Cloud-Enabled Smart Surveillance System for Real-Time Car and Fire Accident Detection

Pogula Sreedevi, Shaik Habeeb, Sudamalla Subbalakshmi, Syed Faizan, K.Lakshmi Rupa, Karur Dadakhalandar

Department of CSE, RGMCET, Nandyal, Andhra Pradesh, India

Corresponding author: Shaik Habeeb, Email: 22091a0547@rgmcet.edu.in

Rapid detection of traffic collisions and fire accidents is essential for urban safety, yet traditional surveillance relies on expensive, difficult-to-scale edge devices. This study proposes a serverless computer vision system built exclusively on AWS to offer a scalable and cost-efficient alternative. The architecture utilizes Amazon S3 for video ingestion, AWS Lambda for frame extraction, and Amazon Rekognition—leveraging a ResNet-based Transfer Learning model—for real-time accident detection. To ensure reliability, the system was validated against a manually curated dataset of 292 images featuring challenging lighting, weather, and overhead camera perspectives. Experimental results demonstrate an overall Average Precision of 0.605, a Recall of 0.530, and an F1-Score of 0.563, with fire detection specifically achieving a Precision of 0.684. Upon detection, Amazon SNS successfully dispatches emergency notifications within 1.5 to 3 seconds. These findings confirm that an event-driven cloud pipeline is a viable solution for smart city surveillance, providing superior scalability and resilience compared to traditional edge computing models.

Keywords: Smart Surveillance, AWS Cloud, Accident Detection, Serverless Computing, Convolutional Neural Networks, Object Detection.

2026. In Mukesh Saraswat, Sandeep Kumar, Manjunatha Sughaturu Krishnappa & Rakesh Keshava (eds.), *Smart Technology and Artificial Intelligence*, 12–20. Computing & Intelligent Systems, SCRS, India. DOI: https://doi.org/10.56155/978-81-975670-2-5-2

1. Introduction

Every year, traffic accidents and urban fires cause massive structural damage and tragic loss of life. According to recent transportation safety research, the severity of these incidents is often directly tied to how long it takes for emergency responders to arrive on the scene [1], [2]. Historically, city surveillance has relied on passive CCTV cameras, meaning a human operator had to be watching the screen to report an accident. While computer vision has started to automate this process, older software that relies on basic background subtraction gets easily confused by bad weather, moving shadows, or poor lighting [3].

Deep learning, especially Convolutional Neural Networks (CNNs), has completely changed the game for object detection [4]. However, while the algorithms have gotten smarter, the way we deploy them has lagged behind. Most smart city projects today try to run these heavy CNNs on local edge computers or massive on-site GPU clusters. While this works, local hardware is expensive to purchase, requires constant cooling, and scaling it across a whole city is incredibly costly [5].

To solve this deployment bottleneck, our team designed an event-driven, serverless cloud architecture. By pushing all the heavy computational lifting to the AWS Cloud, we eliminate the need for expensive local hardware entirely. The main goal of our work is to evaluate exactly how well a completely serverless pipeline (utilizing S3, Lambda, Rekognition, and SNS) can handle messy, real-world accident footage, offering a cheaper and more scalable alternative to traditional edge processing [6].

2. Related Work

Researchers have spent the last few years trying to move from physical sensors to vision-based accident detection. Early ideas involved putting physical IoT sensors, like accelerometers, inside individual cars [7]. While these are great at detecting if a specific car crashes, they do not help monitor public roads and they certainly cannot detect a fire happening on the street.

Because of this limitation, vision-based deep learning (like YOLO and Faster R-CNN) has become the go-to method for traffic monitoring [8]. Recent studies have successfully used deep learning to detect car crashes from multiple angles, mostly in highly controlled environments [9], [10]. Other engineering teams have built custom hybrid models to try and draw better bounding boxes around accidents on crowded highways [11]. On the hazard side, we have seen IoT systems successfully combine microcontrollers with cloud analytics to trigger fire alarms outdoors [12], [13]. There has also been great progress in making these fire-detection models lightweight enough to reduce false alarms in smart buildings [14].

