AI-Powered Incident Detection for Smart City Surveillance Using Computer Vision and Automated Emergency Response

Bhavya Kaushik[1], Archit Kumar[2], Shreyash Bhardwaj[3], Anju Saini[4]
Department of Computer Science & Engineering (AI & DS), Panipat Institute of Engineering and Technology, Panipat, India [1,2,3,4]
bhavyakaushik04@gmail.com[1], architkkr@gmail.com[2], shreyashbhardwaj17@gmail.com[3], anju.cse.ai-ds@piet.co.in[4]

Abstract. *Because of rapid urbanization, today the surveillance systems have become inefficient for real time emergency response. This paper presents an AI-based smart city incident detection system which processes images or videos and helps in detection of incident including the severity of the incident and nearby hospitals or police stations and immediate actions that could be taken. The system has a YOLOv8n-based detection model, rule-based severity classification, OpenStreetMap-based geolocation, PDF report generation, and automated email alerts. The system is evaluated on 9,355 samples across six incident categories, in which the system achieves the system achieves 92.1% accuracy with an average latency of 1.8 seconds. The results demonstrate its effectiveness for real-time deployment in smart city environments.*

Keywords: Smart City; Incident Detection; Computer Vision; YOLOv8; Severity Classification; Emergency Response; Geolocation; FastAPI; OpenStreetMap; Automated Alerting.

Introduction

There has been an increase in urban civilization which has outpaced the ability of traditional monitoring system to provide a timely solution. Road accidents, building fires, and other disturbances now occur frequently and is observed by control rooms govern by humans. Even a two-minute delay can worsen the conditions of a severe incident and may lead to life-threating situations. Governments are investing in smart city infrastructures like CCTV cameras and other devices but still there is a gap in real time response. The human operator watches it on screen, calls and inform dispatch centers and records every detail by hand which adds latency and a room for error and delay in severe situations.

Recent advances in the field of deep learning, especially object detection models like yolo, enables real-time object identification which help in incident identification. Along with yolo, GPU the system is a model with geographic lookup, automated document generation and alerts send to nearby facilities. This leads to a better pipeline solution where an image or video is uploaded, the incident is classified, severity is defined, nearest hospitals facilities are located, a formal pdf report is generated and an email alert is sent.

The system supports six categories of incidents which includes: fire, road accident, flood, public disturbance, road damage, and illegal dumping. A FastAPI backend that ties YOLOv8, an

OpenStreetMap, a ReportLab and an SMTP dispatcher. A single HTML file which has the frontend and handles the GPS coordinates of the incident and gives results.

II Literature Review

The early surveillance system was dependent on background subtraction and optical-flow analysis. Those approaches handled only static and well-defined images but didn't perform on crowded, rain-affected or dynamic urban environment images. But then Deep Convolutional Architecture changed the whole perspective and helped in object identification and frameworks like YOLO and R-CNN reduced the gap between accuracy and throughput. Within the smart city domain,[4] Sabokrou et al., obtained 87% accuracy on crowd disturbance from CCTV footage with a CNN-based model but found that value was limited by the absence of any downstream dispatch mechanism. Emergency Response Systems using IoT and AI, IEEE Access built a rule-based model that reliably classified the incident but it only accepted text input and not supported image input.[6] Haklay & Weber (2008) → OpenStreetMap quality evaluation has been validated as a reliable geospatial data source in urban environments. [5]Sultani et al., Real-World Anomaly Detection in Surveillance Videos, CVPR 2018 combines multiple signals and has multi-factor severity scoring outperforms thresholding. Alam et al., IoT-Based Smart Emergency Response System, IEEE has demonstrated that automated email and push notification pipelines which reduces the response awareness time from minutes to seconds. But no prior system combines image-based classification, live GPS lookups, nearest-facility routing, automated PDF reporting and email alerting in a single pipeline. The proposed system closed the mentioned gap.

