International Journal on Advanced Computer Theory and Engineering
ISSN: 2319 - 2526
Volume 15 Issue 2s (2026) | Pages: 63-68

ASSAA: AI-Based Smart Surveillance System for Analysis and Alerting

Pranav Nalawade[1], Pranav Patil[2], Mantesh Swami[3], Nikhil Hinge[4], Pranjali Kharate[5]
[1,2,3,4,5] Department of Artificial Intelligence and Data Science Engineering G.S.Moze College of Engineering Pune, India

Peer Review Information

Type: *Article*
Received: *23 February 2026*
Revised: *22 March 2026*
Accepted: *21 April 2026*
Published: *19 May 2026*

Abstract

Conventional surveillance systems have been highly dependent on human intervention. They are thus prone to errors due to human factors such as fatigue, lack of concentration, and delayed reaction. ASSAA is an AI-driven intelligent surveillance system aimed at automating the real-time surveillance process. This is achieved through automated detection, recognition, and alerting modules. The ASSAA system analyzes real-time video streams by implementing a modular approach. It involves motion detection, person detection using YOLOv8, weapon detection, and facial recognition using dlib128-dimensional embedding vectors. Per- sons detected are checked against a watchlist that contains criminal history and citizenship information. The proposed solution provides a multi-level scoring model for threat assessment. The algorithm incorporates the level of confidence in matching identities, presence of weapons, number of past convictions, and recency of recorded events. The output is a normalized threat score ranging from zero to one hundred. Threats are categorized into four levels, namely low, medium, high, and critical, depending on their risk scores. The system then generates alerts based on the severity of the detected threats. The performance of the ASSAA system has been val- idated experimentally using a simulated webcam environment. The current version of the ASSAA system exists as a command-line prototype.

Keywords: Surveillance System; Computer Vision; YOLOv8; Face Recognition; Threat Detection; Risk Scoring

How to Cite This Article

Nalawade, P., Patil, P., Swami, M., Hinge, N., Kharate, P., (2026). ASSAA: AI-Based Smart Surveillance System for Analysis and Alerting. *International Journal on Advanced Computer Theory and Engineering*, 15(2s), 63-68.

Introduction

Walk into almost any bank branch, university campus, or metro station today and you will find CCTV cameras mounted at regular intervals. The infrastructure has expanded enormously over the past decade, yet the monitoring model behind it has barely changed: human operators watch feeds, recognize anomalies by eye, and call for a response. At a desk managing two or three screens this is workable; across a real deploy- ment with dozens of simultaneous streams it is not. Attention drifts, reactions slow, and events that demand an immediate response can pass un- noticed for critical minutes [1].

The computer vision research community has made substantial progress on the individual pieces of this problem. Object detectors such as YOLO [3] can now flag people and weapons in milliseconds per frame. Multi-object track- ers like DeepSORT [4] maintain identities across frames in busy scenes. Face recognition libraries built on deep embeddings match faces against ref- erence sets with high accuracy under controlled conditions. What is less common is a system that takes all of these outputs and does something use- ful with them—deciding, automatically and with- out an operator, whether the current scene war- rants an alert and what priority that alert should carry.

That gap is what motivated ASSAA. The system is not attempting to replace human judgment entirely; it is attempting to do the triage work that buries operators under low-stakes events and leaves them under-responsive to genuine threats.

- A multi-factor risk scoring engine that com- bines six surveillance-relevant signals into a single normalized threat score.
- A four-tier alert hierarchy that maps score ranges to concrete response actions, from logging only up to immediate law-enforcement notification.

Literature review

Research on automated surveillance spans several areas. We focus on the lines of work most directly relevant to ASSAA: behavioral and bio- metric analysis, video anomaly detection, object detection and tracking, and integrated multi- modal frameworks.

Behavioral and Biometric Analysis

Mamatha et al. [1] demonstrated that combining a CNN for spatial feature extraction with an LSTM for temporal modeling can simulta neously handle activity recognition and identity verification in CCTV footage. The joint spatio temporal approach is well-motivated: many security-relevant behaviors are sequential rather than instantaneous, and a frame-by-frame detector would miss them. The practical difficulty is that the combined model carries significant train- ing cost and does not transfer well across camera installations with different fields of view or lighting profiles—a real constraint for systems that need to be deployed widely.

