network. Both `InsightFace` and `YOLOv8-Pose` models are deployed at the MEC to meet the stringent latency requirements of the surveillance application. The implementation leverages several libraries, including `OpenCV`, `Flask`, `Ultralytics`, `Pandas`, and `NumPy`, to support model execution and system integration.

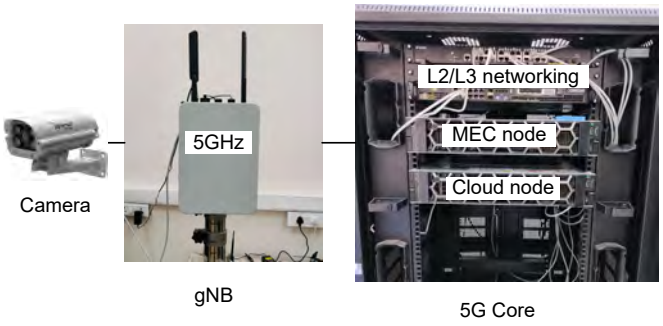


Fig. 4: 5G testbed with smart surveillance camera connected to base-station

IV. Performance Evaluation

Since this work focuses on real-time surveillance using live RTSP video streams, no standard datasets were employed. Instead, the system was tested on live feeds from 5G-enabled IP cameras placed in controlled environments simulating both restricted zones and public areas. The test scenarios included authorized personnel, unknown visitors, and staged intrusion attempts. Various lighting conditions and movement patterns were recorded to evaluate system responsiveness and accuracy. Figure 5 illustrates an example where an authorized person and an intruder are detected within the same frame, while Figure 6 shows a heatmap view highlighting normal and loitering behavior.

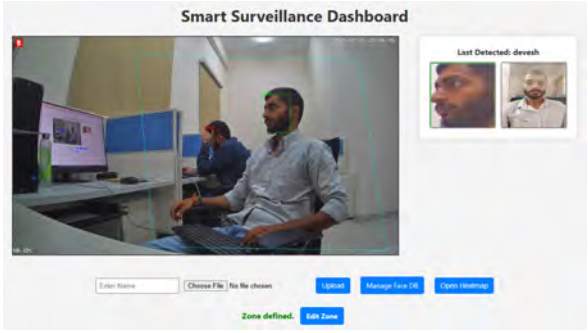
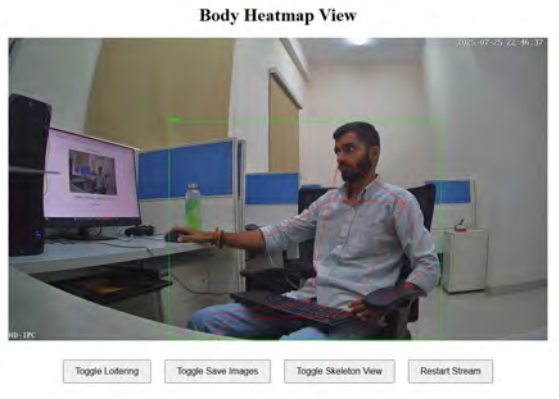
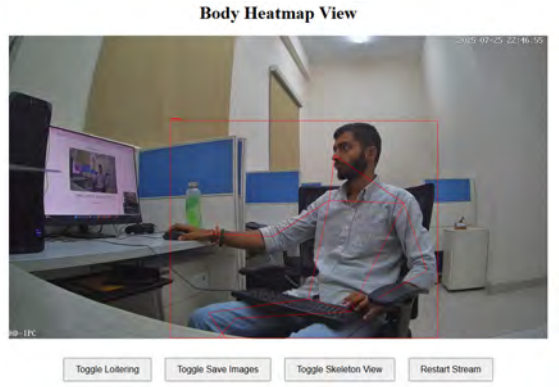


Fig. 5: Both the known person and intruder in the same frame are marked separately



(a) Skeleton view without loitering alert



(b) Skeleton view with loitering alert

Fig. 6: Comparison of body heatmap view in normal state and during loitering detection

A. Detection Accuracy

Detection accuracy reflects the system's ability to correctly identify registered individuals in camera frames and is influenced by factors such as pose, lighting, occlusion, and crowd density. Table I summarizes the number of frames tested along with true positives (TP), false positives (FP), false negatives (FN), and the corresponding percentages for accuracy, precision, and recall. Detection errors primarily occurred when faces were non-frontal or partially occluded, occasionally leading to misclassification of known individuals as intruders.