Table 1: Summary of Related Work

Sr. no.	Year	Papers	Focus	Key Finding	Limitation
[1]	2012	Krizhevsky et al. – ImageNet Classification with Deep CNNs	Deep Learning (CNNs)	Demonstrated CNNs outperform handcrafted features in large-scale image classification	Requires large labeled datasets and high compute
[2]	2018	Wong et al. – YOLO-LITE: A Real-Time Object Detection Algorithm Optimized for Non-GPU Computers	Embedded Vision	Achieved real-time inference on low-resource devices	Reduced accuracy compared to full YOLO
[3]	2016	Hasan et al. – Learning Temporal Regularity in Video Sequences	Video Anomaly Detection	Used deep models to detect anomalies in surveillance videos	Limited contextual understanding
[4]	2017	Sabokrou et al. – Deep-Anomaly Detection for Crowded Scenes	Crowd Surveillance	Efficient anomaly detection in crowded environments	Sensitive to noise and lighting variations
[5]	2018	Sultani et al. – Real-World Anomaly Detection in Surveillance Videos	Severity Scoring / Anomaly Ranking	Demonstrated multi-instance learning improves anomaly detection accuracy	Requires large annotated datasets

[6]	2008	Haklay & Weber – OpenStreetMap: User-Generated Street Maps	Geospatial Data (OSM)	Validated OSM as a reliable, low-cost mapping source	Data quality varies by region
[7]	2014	Yosinski et al. – How Transferable Are Features in Deep Neural Networks?	Transfer Learning	Showed fine-tuning enables strong performance with limited data	Transferability depends on domain similarity

III Methodology

A. Dataset

Images were sourced form open public urban incident repositories and additional with original photographs of incidents covering all the sic categories. The sample had a difference in lighting, viewing angles and weather which helped in generalisation of the model rather than memorisation of the dataset. Table 2 describes the train/validation/test split across the 9,355 total sample.

Table 2: Dataset Composition

Incident Category	Train	Validation	Test
Fire	1,200	300	200
Road Accident	1,400	350	220
Flood	1,100	280	190
Public Disturbance	950	240	160
Road Damage	1,050	260	175
Illegal Dumping	900	225	155
Total	6,600	1,655	1,100

B. Detection Model

YOLOv8 was used because of its 18ms GPU inference time which in within the three-second end-to-end budget and is competitive on small-object detection tasks. Starting from COCO-pretrained weights, fine tuning ran for 50 epochs with SGD. The best validation checkpoint was retained for deployment.

C. Severity Classification

There is a lookup table maps which has detected category to a base severity tier level i.e., Low, Medium, High or Critical and is consistent with a standard emergency management step. The confidence level of the detector adjusts that base: detects below a conservative threshold are downgraded to a one lower tier.

D. Geolocation and Notification

The browser Geolocation API helps in capturing the GPS coordinates at the time of submission of the image of incidence. The backend queries for hospitals and police-station within a three-kilometre radius. Also, if the user provides an email address, an email alert is send with the help of SMTP which has the severity level of the incident, nearest hospitals and police station listed and recommended immediate actions. Also a PDF report is generated in the system.

IV System Architecture & Implementation

The system has five logical layers that exchange data through each other and each layer is independent and replaceable. The input layer is the first layer and it captures the media file, GPS coordinates and optional alert email. The AI Detection layer runs the YOLO model and checks the incident type and confidence score. The Processing layer has the severity.py and location.py and classifies the incident into the severity tier (low, medium and high) and locates nearest hospitals and police stations. The data layer writes a timestamped record to SQLite for audit. The Output layer builds a PDF report and sends an email alert. The backend is a FastAPI application which helps in native multipart support. A single point accepts form, calls each module in sequence and responds. The frontend is a single HTML file which has a drag and drop zone, severity banner, incident summary tiles with map, email submit record and PDF download button.

Table 3: System Layers and Responsibilities

Layer	Module	Responsibility
Input	Web UI, Geolocation API	Media file, GPS coordinates, optional alert email
AI Detection	model.py (YOLOv8n)	Incident type + confidence score
Processing	severity.py, location.py	Severity tier; nearest hospitals and police stations
Data	database.py (SQLite)	Audit log and analytics store
Output	report.py, notifier.py, index.html	PDF report; email alert; browser result panel

V Results and Discussion

A. Detection Accuracy

Table 4 reports precision, recall, F1, and AP@0.5 on the 1,100-image held-out test set. The overall mAP@0.5 of 0.90 is driven by strong performance on visually distinct categories (fire: 0.93, road damage: 0.92) and is pulled down modestly by public disturbance (0.87)

Table 4: Per Category Detection Performance

Incident Category	Precision	Recall	F1	AP@0.5
Fire	0.94	0.91	0.92	0.93
Road Accident	0.91	0.89	0.90	0.91
Flood	0.89	0.88	0.88	0.89
Public Disturbance	0.87	0.85	0.86	0.87
Road Damage	0.93	0.91	0.92	0.92
Illegal Dumping	0.90	0.88	0.89	0.89
Mean (mAP@0.5)	0.91	0.89	0.90	0.90

B. Comparison with Baseline Methods

The presented system is compared with four alternatives presented in Table 5. The proposed system is more accurate compared to the rest alternatives as indicated in the table.

