Smart Video Activity Detection System

Jignesh Chaudhari[1], Shreeyash Pawar[2], Dimpal Phalak[3], Prof. Purva D. Thakare[4]
[1,2,3]*School of Computer Science, Engineering and Applications, D Y Patil International University, Akurdi, Pune, India 411044*
[4]*Project Guide, School of Computer Science, Engineering and Applications, D Y Patil International University, Akurdi, Pune, India 411044*

***Abstract*—An incident occurred in our college in which there was an occurrence of a knife-based threat. This necessitated the development of a Smart Video Activity Detection System that could detect threat situations in real-time. It is based on YOLOv8 Nano model which was trained on a custom dataset containing 4,048 images sourced from various sources like Roboflow, Google Images, Kaggle, and many other different sources. The dataset contains annotated images totaling to 4048 with threat classes/ categories being knife, gun, alcohol, smoking, violence and grenades. We also incorporated YOLOv8 pose estimation to know body posture and the relation of the object with others for better understanding of what activity is going on. The training and testing of the model was done in local PC and got mean Average Precision score of 0.749, precision of 0.817 and recall of 0.734. Our interface is simple and built in Python using Tkinter library supporting real-time video from webcam as well as video file. In case of any detection, a message gets triggered on our email account as well as WhatsApp and a picture is saved with a sound alarm and logging of daily CSV. Our testing confirmed that this system can work effectively in real-world surveillance scenarios like college campuses which will help to prevent a threat.**

***Index Terms*—YOLOv8, YOLOv8 Nano, object detection, pose estimation, real-time surveillance, weapon detection, violence detection, threat detection, activity recognition, deep learning, computer vision, multithreading, alert system, computer vision, Tkinter, WhatsApp alert.**

I. INTRODUCTION

Public safety monitoring and surveillance necessitate the presence of automated surveillance in smart city architecture. Conventional surveillance systems based on CCTV, which involve the use of humans to monitor their feed, encounter numerous challenges because of the human's inability to focus on such tasks due to fatigue and other limitations. The initial attempts at automated surveillance involved background subtraction-based motion detection techniques [1]. It is a widely held view that advancements in the fields of deep learning and CNN have made automated activity detection possible [2], [3].

The current state-of-the-art techniques for monitoring tend to target a specific kind of threat, such as automatic gun recognition in video surveillance [6], [7]. Although real-time object detection algorithms such as YOLO have achieved tremendous progress in terms of one class object detection [4], and techniques like pose estimation have successfully extracted key points of human bodies [8], [9], all of them have been developed independently. What is lacking in today's automatic surveillance system is the capability of detecting a variety of objects and inferring the relationship between them.

This study addresses the problem by providing a pipeline of surveillance system that can detect six kinds of suspicious objects namely weapons such as guns, sharp-edge weaponries, intoxicant liquids, and objects related to tobacco usage. In calculating the risk associated with a particular object, analysis of the position of an object detected against the human body will be carried out. This project utilizes YOLOv8 [5], which is a recent object detection tool among the You Only Look Once series. Detection of the human body pose is performed to extract the human skeleton key points for performing proximity-based analysis [8], for instance, if the gun is detected around the joints of the hand rather than on a flat surface, the risk becomes high [10].

II. LITERATURE REVIEW

A. Traditional Surveillance and Action Recognition

The traditional systems used for video surveillance have relied largely on background subtraction and optical flow algorithms for identifying anomalies in motion [1]. Even though these methods are computationally efficient, they are highly susceptible to illumination variations and camera motion. They do not have the capability for recognizing object-level threats. In practice, there are usually many false alarms generated by such alarms.

Action recognition research has transitioned from feature-based approaches to deep learning approaches. The two-stream convolutional networks [2] analyze the appearance and motion streams independently before merging them. On the other hand, 3D CNNs such as C3D [3] and SlowFast networks analyze time-varying patterns using 3D convolutions. However, the heavy computational cost makes them unsuitable for real-time systems.