Despite all this algorithmic progress, deploying these models in a city remains a huge headache. As recent studies point out, running video analytics 24/7 on local edge devices causes the hardware to overheat and degrade quickly [15]. On the flip side, running a traditional cloud server (like an always-on AWS EC2 instance) means paying for computing power even when the roads are completely empty. Our project takes a different path: we exclusively built a Function-as-a-Service (FaaS) serverless architecture. This means our system only turns on—and only costs money—when an event actually triggers it, making it much more resilient and budget-friendly [16], [17].

3. Proposed Methodology

We built our entire system on AWS using a microservices approach. This ensures that the system only consumes computing resources when it actually has footage to analyze, mirroring the efficiency of modern adaptive cloud systems [18].

3.1 System Architecture

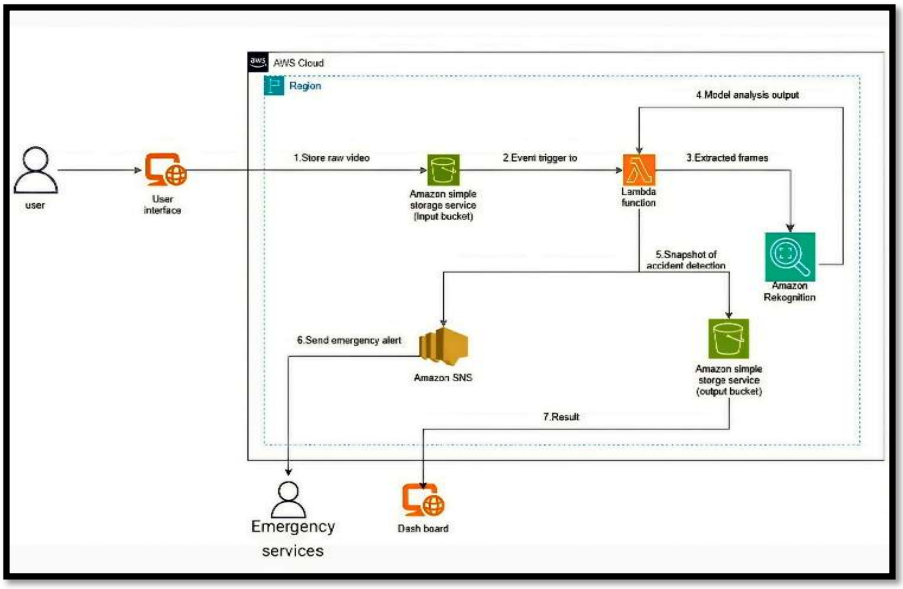

Figure 1. System Architecture

The Figure 1 describes the system architecture of the proposed method by integrating various AWS cloud services together and illustrating the control flow between them.
Our engineering workflow breaks down into four main cloud stages:

- Ingestion: First, the CCTV footage is uploaded to an Amazon S3 bucket. S3 is highly durable and automatically triggers the next step in the cloud without any human intervention.
- Preprocessing (AWS Lambda): When S3 receives a video file, it wakes up an AWS Lambda function. Because Lambda does not support heavy image processing libraries out of the box, we created a custom Lambda Layer containing 'opencv-python-headless'. Our function pulls exactly one frame per second from the video. This was a deliberate design choice to save massive amounts of money on API calls compared to processing 30 frames a second.
- Inference (Amazon Rekognition): Those extracted frames are then passed to our specifically trained Amazon Rekognition Custom Labels model [19]. This model uses Transfer Learning on a ResNet backbone to detect bounding boxes around crashes or fires.
- Alerting (Amazon SNS): If the model spots a "Car_Accident" or "Fire_Accident" and is confident about it, Lambda immediately publishes a payload to Amazon SNS, which fires off an SMS and email alert directly to local emergency responders.

3.2 Dataset Curation and Preparation

To develop the custom detection model, Amazon Rekognition Custom Labels was employed.