Anomaly Detection

Elmetwally et al. [2] used an I3D-ResNet50 backbone with a Multiple Instance Learning (MIL) training objective for video anomaly detection, reporting strong performance on benchmark datasets. The dual spatial-temporal representation handles complex abnormal events effectively. However, MIL requires large volumes of weakly labelled video, which is difficult to curate for a specific deployment context like a campus or hospital. Muna et al. [5] approached the same problem with Vision Transformers and temporal attention, achieving improved accuracy on stan- dard benchmarks at the cost of inference latency that makes real-time single-machine deployment challenging without dedicated GPU hardware.

Object Detection and Tracking

Redmon et al. [3] reframed detection as a single- stage regression task, enabling frame rates that made real-time video analysis practical for the first time. Later YOLO iterations continued im- proving the accuracy-speed tradeoff; Ultralytics YOLOv8 [6] introduced an anchor-free decoupled head that meaningfully improves recall for small and partially occluded objects—relevant to detecting weapons held at an angle or persons at the edge of a frame. For tracking, Wojke et al. [4] extended the SORT algorithm with deep appearance features to reduce identity switches in crowded scenes. Both detection and tracking systems share a fundamental scope limitation: they characterize what is present in a frame but say nothing about how concerning it is.

Integrated Frameworks and the Remaining Gap

Recent work has combined detection, tracking, and recognition into more complete pipelines, improving situational coverage. Even so, most of these systems terminate at raw output—a list of detected identities or object classes. A security operator still has to look at that list and decide whether any entry warrants escalation. This recreates, at a later stage, exactly the human bottleneck the automation was meant to reduce. To our knowledge, no published system fuses face identity with criminal history records, le- gal status, and weapon co-detection to produce a graded numeric threat score. That combination is the direct motivation for ASSAA's scoring

Problem Statement

Modern surveillance deployments face a set of compounding problems that go beyond simple detection performance. We identify five

concrete issues that shaped the design of ASSAA. Fragmented pipelines. Detection, recognition, and anomaly analysis are typically implemented as separate tools without shared state. Person detection results are not forwarded to the face recognizer in any structured way, and weapon detections are not cross-referenced against who is carrying them. The system can tell you what is in the frame; it cannot tell you who it is or what the combination implies. Binary outputs. Most systems report a detection or not. A known offender on active bail carrying a firearm and an unidentified pestrian both produce the same output "person detected." The difference in urgency is completely lost.

No historical context. Even accurate face recognition is rarely connected to structured records. Without legal status, prior conviction count, or recency of last incident, a positive identity match contributes little to threat assessment beyond naming the person. Human-in-the-loop response decisions. Many pipelines still require an operator to review detections and decide on a response. Across dozens of concurrent streams, this is not a scalable model. Operator fatigue is not an edge case—it is the normal operating condition, and it directly raises the probability of a critical event going unnoticed. Undifferentiated alerting. Systems that do generate alerts often apply a single severity level to all events. Under continuous feeds this quickly desensitizes operators, and genuinely urgent events compete for attention with routine motion triggers. These problems compound as deployment scale grows. The need is not better detection alone but a layer that translates detections into ranked, actionable decisions without requiring a human to do the ranking.

Proposed System

ASSAA is structured as a sequential, modular pipeline. Each stage gates the next: frames that fail the motion check never reach the detection models, and only frames containing a detected person are forwarded to face recognition. This chained filtering keeps per-frame compute man- ageable on CPU-only hardware.

The pipeline has five functional layers:

1. Input Acquisition. Frames are captured from a webcam or IP camera at the native device frame rate. No resolution scaling or preprocessing is applied at this stage.

2. Motion Screening. MOG2 background sub- traction is applied to each frame. Those whose largest foreground contour falls be- low a configurable area threshold are dis- carded. This is the most important computational shortcut in the system—on a fixed indoor camera, the majority of frames are static, so this single check eliminates most model invocations entirely.

3. Detection. On motion-positive frames, YOLOv8 person detection and two complementary weapon detection models run in parallel. Weapon detections are accepted only when they spatially overlap with a confirmed person bounding box, which pre- vents isolated object detections from influencing the risk score.