B. Object Detection in Surveillance

YOLO class of detectors are popular due to their real-time surveillance ability. It utilizes the one-pass detection algorithm. YOLO was introduced by Redmon et al. [4] in the form of a single unified architecture and was further optimized in YOLOv2, YOLOv3, and YOLOv4. YOLOv5 performed effectively on the COCO dataset but had a small model size. The latest YOLOv8 was proposed by Jocher et al. [5], which consisted of decoupled heads, anchor-free detector, and the C2f module providing smooth gradient flow, making it faster and more accurate than its predecessors.

The weapon detection application has been recently researched. Salido et al. [6] performed real-time firearm detection using CNNs from CCTV images with high accuracy in laboratory conditions. Olmos et al. [7] implemented transfer learning for handgun detection in urban areas. The paper also highlighted challenges related to low resolution and partial occlusion issues. Substance-based object detection of alcoholic beverages and cigarettes has been overlooked in most prior studies.

C. Human Pose Estimation

Human pose estimation is a technique that detects the positions of body joints like shoulders, elbows, wrists, and knees from images or video. This gives a clear picture of how a person is positioned or moving. OpenPose [8] was one of the first systems to handle multi-person pose estimation using Part Affinity Fields. HRNet [9] improved this further by keeping high resolution features at every stage of the network, which helped it achieve strong results on the COCO keypoints benchmark. In our system we used the YOLOv8 pose model developed by Ultralytics, which detects keypoints at the same time as objects in a single forward pass — making it fast and efficient for real-time use.

Pose-aware activity recognition has been explored for gesture recognition, fall detection, and ergonomics monitoring. Context-aware weapon detection using pose cues represents a less explored but highly practical direction. The proximity of a detected firearm to a person's hand keypoints, for instance, dramatically changes the interpretation of the detection and the appropriate system response.

D. Alert and Notification Systems in AI Applications

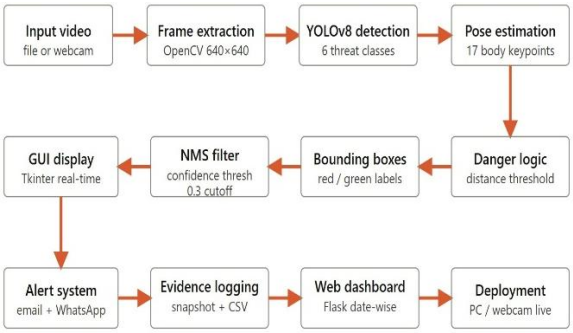
Sending automatic alerts is an important part of any real time surveillance system. Some recent studies have looked at using messaging APIs to deliver instant notifications when a threat is detected [10]. One common issue in continuous monitoring is that too many alerts can overwhelm the operator. One way to deal with this problem is by adding a cooldown timer between alerts, so the operator is not constantly disturbed for the same threat. Saving a snapshot image at the time of detection is also becoming a standard practice as it provides visual evidence for later review.

III. METHODOLOGY

A. System Architecture Overview

In the system architecture, the threat detection happens in real-time using the video feed captured either from a file source or camera feed. Here, the video is broken down into individual frames, processed using object detection models such as YOLOv8 and pose estimation for analysis of the human keypoints present in the frame. Threat analysis is done using danger logic and the findings are marked with colored bounding boxes. Any redundant threats detected will be filtered using non-max suppression algorithm before displaying the result using the GUI interface. Alerts will be generated by sending a notification either by

emails or WhatsApp messages. Evidence of the detected threats will be saved as image snapshots and CSV files while an accessible web-based logging system will be used for organizing the logs.



B. *Dataset Construction*

Since no public dataset had all the needed classes, it was decided to use several data sources such as Google Images, Roboflow Universe, Open Images Dataset, and Kaggle to develop a custom dataset. In total, there were 4,048 images, which are allocated in six classes: Alcohol (1,169 images), Grenade (598 images), Gun (818 images), Knife (874 images), Smoking (849 images), and Violence (681 images). To ensure the best performance of the model on the different types of datasets, various illumination, backgrounds, orientations, scales, and viewpoints were considered.