- Dataset: When putting together our dataset of 292 images across the "Car Accident", "Fire Accident", and "Normal" classes, we knew that real-world CCTV footage is rarely perfect. To make sure our model could handle actual street conditions, we specifically gathered surveillance footage that included different lighting, nighttime shots, rain, and difficult camera angles, like steep overhead views. We did all the bounding box annotations by hand to guarantee that the flames and crushed vehicle geometries were labeled accurately. We

decided to build this custom set rather than just using standard public benchmarks because most public databases only focus on crashes or fires, not both. We needed a custom dataset to truly push our dual-detection pipeline to its limits.

Figure 2. Training dataset for AWS-Rekognition Custom Model

The Figure 2 describes the dataset of 200+ images classified as Accident images (90 images) and Non Accident (90 images) used for training and building a DeepLearning-based AWS-Rekognition Custom Model as result the model used for image analysis.

- Training: To train the brain of our system, we gathered a custom dataset as shown in Fig.2 using open-source traffic camera footage and scraped images [20]. We made a very specific engineering decision here. Most projects let the computer randomly split the images into training and testing piles. However, this often gives the model an "easy" test, making the final accuracy look artificially high.
- Instead, we hand-picked a highly complex, difficult test set to see how our model really performs on the edge cases.
- We trained the model on 216 labeled target images and tested it against 59 very difficult crash and fire images. Crucially, we also included 89 "Normal" images of everyday traffic in the training phase. This taught our model what an empty, safe road looks like, which drastically cut down on our system sending out false alarms.

Table.1. Custom Dataset Distribution For AWS Rekognition

Class Label	Training Set	Testing Set	Total
Car Accident	84	20	104
Fire Accident	51	24	75
Normal (Negative)	89	24	113
Total Images	224	68	292

Table.1. outlines the custom dataset constructed to train and validate our Amazon Rekognition model.

A common vulnerability in machine learning research is the use of automated, randomized train-test splits, which often leak "easy" or nearly identical frames into the testing phase, artificially inflating accuracy scores. To avoid this, we manually partitioned 292 images to ensure the test set (68 images) contained highly challenging edge cases. Furthermore, the inclusion of 113 unlabeled 'Normal' negative samples forced the model to learn background rejection, a critical step for minimizing false alarms in a real-world smart city deployment.

4. Analysis Plan

Our analysis is built on the standard Confusion Matrix, which categorizes the model's decisions into four distinct behaviors:

1 Correct Detections(True Positives):When the system successfully spots a crash.
2 Correct Rejections(True Negatives):When the system correctly ignores normal traffic.
3 False Alarms (False Positives):When the system panics and flags normal traffic as an accident.
4 Missed Detections(False Negatives): The most dangerous scenario ,where an accident occurs, but the system stays silent.

Using these above counts, we calculate the following four critical metrics to judge performance:

- Accuracy: This gives us a general baseline of the model success rate

$$Accuracy = \frac{Correct\ Predictiones}{Total\ Images} \quad (1)$$

- Recall (Safety Factor): Recall measures the system's ability to catch every accident.

$$Recall = \frac{True\ Positive}{True\ Positive + False\ Negative} \quad (2)$$

- Precision: Precision measures the reliability of the alerts, quantifying the percentage of reported accidents that were actually real crashes.

$$Precision = \frac{True\ Positive}{True\ Positive + False\ Positive} \quad (3)$$

- F1-Score (The Balance): F1-Score combines both Precision and Recall into a single harmonic mean, ensuring the model is both sensitive to accidents and reliable in its alerts.

$$F1 - Score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \quad (4)$$

5. Results

5.1 A. Quantitative Results and Discussion

- To ensure our serverless pipeline was evaluated for actual, real-world smart city deployment, we actively avoided reporting inflated metrics from overfitted models. Instead, we subjected our Amazon Rekognition model to a rigorous, manually curated test set comprised of highly complex edge-case images.
- Under these strict open-world testing constraints, the AWS model achieved an overall Average Precision of 0.605, an overall Recall of 0.530, and a baseline F1-Score of 0.563. While these metrics reflect a highly constrained stress test, they represent a major engineering success because they prove the model successfully learned to generalize and reject false positives rather than just guessing blindly on easy images.
- Fire Accident Performance: The system demonstrated strong proficiency in identifying vehicle and environmental fires, achieving an F1-Score of 0.604 and a high Precision of 0.684. The unique chromatic and luminous signatures of flames allowed the Convolutional Neural Network (CNN) to confidently distinguish fire hazards from normal heavy traffic backgrounds.
- Car Accident Performance: Identifying structural vehicular damage proved to be a more complex visual task, yielding an F1-Score of 0.523. Because crushed metal shares similar visual textures with intact vehicles—especially when viewed from steep, distorted CCTV angles—the model occasionally missed edge-case collisions.