TABLE I: Comparison of detection performance in single-person and multi-person scenarios

Metric	Single-Person Scenario	Multi-Person Scenario
Frames Tested	300	300
True Positives (TP)	284	287
False Positives (FP)	73	81
False Negatives (FN)	37	234
Accuracy (%)	72.08	47.67
Precision (%)	79.55	77.99
Recall (%)	88.47	55.09

Single-Person Performance: In scenarios with only one individual, the system achieved an accuracy of 72.08% with a high recall of 88.47%, indicating that most frames correctly detected the person. False positives were rare and mostly resulted from misclassification or threshold mismatches. These results demonstrate that system reliability is highest when a single person is present, particularly if the face is frontal and well-lit.

Multi-Person Performance: Accuracy dropped to 47.67%, with recall decreasing to 55.09% in multi-person scenarios. This reduction was mainly due to partial occlusion, non-frontal face poses, or individuals turning away from the camera, causing frequent misclassification as "intruders." The results indicate that recognition accuracy is significantly affected by pose variation, occlusion, and overlapping bounding boxes. Potential improvements include using pose-robust models, multi-face tracking, and optimized similarity thresholds.

Precision Stability: Precision remained relatively stable across both single- and multi-person scenarios (approximately 78–80%), indicating that when the system identified a face as a registered individual, it was usually correct. However, recall suffered considerably in crowded scenarios due to missed detections.

B. Detection Time

As discussed in Section I, latency is a critical QoS requirement for real-time applications. To evaluate this, we measured the detection time, which includes the total processing of frames at the MEC. Processing time is defined as the duration from reading a frame from the camera to completing all operations, including detection, labeling, and image saving. Table II summarizes the processing times in milliseconds, with the effective FPS representing the average number of frames processed per second for detection.

TABLE II: Processing time at the MEC in milliseconds

Mean	Median	Min	Max	Effective FPS
218.29	285.23	39.91	344.49	4.58

Table III isolates the time spent solely on detection, excluding the frame acquisition and preprocessing steps. Results indicate that detection accounts for the majority of processing time, averaging approximately 155 ms per detection. Scene complexity significantly affects latency, as processing time grows with the number of detected objects and frame resolution. Consequently, system tuning must carefully balance detection accuracy, processing latency, and real-time responsiveness according to deployment requirements.

We also evaluated the effect of input downscaling on latency. Table IV presents network and inference latency for various downscale factors. At a factor of 0.5, network and resizing overheads remain minimal, providing an efficient configuration while maintaining acceptable detection reliability. At a factor of 0.75, detection accuracy remains high,

TABLE III: Detection time in milliseconds

Mean	Min	Max
155.45	39.64	237.70

TABLE IV: Comparison of latency metrics across different downscale factors.

Downscale Factor	Frames	Average FPS	Average Network Latency	Average Inference Latency
0.5	300	3 - 4	4 - 5 ms	222.12 ms
0.75	300	2 - 3	11 - 12 ms	244.6 ms
1.0	300	4 - 5	3 - 4 ms	194.2 ms

but the system experiences slower real-time response due to increased computational overhead. Processing at the original resolution achieves the highest throughput (4–5 FPS) with minimal network latency (3–4 ms) and reduced inference latency (194.2 ms), benefiting from more efficient GPU utilization. However, higher-resolution frames increase the likelihood of false positives, reducing overall precision. These observations highlight the trade-offs between speed, accuracy, and detection reliability.

C. Key Insights

From the analysis, three key trends emerge:

- **Resolution vs. Latency:** Full-resolution frames reduce inference latency but increase false positives, while aggressive downscaling improves robustness at the cost of slower processing.
- **Optimal Trade-off:** A downscale factor of 0.5 provides a balanced configuration, offering reliable detection while maintaining near real-time performance.
- **Scene Dependence:** System latency is affected by scene complexity, with higher resolution and a greater number of detected objects leading to increased processing times.

D. Alert Latency on Intrusion Detection

Unlike detection and inference, which consistently operate within the 200–300 ms range, the end-to-end alert latency exhibits greater variability. This latency measures the time elapsed from intruder detection to the delivery of an alert to the user via Telegram. Empirical measurements show an average latency of 4.2 seconds, ranging from 3 to 5 seconds. As shown in Figure 7, the alert latency for each component is as follows:

- **MEC Processing** ($\approx$ 250 ms): Local face detection, cropping, and metadata preparation.
- **Network Transmission** ($\approx$ 200 ms): Time spent on outbound HTTP requests and responses.
- **Telegram Backend** ($\approx 2.5 - 4.5$ s): Server-side processing and push notification delivery.