For image annotation, the Roboflow software was employed. In the initial stage, bounding boxes were manually annotated for each object, with every instance classified into one of six categories: Alcohol, Grenade, Gun, Knife, Smoking, and Violence. The final annotations were saved in YOLO format, and the coordinates were normalized (<x_center> <y_center> <width> <height>). As far as splitting is concerned, all images were automatically split into training (70%), validation (20%), and testing (10%) datasets. All images were resized to 640 × 640 pixels and oriented according to the algorithm.

Table I: Dataset Class Distribution

Class	Images	% of Total
Gun	~818	20.21%
Knife	~874	21.59%
Alcohol	~1169	28.88%
Smoking	~849	20.97%
Grenade	~598	14.77%
Violence	~681	16.82%

C. *Model Selection and Training*

It was decided to use YOLOv8 Nano (yolov8n) as a basis for detecting pedestrians. This model is best suited in terms of trade-off between speed and precision to process video in real time, both on CPU and on GPU. It was initialized with pretrained weights on ImageNet and fine-tuned on the custom dataset for 60 epochs with images of 640 x 640 pixels. The Adam optimizer with cosine annealing of learning rate was used for training. Model training was carried out in Python 3.10 virtual environment using PyTorch and Ultralytics. Best model weights (best.pt) have been chosen according to validation mAP@0.5 metric.

Secondary inference model (YOLOv8 Nano Pose, yolov8n-pose) detects persons' bounding boxes as well as 17 COCO keypoints, including keypoints for wrists (keypoints 9 and 10) and nose (keypoint 0).

D. *Context-Aware Decision Logic*

The detection alone cannot be enough to assess the degree of threat; context must come into play, especially when there is involvement of a human actor. Proximity-based classification is done by finding the distance between the object centroids and anatomically related keypoints using the pose estimation network. The following proximity-based rules have been established for threat classification:

1. Weapon (gun, knife, grenade):

Such objects will be considered as DANGER when their centroids fall within 110 pixels of either wrist keypoints. This means that the object is being used or handled by the person.

2. Substances (alcohol, smoking):

They will be classified as DANGER when their centroid distances are within 150 pixels of the nose keypoint. This indicates that the substance is in use by the individual.

3. Violence class:

When a detection is assigned the violence label, the threat classification is done immediately because it is an inherently dangerous activity.

Fallback method is used in case there are no available keypoints. If in such a case, there is any target object detection above the specified threshold, then the DANGER flag is used.

Objects detected as dangerous threats are highlighted using red bounding boxes, while the rest are shown using green bounding boxes.

E. *Alert System*

The alerts mechanism is initiated when the detection falls under DANGER category. For the email alerts, Gmail SMTP SSL (port 465) protocol along with App password is used for sending emails through the smtplib library of Python. The body of the email message is HTML formatted which consists of the threat type, confidence value, and timestamp with a JPEG attachment of the frame of detection.

For the WhatsApp alerts, UltraMsg API is used and the connection of the user's account on WhatsApp is established using QR code-based authentication method. The interval of time after which alerts will stop due to continued presence of the threat is 30 seconds for each object class. In order to create sound alerts, an audio file called alarm.wav is generated and played using playsound library.

F. *Evidence Logging*

A DANGER incident produces two continuous data logs. Firstly, a JPG image snapshot is created and saved in a folder named snapshots/, using a filename format that includes class name, date, and time (for example, gun_2025-04-03_14-23-01.jpg). Secondly, a CSV file log is created and added to an existing daily report file in the reports/ folder. The CSV file contains the date, time, threat class, model confidence value, image snapshot filename, and camera type used in video analysis. The CSV headers are logged only at the time of file creation.

G. *GUI and Multithreading Architecture*

The GUI was implemented using Python's Tkinter module. In the GUI, options have been provided to choose between browsing files and activating the webcam as a video source, as well as for enabling and disabling the monitor, activating and deactivating the audio alert, removing the alerts logged, and accessing the snapshots folder and CSV log.