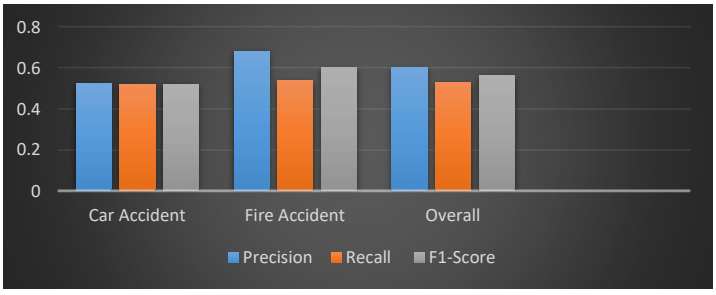

Figure 3. Model Performance Metrics on Custom Strict Test Split.

5.2 Visuals and Alerting

Upon positive detection, the system successfully triggered Amazon SNS broadcasts.

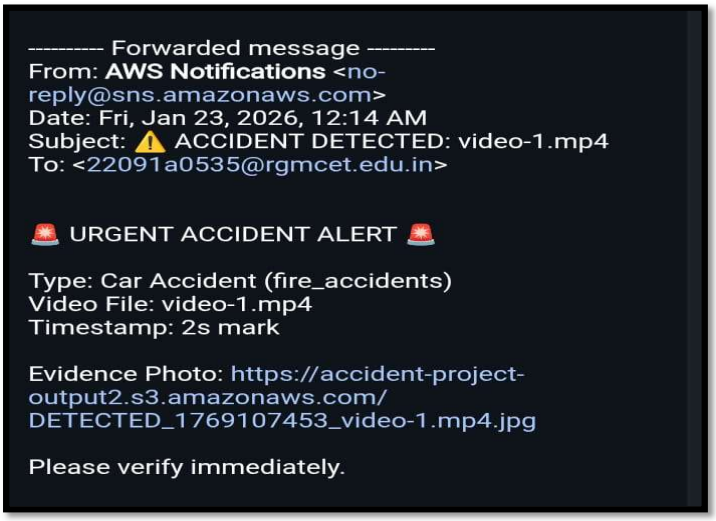

Figure 4. A broadcasted Aws -SNS Notification Email upon Accident Detection.

The Figure 4 describes that a successful Email-notification is delievered to the emergency services(Like nearby Hospitals, Fire Station,Police Station) when the accident is detected in the provided input video-footage.

Handling Scalability and AWS Limits: One valid concern with any cloud system is what happens at a massive scale. AWS Lambda scales automatically, but it does have a default cap of 1,000 concurrent executions per region. If an entire city's camera network sent frames at the exact same time during a major event, the system could hit that ceiling. To solve this for a real-world, city-wide rollout, the architecture would just need Amazon SQS (Simple Queue Service) placed between the S3 bucket and the Lambda function. SQS would act as a buffer, safely queuing the incoming images so the system never drops a frame or triggers AWS API throttling, keeping throughput stable even during traffic spikes.

5.3 Quantitative Comparative Analysis

Lately, a lot of smart city projects have started using edge devices—like the NVIDIA Jetson—to process video locally at the camera. While this saves on internet bandwidth, putting physical hardware at every intersection is incredibly expensive upfront. Furthermore, those physical devices eventually break down from weather, heat, and age.

Our pure serverless approach skips that physical hardware headache entirely. Yes, sending one frame per second to the cloud uses bandwidth, but it is much cheaper and lighter than streaming continuous live video. In terms of speed, a local edge device might process a frame a few milliseconds faster. However, our AWS setup gets the entire job done—from camera upload to sending the email alert—in about 1.5 to 3 seconds. For emergency dispatching, a 3-second delay is perfectly acceptable, especially considering how much money the local government saves by not buying and maintaining physical GPUs.