4. Recognition. For each confirmed person bounding box, the face region is cropped and a 128-dimensional embedding is extracted using a ResNet-based dlib model. The embedding is compared against precomputed watchlist embeddings by Euclidean distance. A match is declared when the distance falls below threshold $\tau = 0.55$; individuals with no match are carried through the pipeline as Unknown.

5.Risk Assessment and Alerting. All signals from upstream stages—identity, legal status, crime history, weapon presence, conviction count, incident recency—feed into the risk scoring engine, which computes a normalized score $R \in [0, 100]$. The score maps to one of four threat tiers; at medium severity and above, a structured alert record is written to SQLite and printed to console.

Keeping these five layers loosely coupled means individual components can be upgraded independently. For example, the dlib recognizer could be swapped for a quantized ArcFace model for an edge deployment, or the rule-based scoring engine could be replaced by a trained classifier, without touching the detection or alerting stages.

Methodology

Motion Detection

Background subtraction uses the MOG2 Gaussian Mixture Model [7] applied frame-by-frame. Morphological erosion followed by dilation sup- presses small noise artifacts and consolidates foreground regions into coherent blobs. If the largest detected contour falls below a config urable pixel-area threshold, the frame is dis- carded before any model inference takes place.

Person Detection

Person detection uses a pretrained YOLOv8 model [6]. The anchor-free decoupled head architecture in YOLOv8 gives it better recall on smaller and partially occluded targets than earlier YOLO variants, which matters for real-world camera angles where subjects are not always perfectly framed. Only detections in the person class with confidence above 0.5 are retained.

Weapon Detection

Weapon detection combines two YOLO models: a custom model fine-tuned on a firearm dataset from Roboflow [11], and a COCO-pretrained model used for its knife class. Predictions from both are merged. To reduce false positives from visually similar objects such as

umbrellas or folded bags, weapon detections are validated only when they overlap spatially with a con- firmed person bounding box.

Face Detection and Recognition

Face detection within confirmed person regions uses the HOG-based detector from the face recognition library [8], which wraps dlib's implementation. For each detected face, a 128- dimensional embedding is extracted using dlib's ResNet-based model. Watchlist embeddings are precomputed offline and cached in memory at startup. At runtime, Euclidean distance is computed between the query embedding and each stored embedding; a distance below $\tau = 0.55$ constitutes a match.

Risk Scoring Engine

The core contribution of ASSAA is the risk scoring engine. For each detected individual, a raw score R is computed as the sum of six components is a strong situational signal; and recency, convictions, and prior victims are secondary contextual factors. We acknowledge that these weights were set by judgment rather than learned from labelled incident data, and refining them empirically is part of the planned future work.

Alert Generation

Each alert record stores: detected identity (or)

R = Sstatus+Scrime+Sweapon+Srecency+Sconv+SvictimsUnknown), face match confidence, computed

Table 1 defines each component and its maxi- mum contribution. The weights reflect the intuition that legal status and crime severity are the strongest indicators of threat; weapon presence risk score, assigned threat tier, detected weapon classes, and a UTC timestamp. Records are writ- ten to a SQLite database. In future versions, alerts will also be pushed to a web dashboard and a mobile notification service.

Table 1. Threat Tier Classification and Response Actions

Score Range	Tier	Response Action
0–24	Low	Log entry only
25–49	Medium	Alert written to SQLite; console output
50–74	High	Alert with priority flag
75–100	Critical	Immediate law- enforcement notification