There is an alert log display in real-time that shows timestamped log entries for all alerts. The source indication, as well as live timestamp watermarks, are added to the processed frames.

All the intensive tasks, such as processing video frames, inferring models, sending out alerts, saving snapshots, and generating logs in CSV format, are performed by daemon threads using Python's threading library.

IV. RESULTS AND DISCUSSION

A. Model Performance Metrics

The trained YOLOv8 model was evaluated on the held-out validation set using the standard COCO detection metrics. Table II summarizes the overall and class-wise performance.

Table II: Model Evaluation Results

Class	Precision	Recall	mAP@0.5	mAP@0.5:0.95
All Classes	0.72	0.73	0.749	0.461
Gun	0.77	0.784	0.812	0.417
Knife	0.783	0.81	0.857	0.496
Alcohol	0.698	0.735	0.719	0.37
Smoking	0.491	0.494	0.461	0.238
Grenade	0.812	0.922	0.917	0.815
Violence	0.768	0.635	0.731	0.432

A high mAP@0.5 value of 0.749 indicates efficient detection capability in all six categories. The mAP@0.5 score of 0.917 with high recall (0.922) for grenade detection shows that the algorithm is very good at recognizing explosive objects irrespective of their shape or orientation. Knife detection was found to be fairly efficient (mAP@0.5 = 0.857), with a high recall score of 0.810 that reflects efficient recognition of bladed weapons. The gun detection task had achieved fair success (mAP@0.5 = 0.812) with relatively balanced precision and recall scores of 0.770 and 0.784, respectively. The alcohol detection task had performed fairly (mAP@0.5 = 0.719); however, the lower value of precision (0.698) was attributed to

varying sizes, colors, shapes, and transparencies of bottles. The violence detection task had performed decently (mAP@0.5 = 0.731), while lower recall (0.635) suggested difficulty in detecting some forms of violent behavior. Lastly, smoking detection performed poorly (mAP@0.5 = 0.461), since the small physical size of cigarettes made them appear low resolution in images and resulted in poorer precision (0.491).

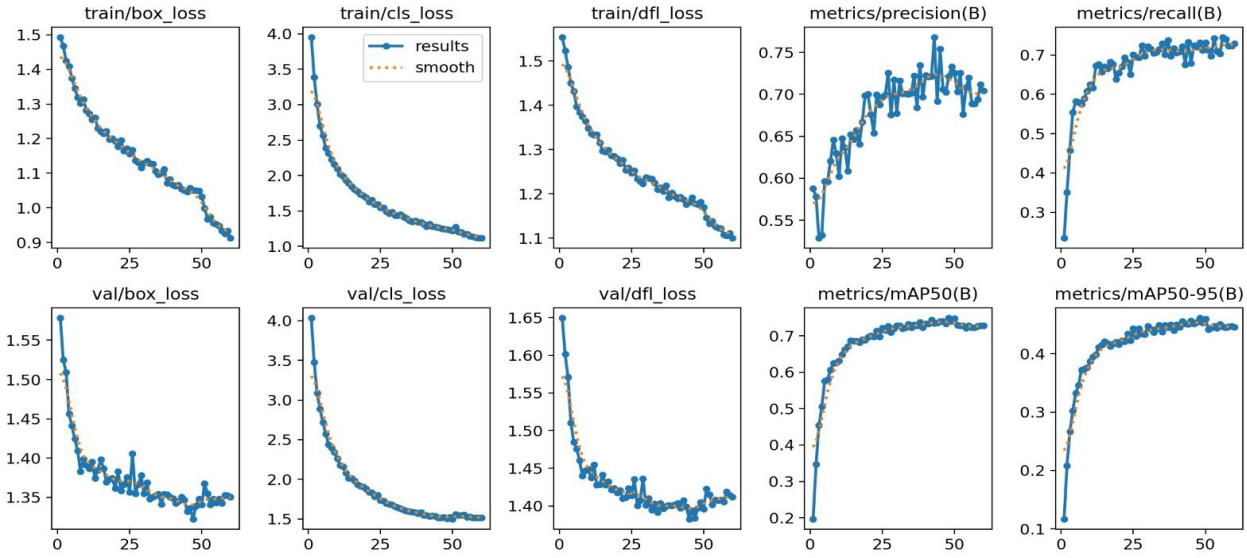


Figure. 1 Training Curves

Figure 1 depicts the learning curves for training and validation phases over 60 epochs, proving that the model converges well in all loss and metric graphs.