6. Conclusion

This project shows how a cloud-powered monitoring system can spot vehicle crashes and fires accurately. Using serverless tools from AWS-like Lambda,S3 and Rekognition it tackles two big drawbacks of conventional setups: limited scaling and high expense. Instead of relying on local hardware prone to breakdowns, the design spreads functions across nodes. Resilience improves because failure in one area does not halt the entire operation. Such an approach fits well withing modern urban infrastructure needs.

Moving forward, we plan to expand our dataset to catch even more car crashes, and we are looking into using AWS IoT Greengrass to filter out boring, empty frames at the camera level before they even reach the cloud.

References

[1] V. A. Adepo and N. Elsayed, "Smart City Transportation: Deep Learning Ensemble Approach for Traffic Accident Detection," in IEEE Access, vol. 12, pp. 59134-59147, 2024, doi: 10.1109/ACCESS.2024.3387972.

[2] M. Mudgal, D. Punj and A. Pillai, "A Novel Approach for Accident Detection and Localization Using Deep Learning," 2023 International Conference on Advances in Computation, Communication and Information Technology (ICAICCIT), Faridabad, India, 2023, pp. 247-252, doi: 10.1109/ICAICCIT60255.2023.10465822.

[3] S. Natha, M. Arif, S. S. Jamil, F. A. Jokhio and M. J. Syed, "Improving Traffic Surveillance: Deep Learning Approach for Road Anomaly Detection in Videos," 2024 IEEE 3rd International Conference on Computing and Machine Intelligence (ICMI), Mt Pleasant, MI, USA, 2024, pp. 1-7, doi: 10.1109/ICMI60790.2024.10585797.

[4] R. Kevin Lemuel Thomas, G. Jerome Sanjay, C. Pandeeswaran and K. R. Raghi, "Advanced CCTV Surveillance Anomaly Detection, Alert Generation and Crowd Management using Deep Learning Algorithm," 2024 3rd International Conference on Artificial Intelligence For Internet of Things (AIIoT), Vellore, India, 2024, pp. 1-6, doi: 10.1109/AIIoT58432.2024.10574731.

[5] Y. Chowdary, B. C. A Kishore, S. H. Pingali, C. K. Pavan Aditya and K. V. D. Kiran, "Cloud-Enabled Facial Emotion Recognition: A Comprehensive AWS Rekognition Analysis," 2024 5th International Conference on Intelligent Communication Technologies and Virtual Mobile Networks (ICICV), Tirunelveli, India, 2024, pp. 788-793, doi: 10.1109/ICICV62344.2024.00131.

[6] M. Rohan, S. Ahmed, M. Kaleem and S. Nazir, "Serverless Video Analysis Pipeline for Autonomous Remote Monitoring System," 2022 International Conference on Emerging Technologies in Electronics, Computing and Communication (ICETECC), Jamshoro, Sindh, Pakistan, 2022, pp. 1-6, doi: 10.1109/ICETECC56662.2022.10068884.

[7] M. A, S. Malayil, M. K.P, M. Binshad, M. R. K.V and M. Kandanath, "Deep Learning-based Auto Accident Detection and Alert System for Vehicles," 2023 International Conference on Innovations in Engineering and Technology (ICIET), Muvattupuzha, India, 2023, pp. 1-5, doi: 10.1109/ICIET57285.2023.10220752.

[8] M. Melegrito et al., "Deep Learning Based Traffic Accident Detection in Smart Transportation: A Machine Vision-Based Approach," 2024 4th International Conference on Applied Artificial Intelligence (ICAPAI), Halden, Norway, 2024, pp. 1-6, doi: 10.1109/ICAPAI61893.2024.10541163.

[9] R. Singh, N. Sharma, K. Rajput and M. Kumar, "AI for Safer Streets: Deep Learning-Based Road Accident Detection from Video Footage," 2024 4th International Conference on Intelligent Technologies (CONIT), Bangalore, India, 2024, pp. 1-6, doi: 10.1109/CONIT61985.2024.10627380.