E. System Resource Monitoring

To evaluate the real-time performance of the proposed surveillance system, MEC (Multi-access Edge Computing) resource monitoring was conducted during operation. Figure 8 presents CPU and RAM utilization, FPS with detection latency, and alert latency over time. The results indicate that CPU usage remains relatively high, between 70–80%, with occasional dips corresponding to lighter workloads or frames containing fewer objects. In contrast, RAM usage remains stable at approximately 28–30%, suggesting that memory is not a limiting factor. FPS fluctuates between 10–13, directly influencing inference latency, which varies from 80–200 ms depending on frame complexity. Alert latency is observed in the range of 2.9–3.5 seconds, consistent with earlier findings that the primary source of delay is external communication with the Telegram API rather than local processing.

V. Limitations and Challenges

While the proposed surveillance system demonstrates promising results in integrating face recognition, intrusion detection, and loitering monitoring, several limitations were identified during experimentation. These challenges underscore areas that require further refinement for deployment in real-world, large-scale environments.

A. Model Misclassification Issues

A key challenge is the occasional misclassification of individuals. These errors primarily result from similarities in facial features, limited training data, and suboptimal recognition thresholds. Misclassifications can reduce system reliability and trigger unnecessary alerts, potentially undermining trust in automated surveillance.

B. Effect of Lighting, Occlusion, and Camera Angle

The accuracy of both face recognition and pose-based loitering detection is highly sensitive to environmental conditions. Poor lighting, strong shadows, or glare can compromise detection reliability. Partial occlusions—such as masks, hats, or objects obstructing the person—further degrade recognition performance. Additionally, camera placement and viewing angle significantly influence detection accuracy, as extreme angles can distort facial features and pose keypoints, increasing false positives or missed detections.

C. Zone Drawing and Resolution Mismatch

A practical limitation was observed during polygonal zone definition for intrusion detection. The RTSP stream is captured at high resolution, making interactive zone drawing challenging. To simplify this process, snapshots were downscaled during zone definition. However, this led to a mismatch: the coordinates drawn on the downscaled snapshot did not accurately align with the original high-resolution video stream, resulting in imprecise intrusion detection boundaries.

VI. Conclusion

This work demonstrates the feasibility of a real-time smart surveillance system integrating face recognition, intrusion detection, and loitering monitoring. Deep learning models automatically identify individuals, detect unauthorized entry,

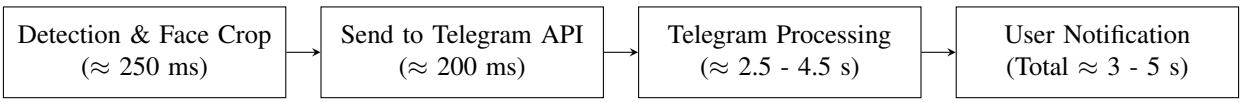


Fig. 7: Timeline of alert latency showing variability between 3–5 seconds.

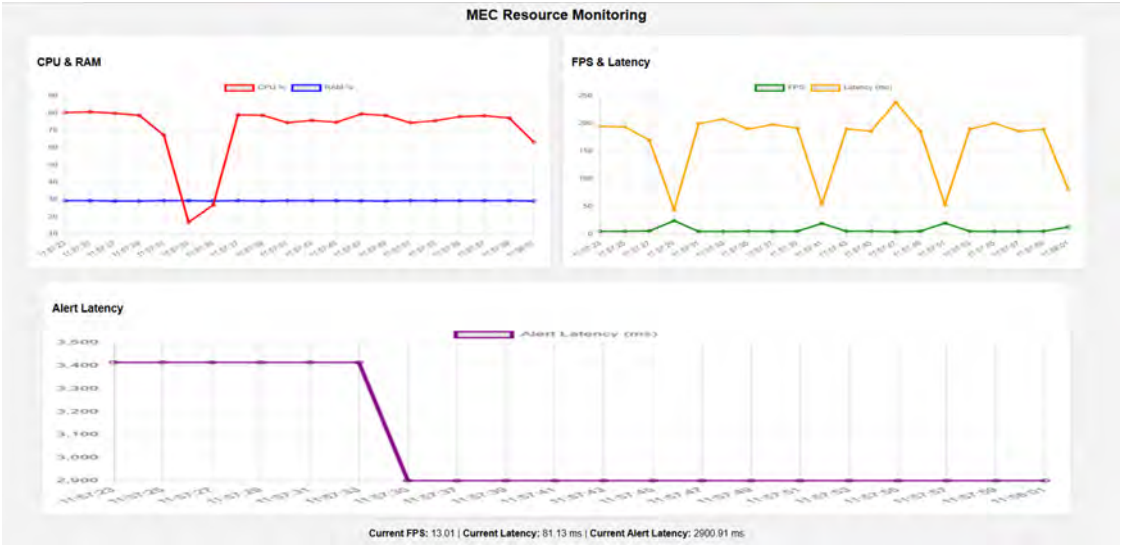


Fig. 8: MEC Resource Monitoring showing CPU/RAM utilization, FPS with detection latency, and alert latency in real time.

and flag unusual behaviors such as prolonged presence in restricted areas. The system is designed for flexibility, allowing users to enable or disable features like zone monitoring, loitering detection, or intrusion alerts via a Flask dashboard, making it adaptable to diverse security requirements.