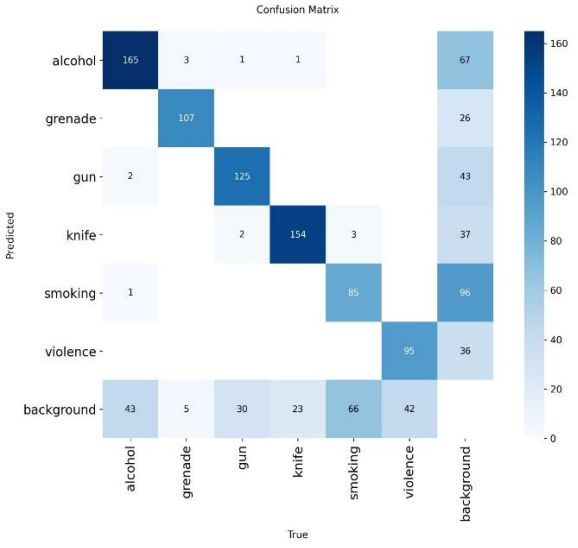


Figure. 2 Confusion Matrix

Figure 2 shows the confusion matrix obtained using the validation dataset to evaluate the performance per class for the proposed model.

B. System Performance Analysis

The proposed framework had real-time capability on consumer-grade devices. Using CPU-based devices only, the object detection speed for YOLOv8 Nano was 15-20 FPS, while pose estimation had additional 5-8 FPS overhead for every frame with person detection. Using NVIDIA-based GPU hardware, the inference speed was more than 60 FPS. The use of multithreading for alerting ensured that both email and WhatsApp alerts had no impact in terms of latency on the video processing workflow.

With a 30-second cooling period, there were no cases of flooding alerts during surveillance mode. In all testing sessions of videos belonging to different categories, all events classified as DANGER caused the email and WhatsApp alerts to be triggered after approximately 2-3 seconds since the detection of the first frame. For the sound alarm, the response was within one video frame.

C. Comparative Analysis

Table III presents a comparison of the proposed system against related approaches in the literature.

Table III: Comparison with Related Systems

System	Model	Classes	mAP@0.5	Alert System
Salido et al. [6]	SSD	1 (gun)	0.812	None
Olmos et al. [7]	Faster R-CNN	1 (gun)	0.846	None
Kumar et al. [10]	CNN + IoT	1 (Violence)	~0.72	SMS
Tran et al. [3]	3D CNN	Action-based	~0.70	None
Redmon et al. [4]	YOLO	Multiple (Objects only)	~0.63	None
Proposed System	YOLOv8 + Pose	6(Objects + Activity)	0.749	Email + WhatsApp + Sound

Unlike other systems in the list, which only consider one or two threats in their classification process, the proposed system tackles all six threats using contextual pose reasoning and an automatic multi-channel alerting system. Although the proposed system may perform at a similar mean average precision to systems that deal with one or two classes of threats, it should be noted that the proposed system deals with a much more challenging problem.

D. Limitations and Challenges

Some limitations have been observed during development and testing. Cigarette detection accuracy was consistently not reaching target threshold values because of the small size of the objects involved at the distance of surveillance cameras. The keypoint extraction step of the pose model does not work on people that are partially occluded, thus, the fallback classification approach had to be used. It is advisable to use GPU acceleration for this application to be able to process multiple streams simultaneously.