System Architecture

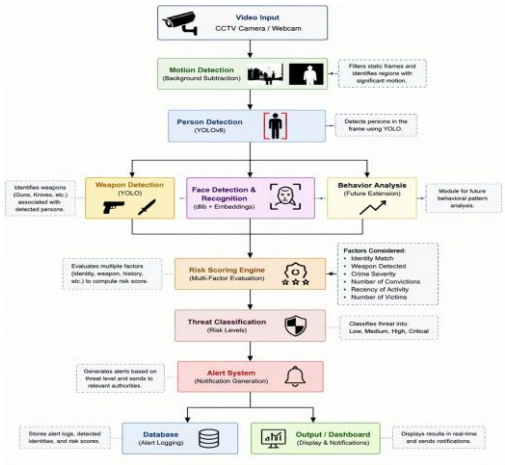

Fig 1. *End-to-end architecture of ASSAA.*

Figure 1 illustrates the end-to-end architecture. Three design decisions are worth explaining ex- plicitly. Motion gating at the front. Placing motion detection first means the two most expensive operations in the pipeline—YOLOv8 inference and face embedding extraction—only fire when something is actually moving in the scene. On a fixed indoor camera this alone reduces active inference frames to a small fraction of total cap- tured frames, which is what makes real-time operation feasible on CPU hardware. Parallel detection with spatial linking. Per- son detection and weapon detection run concurrently on the same motion-positive frame. The bounding-box overlap check is the architectural join between the two streams: a weapon detec- tion only counts if it falls within or significantly overlaps a confirmed person region. This prevents a weapon lying on a table, for instance, from generating a risk score on its own. Two-stage identity resolution. Face recogni- tion is scoped to confirmed person regions rather than applied to the full frame. Running the HOG detector and embedding extraction on a cropped bounding box rather than on a 1080p frame re- duces compute and, importantly, reduces spuri- ous detections from background surfaces that can superficially resemble faces.

Implementation

Development Environment

The prototype is written in Python 3.10. Depen- dencies are: OpenCV 4.9 [7] for frame capture and preprocessing; Ultralytics YOLOv8 [6] for detection; the face recognition library [8] wrap- ping dlib for face embedding; NumPy for dis- tance arithmetic; and SQLite3 for alert persis- tence. All development and testing ran on an Intel Core i5 machine with 16 GB RAM and no GPU. The CPU-only constraint was intentional— it gives a conservative lower bound on what the system can sustain in deployment.

Watchlist Dataset

We built a simulated watchlist of 20 individu- als, each entry carrying: name, crime type, le- gal status, prior conviction count, victim count, and date of last incident. These fields corre- spond directly to the six components in Equa- tion 1. Face images for enrollment came from the Kaggle Criminal Detection Dataset [9] and the Roboflow Criminal Faces Dataset [10]. The firearm detection model was fine-tuned on a Roboflow firearms dataset [11]; knife detection reuses COCO-pretrained weights without addi- tional fine-tuning.

Pipeline Execution Loop

At startup, all watchlist face images are pro- cessed through the embedding model and the resulting vectors are cached in memory, avoiding repeated disk reads during the main loop. Each frame then proceeds through the following steps:

1. Capture frame from webcam via cv2.VideoCapture.

2. Apply MOG2 background subtraction; dis- card if maximum contour area is below threshold.

3. Run YOLOv8 person detection and both weapon models on motion-positive frames.

4. For each person bounding box, run HOG face detection and compute a 128-d embed- ding; compare against cached watchlist vec- tors using Euclidean distance.

5. Pass all signals to the risk scoring engine; compute R per individual per Equation 1.

6. Write alert records at medium severity and above to SQLite; print structured output to stdout.

Performance Optimizations

Three practical adjustments were made during testing. First, MOG2 gating (step 2 above) is the single biggest contributor to keeping the system real-time—most frames on a fixed camera are static and are dropped cheaply before any neural inference runs. Second, restricting face recog- nition to person bounding boxes rather than the full frame meaningfully reduces HOG detector load. Third, the weapon detection confidence threshold was raised from the default to 0.65 af- ter we observed consistent false positives from umbrellas and long-handled tools; this reduced those spurious detections without causing any missed true detections in our test set.

Deployment Pathway

The current prototype handles one camera stream in a single process. Scaling to multiple streams is straightforward: each stream can run in an inde- pendent process writing alerts to a shared SQLite database or a message queue. For resource- constrained hardware, the dlib embedding model is the primary candidate for replacement—a quantized MobileNet-based model would cut memory footprint substantially. Edge deploy- ment on NVIDIA Jetson hardware would addi- tionally enable GPU-accelerated YOLOv8 infer- ence, reducing per-frame latency considerably below what CPU-only hardware achieves.

