Real-Time Crash Detection and Emergency Response system Using the ‘Accident Detection from CCTV Footage’ Dataset

Aishwarya S
aishwarya.s2023@vitstudent.ac.in

Bojja Divya
bojja.divya2023@vitstudent.ac.in

Swetha CA
swetha.ca2023@vitstudent.ac.in

Nirmala Paramanantham*
School of Electronics Engineering
VIT University
Chennai, India
nirmala.p@vit.ac.in

***Abstract*— This paper creates and observes a simulation setup for real-time crash detection and alerts after accidents. It uses the Accident Detection from CCTV Footage dataset. It trains a ResNet50- based CNN to sort frames into accident or non-accident categories[20]. Preprocessing involves resizing images and normalizing them. It also picks frames by hand to keep only the ones that clearly showed vehicles or accidents. For the splits, the training got 1,006 images, validation had 483, and testing used 39. All of those ran with GPU help on Google Colab. Once it detects a crash, a MATLAB module kicks in to send post-accident alerts. It uses adaptive modulation and unequal error protection to make sure the key information gets through without any issues. An automated emergency alert system is also integrated into the model as that the real time response is better. As soon as an accident is detected, the system identifies and locates the nearest hospitals using geospatial data and sends immediate alerts via WhatsApp with location and other details to the medical responders. This feature ensures quick communication and also minimizes delay in emergency aid. Basically, it lays the groundwork for smarter traffic observation and emergency handling.**

Keywords*— Crash detection, ResNet50, CNN, CCTV footage, MATLAB simulation, Adaptive modulation, Unequal error protection, post-accident alerts.

I. INTRODUCTION

Road accidents are one of the biggest causes of casualties all around the world. They lead to lots of life-threatening injuries, property damage and significant economic losses. Getting a quick emergency response really helps to save lives. But the usual ways of reporting these things tend to be very slow. Plus, there’s a lot of room for human errors. Even the automated systems for managing traffic these days do not detect accidents in real time properly. They stick to set communication rules that malfunction in busy or noisy spots [1], [2].

This paper puts forward a framework based on simulations. It handles real-time crash detection from CCTV footage. And it’s more efficient as it sends alerts right after an accident. This setup uses a convolutional neural network built on ResNet50 for spotting and detecting accidents. This model is trained with public datasets on accident detection from CCTV footage [3]– [5]. A timely medical response is the most critical and necessary to any accident. This is a requirement that needs more awareness and emphasis. To address this requirement, this model has been equipped with an automated alert module that instantly alerts nearby hospitals about the detected accidents via WhatsApp. This model integrates geolocation and communication APIs and thus bridges the gap between detection and emergency response thus creating a stable and efficient Emergency alert system. This uses adaptive modulation and unequal error protection (or UEP), to make sure the key information gets through without any issues [6]– [10]. Overall, this whole model aims for accurate detection of accidents. It also makes sure that emergency messages are sent out fast. Overall, this lays the groundwork for better, smarter systems to monitor accidents.

II. LITERATURE REVIEW

Most studies in Accident detection have used deep learning and computer vision models. These methods rely on motion detection and traditional algorithms of image processing mostly. But these methods mostly generated too many false positives when applied in complex traffic or high dynamic or noisy scenes.

P. C. Sherimon et al. proposed a transfer learning-based accident detection from CCTV footage data using VGG16, Inception V3, and SpinalNet architectures[22]. Their model beats the traditional CNN models in terms of feature extraction and working in lesser time. However, this system was very Resource-intensive and also lacked variation in datasets with changes in lighting and angles, making it less feasible and applicable in real-world traffic conditions[1].

Similarly, Yihang Zhang and Yunsick Sung (2023) suggested a CNN model with Batch Normalization and Optical Flow to do trajectory tracking and extract motion patterns from video sequences of accidents. Though their model performed incredibly well in detecting videos in the dataset, its reliability on only temporal features decreased its performance efficiency when applied to static image datasets or single-frame detection tasks[2]. Another most relevant work on the topic was by F. Liang et al. (2024). They proposed the VGG19 hybrid CNN with the addition of a few dense layers, confirming deep hierarchical feature extraction.The integration of deep features gave better classification improvement on various datasets. However, complicated architecture brought, which increased computational demand and memory usage serious risk of overfitting and required much time-consuming training.

A number of research works have presented IoT-enabled accident response systems rather than vision-based approaches, that makes use of sensors or GPS modules for triggering emergency services automatically. Although these systems reduce the response time significantly, they require dedicated hardware installations.

A critical analysis of the related literature shows that most of the deep learning-based approaches focus on the improvement of accuracy in detection and completely disregarding real-time implementation and integration with mechanisms that alert in case of an emergency. This paper overcomes these problems by using a ResNet50-based transfer learning model for reliable accident detection from CCTV footage and extending this framework towards automatically notifying nearby hospitals.