V. CONCLUSION

The present paper provided information on the development of the Smart Video Activity Detection System as a complete real-time surveillance system that uses YOLOv8 object detection as well as human pose estimation. The project was created, implemented, and tested throughout twelve stages, including problem definition, environmental setup, creation of the dataset, training of the models, evaluations, GUI development, inclusion of the alerting system, tests, feature improvements, and system consolidation.

The trained YOLOv8 Nano detector obtained an overall score of mAP@0.5 at 0.749 for six surveillance-related threat categories; furthermore, a remarkable performance was observed in weapon detection (Grenade: 0.917, Knife: 0.857). The use of pose estimation makes the identification of potential threats more intelligent than based solely on their presence, while the multi-channel alert system brings practical benefits.

This project shows that building an effective and deployable surveillance system based on artificial intelligence is feasible using open-source tools, consumer-grade hardware, publicly available pre-trained models, and our own data. In future research, we aim at (1) expanding the number of classes for recognition by adding new threat categories such as syringes and fire; (2) developing spatio-temporal models for action recognition; (3) running our solution on edge computing devices like NVIDIA Jetson and Raspberry Pi for the purpose of surveillance; (4) applying federated learning to improve the generalization ability of the model across cameras located in different geographical areas; and (5) publishing our dataset for benchmarking purposes.

REFERENCES

[1] C. Stauffer and W. E. L. Grimson, “Adaptive background mixture models for real-time tracking,” in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 1999, pp. 246–252.

[2] K. Simonyan and A. Zisserman, “Two-stream convolutional networks for action recognition in videos,” in *Adv. Neural Inf. Process. Syst. (NeurIPS)*, 2014, pp. 568–576.

[3] Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri, "Learning spatiotemporal features with 3D convolutional networks," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, 2015, pp. 4489–4497.

[4] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2016, pp. 779–788.

[5] Jocher, A. Chaurasia, and J. Qiu, "Ultralytics YOLOv8," [Software], 2023. Ultralytics YOLOv8 GitHub

[6] J. Salido, J. Lomas, J. Ruiz-Santaquiteria, and O. Deniz, "Automatic handgun detection with deep learning in video surveillance images," *Appl. Sci.*, vol. 11, no. 13, p. 6151, 2021.

[7] R. Olmos, S. Tabik, and F. Herrera, "Automatic handgun detection alarm in videos using deep learning," *Neurocomputing*, vol. 275, pp. 66–72, 2018.

[8] Z. Cao, T. Simon, S.-E. Wei, and Y. Sheikh, "Realtime multi-person 2D pose estimation using part affinity fields," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2017, pp. 1302–1310.

[9] K. Sun, B. Xiao, D. Liu, and J. Wang, "Deep high-resolution representation learning for visual recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 43, no. 10, pp. 3349–3364, 2021.

[10] Kumar and S. Sharma, "Real-time violence detection and alert system using deep learning and IoT," *Int. J.*, 2021.

[11] Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "YOLOv4: Optimal speed and accuracy of object detection," *arXiv preprint* arXiv:2004.10934, 2020.

[12] C.-Y. Wang, A. Bochkovskiy, and H.-Y. M. Liao, "YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2023, pp. 7464–7475.

[13] J. Carreira and A. Zisserman, "Quo Vadis, action recognition? A new model and the Kinetics dataset," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2017, pp. 6299–6308.

[14] W. Kay *et al.*, "The Kinetics human action video dataset," *arXiv preprint* arXiv:1705.06950, 2017.

[15] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 6, pp. 1137–1149, 2017.

[16] T.-Y. Lin *et al.*, "Microsoft COCO: Common objects in context," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2014, pp. 740–755.

[17] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2016, pp. 770–778.

[18] Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Adv. Neural Inf. Process. Syst. (NeurIPS)*, 2012, pp. 1097–1105.

[19] M. Everingham *et al.*, "The Pascal Visual Object Classes (VOC) challenge," *Int. J. Comput. Vis.*, vol. 88, no. 2, pp. 303–338, 2010.

[20] Xiao, H. Wu, and Y. Wei, "Simple baselines for human pose estimation and tracking," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 466–481.