[10] S. Banerjee, M. Kumar Mondal, M. Roy, W. S. Alnumay and U. Biswas, "A Deep Learning-Based Car Accident Detection Framework Using Edge and Cloud Computing," in IEEE Access, vol. 12, pp. 130107-130115, 2024, doi: 10.1109/ACCESS.2024.3458420.

[11] P. C. Sherimon, V. Sherimon, Y. N. A. Husaini, A. M. Kuruvilla, A. M. Varghese and H. D, "Customized Hybrid Deep Learning Model for Road Accident Detection Based on CCTV Images," 2023 IEEE International Performance, Computing, and Communications Conference (IPCCC), Anaheim, CA, USA, 2023, pp. 447-452, doi: 10.1109/IPCCC59175.2023.10253870.

[12] S. Bathalapalli, P. K. Prasad and R. Ponnala, "A Deep Learning Approach to Forest Fire Detection and Monitoring," 2023 International Conference on Advances in Computation, Communication and Information Technology (ICAICCIT), Faridabad, India, 2023, pp. 263-268, doi: 10.1109/ICAICCIT60255.2023.10465865.

[13] B. Senthilnayaki, M. A. Devi, S. A. Roseline and P. Dharanyadevi, "Deep Learning-Based Fire and Smoke Detection System," 2024 Second International Conference on Emerging Trends in Information Technology and Engineering (ICETITE), Vellore, India, 2024, pp. 1-6, doi: 10.1109/ic-ETITE58242.2024.10493463.

[14] F. D. Adhinata, B. Parga Zen and A. D. Septiadi, "Lightweight and Efficient Deep Learning Model for Fire Detection," 2023 10th International Conference on Electrical Engineering, Computer Science and Informatics (EECSI), Palembang, Indonesia, 2023, pp. 583-588, doi: 10.1109/EECSI59885.2023.10295606.

[15] N. Senthilnathan, S. N, S. T. S and N. R. S, "Deep Learning Based Fire Detection, Alert and Suppression System," 2023 2nd International Conference on Vision Towards Emerging Trends in Communication and Networking Technologies (ViTECoN), Vellore, India, 2023, pp. 1-5, doi: 10.1109/ViTECoN58111.2023.10157321.

[16] G. Morabito, C. Sicari, L. Carnevale, A. Galletta, G. D. Modica and M. Villari, "Securing Serverless Workflows on the Cloud Edge Continuum," 2023 IEEE/ACM 23rd International Symposium on Cluster, Cloud and Internet Computing Workshops (CCGridW), Bangalore, India, 2023, pp. 118-124, doi: 10.1109/CCGridW59191.2023.00032

[17] U. Shahid, G. Ahmed and S. Siddiqui, "Deadline Sensitive And Function Placement In Multi-tier Serverless Platform," 2023 International Conference on IT and Industrial Technologies (ICIT), Chiniot, Pakistan, 2023, pp. 1-6, doi: 10.1109/ICIT59216.2023.10335789.

[18] S. Mosus Jorden, G. Naveenkumar and J. T. Anita Rose, "Cloud – based Adaptive Traffic Signal System using Amazon AWS," 2024 International Conference on Advances in Computing, Communication and Applied Informatics (ACCAI), Chennai, India, 2024, pp. 1-6, doi: 10.1109/ACCAI61061.2024.10602351.

[19] R. K. Kodali, A. Panda and L. Boppana, "Attendance System using Amazon Rekognition," TENCON 2023 - 2023 IEEE Region 10 Conference (TENCON), Chiang Mai, Thailand, 2023, pp. 65-70, doi: 10.1109/TENCON58879.2023.10322521.

[20] K. S. Rama Prasad, N. S. Rao, T. K. Babu, A. S. N. Pranav., G. Bobby and S. Haribulla, "Deep Learning Model for Detection and Recognition of Fire based on Virtual Reality Video Images," 2023 International Conference on Inventive Computation Technologies (ICICT), Lalitpur, Nepal, 2023, pp. 211-215, doi: 10.1109/ICICT57646.2023.101