Results and Discussion

We tested the prototype in a controlled indoor en- vironment using a standard 1080p webcam under typical artificial lighting. Table 3 summarizes observations across each pipeline stage. Person and weapon detection. YOLOv8 person detection was consistent up to roughly 3 m. Beyond that, bounding box confidence dropped below the 0.5 threshold and subjects were missed. Weapon detection at the default

Table 2. Module Performance Summary (Con- trolled Indoor Environment)

Module	Observed Behaviour	Known Limitation
Motion Detection	Effective noise filtering under stable lighting	Reduced accuracy with variable frame rates or network-streamed feeds subject to compression artifacts
Person Detection	Reliable at distances up to 3 m	Performance decreases in low-light or partially occluded environments
Weapon Detection	Functional; occasional false positives	Sensitive to object orientation and image quality
Face	Accurate for frontal, unoccluded faces	Reduced performance for side profiles and masked faces

Recognition	within 2 m	
Risk Scoring	Correct tier assigned in all test scenarios	Two boundary cases illustrate the need for further calibration in complex scenarios

Conclusion

This paper presented ASSAA, a modular surveil- lance pipeline that pulls together motion screen- ing, YOLOv8-based detection, dlib face recog confidence threshold produced noticeable false positives from umbrella handles and similar elon- gated objects; raising the threshold to 0.65 re- solved most of these. We did not lose any true weapon detection during testing at this higher threshold, though we acknowledge the test set was small. Face recognition. Frontal, unoccluded faces within approximately 2 m matched correctly against watchlist embeddings in our tests. Per- formance degraded predictably with increasing distance and partial occlusion. One specific issue worth noting: at frame rates below around 15 fps, motion blur on walking subjects degraded embed- ding quality enough to cause false non-matches against individuals who matched correctly in still images. This frame-rate sensitivity is probably the most operationally significant limitation we found, since real deployments may use cameras nition, and a rule-based risk scoring engine into a single real-time system. The core idea was to move beyond binary detection outputs—person present, weapon present—toward a graded threat assessment that takes into account who a per- son is, what their history is, and what they are currently doing. We implemented this as a nor- malized score between 0 and 100 that maps to four operational alert tiers. Testing on CPU-only hardware under con- trolled indoor conditions confirmed that the pipeline is feasible at real-time frame rates and that the scoring engine produces sensible tier assignments across a range of constructed sce- narios. We make no stronger claim than that; the evaluation was small-scale and the weights were not fitted from data. Three directions are clear priorities for follow- up work. First, the heuristic scoring weights should be replaced with weights learned from labelled incident records—logistic regression or gradient boosting are natural starting points given the tabular feature structure. Second, integrat- ing DeepSORT-based tracking would address the duplicate alert problem (where the same indi- vidual triggers repeated alerts across consecu- tive frames) and would enable trajectory-level reasoning that single-frame analysis cannot sup- port. Third, the system needs validation on es- tablished public benchmarks—UCF-Crime and ShanghaiTech are the most relevant—which in- clude realistic lighting variation, occlusion, and crowd conditions that our controlled environment did not.

References

1. K. R. Mamatha, P. Anand D, R. Umasankaran, and N. Joshi, “Opti- mized CCTV monitoring using biometrics and AI-driven surveillance,” International Journal of Electrical Engineering and Computer Science, 2025.
2. A. Elmetwally, R. Eldeeb, and S. Elmougy, “Deep learning-based anomaly detection in real-time video,” Multimedia Tools and Ap- plications, 2024.
3. J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You Only Look Once: Uni- fied, real-time object detection,” in Proc. IEEE/CVF Conf. Computer Vision and Pat- tern Recognition (CVPR), pp. 779–788, 2016.
4. N. Wojke, A. Bewley, and D. Paulus, “Sim- ple online and realtime tracking with a deep association metric,” in Proc. IEEE Int. Conf. Image Processing (ICIP), pp. 3645–3649, 2017.
5. Muna et al., “Transformer-based video anomaly detection using Vision Transform- ers,” 2025.
6. Ultralytics, “YOLOv8: A new state-of- the-art real-time object detection model,” [Online]. Available: https://github. com/ultralytics/ultralytics, 2023.
7. OpenCV Organization, “OpenSource Com- puter Vision Library,” [Online]. Available: https://opencv.org/
8. A. Geitgey, “face recognition: The world’s simplest face recognition library,” [Online]. Available: https://github.com/ ageitgey/face_recognition, 2019.
9. M. Okuyar, “Criminal Detection Dataset,” Kaggle, [online]. Avail- able:https://www.kaggle. com/code/mehmetokuyar/ criminal-dedection, 2024.
10. “Criminal Faces Dataset,” Roboflow Universe, [online]. Available: https://universe.roboflow. com/fishsegmentation-kv89x/ criminal-faces-acxij, 2024.