Table 5: Comparison of Detection Approaches

Method	Accuracy	Inference (ms)	Location
Manual CCTV monitoring	Variable	Minutes	No
Background subtraction + threshold	61.2%	38	No
ResNet-50 transfer learning	79.4%	95	No
YOLOv5n baseline	84.7%	22	No
Proposed (YOLOv8n + full pipeline)	92.1%	18	Yes

C. Latency and Facility Lookup

The end-to-end latency was reduced. Each submission is made in five second budget. The location of nearest hospitals and police station were compared and good results found. The gaps left are related to those facilities already found in commercial mapping databases but not yet added to the OpenStreetMap community dataset.

VI Conclusion and Future Scope

The system is an integrated pipeline which combines YOLOv8 visual detection, severity assessment, OpenStreetMap facility, structured PDF report and SMTP alert emails. It is designed, implemented and evaluated. The system has an accuracy of 92.1% and completes the full response cycle which shows the readiness for smart city emergency operations. It has a layered architecture which means modules can be swapped as city requirements change without affecting the rest of the stack. Three extensions would help in broaden impact in future. First, connecting the pipeline to live RTSP streams from CCTV networks would shift operation from image detection to live detection. Second, a mobile application would improve and help people and help in faster detection of the incident. Third, a cloud deployment which would help in concurrent submissions from multiple cities.

Acknowledgment

The authors would like to express their sincere gratitude to their faculty mentors and institution for their continuous support and guidance throughout this research work. We also acknowledge the contributions of the open-source community and researchers whose work in computer vision, smart city systems, and artificial intelligence has significantly influenced this study.

References:

[1] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "YOLO: Unified, Real-Time Object Detection," in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), 2016, pp. 779–788. :contentReference[oaicite:0]{index=0}

[2] J. Redmon and A. Farhadi, "YOLO9000: Better, Faster, Stronger," in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), 2017.
[3] 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, Jun. 2017. :contentReference[oaicite:1]{index=1}
[4] M. Tan, R. Pang, and Q. V. Le, "EfficientDet: Scalable and Efficient Object Detection," in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), 2020, pp. 10781–10790. :contentReference[oaicite:2]{index=2}
[5] S. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "YOLOv4: Optimal Speed and Accuracy of Object Detection," arXiv:2004.10934, 2020.
[6] T.-Y. Lin et al., "Feature Pyramid Networks for Object Detection," in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), 2017, pp. 2117–2125. :contentReference[oaicite:3]{index=3}
[7] A. Vaswani et al., "Attention Is All You Need," in Adv. Neural Inf. Process. Syst. (NeurIPS), vol. 30, 2017. :contentReference[oaicite:4]{index=4}
[8] F. Chollet, Deep Learning with Python, 2nd ed. Shelter Island, NY, USA: Manning Publications, 2021.
[9] W. Sultani, C. Chen, and M. Shah, "Real-World Anomaly Detection in Surveillance Videos," in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), 2018.
[10] R. Mehran, A. Oyama, and M. Shah, "Abnormal Crowd Behavior Detection Using Social Force Model," in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), 2009.
[11] A. Zanella et al., "Internet of Things for Smart Cities," IEEE Internet Things J., vol. 1, no. 1, pp. 22–32, Feb. 2014.
[12] S. P. Mohanty, U. Choppali, and E. Kougianos, "Everything You Wanted to Know About Smart Cities," IEEE Consumer Electron. Mag., vol. 5, no. 3, pp. 60–70, Jul. 2016.
[13] A. Y. Chen, F. Peña-Mora, and Y. Ouyang, "A Collaborative GIS Framework for Emergency Response," J. Comput. Civil Eng., vol. 25, no. 6, pp. 459–468, 2011.
[14] S. J. Pan and Q. Yang, "A Survey on Transfer Learning," IEEE Trans. Knowl. Data Eng., vol. 22, no. 10, pp. 1345–1359, Oct. 2010.
[15] Y. Li et al., "Edge AI: On-Demand Accelerating Deep Neural Network Inference via Edge Computing," IEEE Trans. Wireless Commun., vol. 19, no. 1, pp. 447–457, 2020.
[16] OpenStreetMap Foundation, "Overpass API Documentation," [Online]. Available: https://overpass-api.de
[17] Ultralytics, "YOLOv8 Documentation," [Online]. Available: https://docs.ultralytics.com