Testing showed that detection and recognition occur within 200–300 ms, while end-to-end alert delivery takes 3–5 seconds due to network and Telegram server delays. Although the current results sufficient for most security applications, we plan to include faster hardware (such as GPU support) and optimized alert decisions to further improve the responsiveness of the detection in real-time. We also plan to include different complex scenarios, such as low-light and occlusion, while integrating federated learning to improve the scalability and preserve user-privacy.

Acknowledgment

The work is supported by the Indian Institute of Technology Jammu (IIT Jammu) and Department of Telecommunications (DoT), Govt. of India, under the 100 5G Use-Case Lab at IIT Jammu, India.

References

[1] J. Navarro-Ortiz, P. Romero-Diaz, S. Sendra, P. Ameigeiras, J. J. Ramos-Munoz, and J. M. Lopez-Soler, "A Survey on 5G Usage Scenarios and Traffic Models," *IEEE Commun. Surveys Tuts.*, vol. 22, no. 2, pp. 905–929, 2020.

[2] L. Yala, P. A. Frangoudis, and A. Ksentini, "Latency and availability driven VNF placement in a MEC-NFV environment," in *IEEE GLOBECOM*, 2018, pp. 1–7.

[3] S. Bera, "Availability-Aware VNF Placement for uRLLC Applications in MEC-Enabled 5G Networks," in *IEEE ANTS*, Dec. 2023, pp. 300–305.

[4] N. H. Motlagh, M. Bagaa, and T. Taleb, "UAV-Based IoT Platform: A Crowd Surveillance Use Case," *IEEE Communications Magazine*, vol. 55, no. 2, pp. 128–134, Feb. 2017.

[5] S. P J, S. Saba, S. C, and G. M, "Object Detection for Video Surveillance Using Edge-Cloud Collaboration," in *International Conference on Circuits, Power and Intelligent Systems (CCPIS)*, Sep. 2023, pp. 1–6.

[6] "Video Surveillance in a 5G-Enabled IoT World | Sierra Wireless," https://info.sierrawireless.com/white-paper-5g-video-surveillance.

[7] I. Ahmed, M. Ahmad, J. J. P. C. Rodrigues, and G. Jeon, "Edge computing-based person detection system for top view surveillance: Using CenterNet with transfer learning," *Applied Soft Computing*, vol. 107, p. 107489, Aug. 2021.

[8] Z. Wei, W. Jiang, Z. Feng, H. Wu, N. Zhang, K. Han, R. Xu, and P. Zhang, "Integrated Sensing and Communication Enabled Multiple Base Stations Cooperative Sensing Towards 6G," *IEEE Network*, vol. 38, no. 4, pp. 207–215, Jul. 2024.

[9] C. Yang, P. Liang, L. Fu, G. Cui, F. Huang, F. Teng, and Y. A. Bangash, "Using 5G in smart cities: A systematic mapping study," *Intelligent Systems with Applications*, vol. 14, p. 200065, May 2022.

[10] Z. Yuan, T. Azzino, Y. Hao, Y. Lyu, H. Pei, A. Boldini, M. Mezzavilla, M. Beheshti, M. Porfiri, T. E. Hudson, W. Seiple, Y. Fang, S. Rangan, Y. Wang, and J.-R. Rizzo, "Network-Aware 5G Edge Computing for Object Detection: Augmenting Wearables to "See" More, Farther and Faster," *IEEE Access*, vol. 10, pp. 29 612–29 632, 2022.

[11] C. Dong, Y. Liao, Z. Jia, Q. Wu, and L. Zhang, "Joint ADS-B in B5G for Hierarchical AAV Networks: Performance Analysis and MEC-Based Optimization," *IEEE Internet of Things Journal*, vol. 12, no. 12, pp. 22 211–22 223, Jun. 2025.

[12] S. Malik and S. Bera, "Security-as-a-Function in 5G network: Implementation and Performance Evaluation," in *Prof. of SPCOM*, 2024, pp. 1–5.

[13] "Open Source Deep Face Analysis Library - 2D&3D | InsightFace," https://www.insightface.ai/.

[14] Ultralytics, "Pose," https://docs.ultralytics.com/tasks/pose.