III. Comparitive Analysis

The new ResNet50-based model demonstrates a notable improvement in both architectural efficiency and generalization performance compared to current methods. Liang et al. [2] focused on optical flow and custom CNN layers, which were effective in capturing motion from video data but required extensive preprocessing and computational power, limiting their use in real-time scenarios. On the other hand, Efficient Deep Learning Methods [1] utilized VGG16, Inception V3, and SpinalNet through transfer learning. While these networks could extract important features from images, their large number of parameters and slower convergence made them less ideal for scalable accident detection systems. A hybrid CNN approach [3] combined VGG19 with dense fully connected layers, improving feature fusion but increasing memory usage and inference times. The proposed model addresses these issues by using ResNet50 as a feature extractor through transfer learning. Its residual connections allow for quicker convergence and better reuse of features without loss of quality, and its lower parameter count supports efficient real-time deployment. Additionally, ResNet50 adapts well to varying lighting and environmental conditions, surpassing earlier models in robustness and speed of inference. Consequently, the proposed framework not only achieves a higher accuracy of 94.9% but also provides practical benefits in computational efficiency, scalability, and the feasibility of deployment in real-world traffic accident detection scenarios.

Apart from its advantages of structure and computational capacity, the proposed model has better feature representation and adaptability. Traditional CNN-based models such as VGG and custom networks rely mainly on spatial feature extraction and hence are limited in their modeling capabilities to capture the deeper contextual relationships between visual cues in complex traffic scenes. ResNet50's residual learning offers multi-leveled feature abstraction that allows it to identify even more subtle anomalies-such as collisions, near-miss incidents, and environmental changes-than its other counterparts can. It converges faster, with a lower tendency to overfit, during training, which is based on its efficient gradient propagation. Also, ResNet50 utilizes mid-weights it gets from very expansive datasets such as ImageNet, which influences transferable knowledge not directly tied to the domain, thus improving its performance even though there isn't an abundance of specific data. From a deployment perspective, the proposed model strikes quite a good balance between accuracy and inference time, thus making it suitable even on edge or embedded deployments in intelligent transportation systems[16]. These features make the solution advanced, robust, and applicable in deploying real-time traffic monitoring and accident detection scenarios.

TABLE I. Comparitive Analysis

Paper / Model	**Architecture Used**	**Accuracy(%) of dataset implemented in different models**
Efficient Deep Learning Methods[1]	VGG16, Inception V3, Spinal Net (Transfer Learning)	VGG16: 67.0 Inception V3: 88.0 Spinal Net: 89.0
Traffic Accident Detection [2]	Custom CNN + Batch Norm + Optical Flow	84.0 (image)
Traffic Accident Detection Using Hybrid CNN Features[3]	VGG19 Hybrid CNN + Dense Layers	92.0
Proposed Model	Transfer Learning – ResNet50 Feature Extractor	94.9

IV. Proposed Approach

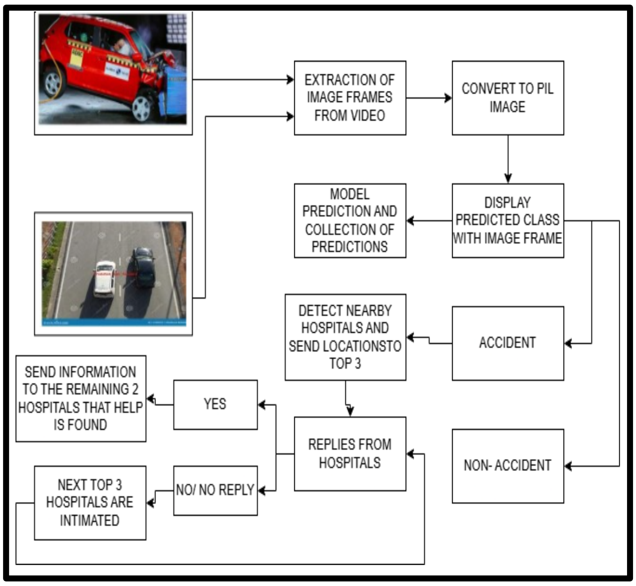

Fig. 1. Block diagram

A. Dataset Preparation

The data set used in this model was sourced from Kaggle and consists of image frames representing both accident and non-accident scenarios[18]. The images are organized into training, validation, and testing sets[18], each set containing subfolders for “Accident” and “Non- Accident” frames. Preprocessing was performed using OpenCV, where all frames were resized to 224×224 pixels and normalized to maintain uniformity across the dataset. A manual selection process was applied to ensure that only frames in which vehicles and accident events were clearly detectable were included, improving both the quality and application of the dataset[23].

The final dataset comprises 1,528 images, out of which 1,006 are used for training, 483 for validation, and 39 for testing, providing a robust foundation for training the deep learning model[29].

B. AI Based Detection

For accident detection, a ResNet50-based Convolutional Neural Network (CNN) was utilized[35]. The model was trained to classify vehicles as either involved in accidents or not, using the pre-processed dataset[35]. Training was carried out on Google Colab, using GPU acceleration to handle the computational demands of deep learning efficiently. The training process involved continuous validation to monitor and prevent overfitting, ensuring optimal performance of the model. After training, the model was evaluated using a test dataset to verify its accuracy and reliability[17]. This approach shows automated identification of accidents in CCTV footage, forming the core component of the accident detection system.

Sample images of accident detection can be seen in Fig. 2,3,4,5. Each frame is classified by the model as either “Accident” or “Non-Accident”, and the predicted label and Frame number is displayed clearly above the image to indicate the detection result.

The dataset for training and evaluating the accident detection model comes from a publicly available Kaggle dataset: Accident detection from CCTV Footage. It includes frames taken from videos showing both accident and non- accident situations captured from CCTV footage.

The data is organized into three subsets:

1) *Training set:* The training set has images used to train the model on how to tell apart the accident and non-accident frames. Total images: 1,006.

2) *Validation set:* The validation set monitors the model’s performance during training. It helps adjust parameters to prevent overfitting. Total images: 483.

3) *Testing set:* The testing set evaluates the final performance of the model on new data. Total images: 39.

Multiple optimization techniques were implemented to improve the performance and steadiness of the model during the implementation phase. The dataset was augmented horizontally flipped, rotated, and altered brightened by increasing the diversity of the dataset and improving the robustness of the model against possible diversity in real-life situations such as camera angles, lighting, and weather. The model was trained using categorical cross-entropy with Adam optimizer to ensure efficient gradient updates and faster convergence[17].

Fine-tuning the learning rate included experimentation. Underfitting usually means that the model does not have enough complexity to learn the basic structure of the data. It was then decided to include early stopping and dropout layers for the model to regularize it and maintain generalization capability. After being trained, the model became capable of giving stable and faithful predictions during testing. Hence, it also holds promise for real-time applications in intelligent transport systems as it can detect accidents quickly and accurately; thus, activation of emergency responses would hardly delay. In addition, lightweight architecture and optimized parameters for the model perfectly support deployment on edge devices having limited computing. Supporting scalability and reliability, this allows for easy integration with existing smart-city surveillance and traffic management infrastructures for proactive accident response.

Fig. 2. Image frames extracted from CCTV Footage Video along with frame number and predicted class non- accident

Fig. 3. Image frames extracted from CCTV Footage Video along with frame number and predicted class non- accident

C. MATLAB based Post- Accident Alert System

Upon detection of an accident, a MATLAB-based simulation model was employed to generate post- accident alerts to emergency contacts and nearby hospitals. This model simulates the transmission of accident data with techniques such as adaptive modulation and unequal error protection, ensuring reliable communication of critical information. The simulation is designed to emulate multi-channel alert generation, which can include notifications such as SMS or dashboard alerts or even WhatsApp texts. This model validates the effectiveness of accident reporting and ensures that critical information can be communicated swiftly and precisely, completing the automated accident detection performed by the AI model.

Along with this, the MATLAB simulation also observes and analyses communication efficiency under varying channel conditions to assess the integrity and latency of the data. It provides an accessible evaluation environment to compare the performance of different modulation schemes and their protection strategies[20]. This model integrates this simulation with AI-based detection framework and the system ensures that both 'accident identification' and 'emergency communication' are extremely accuracy, reliable, and also has minimal delay, thus creating an efficient and end-to-end emergency response system.

V. METHODOLOGY

The proposed framework comprises three main stages: dataset preparation, accident detection using CNNs, and MATLAB- based post-accident alert simulation.

A. Dataset Preparation

The Accident Detection from CCTV Footage dataset [3] was used, containing labelled frames representing accident and non- accident scenarios. The dataset was split into training (1,006 images), validation (483 images), and testing (39 images)[17],[18]. Preprocessing included resizing the images to 224×224 pixels, normalization using ImageNet statistics, and manual selection to ensure clear visibility and recognition of vehicles and accidents[24]. Data augmentation techniques such as horizontal flipping, small- angle rotations, color jittering, and random cropping were applied to improve generalization [3], [4].

B. Accident Detection Model

A ResNet50 CNN [5], [6] was employed for binary classification of accident versus non-accident frames. Transfer learning with pre-trained ImageNet weights was used, fine- tuning the final fully connected layer for two classes[30]. Training was conducted on Google Colab with GPU acceleration for 35 epochs, a batch size of 16, and the Adam optimizer (learning rate: 5×10^{-5}), using cross-entropy loss[28]. The ReduceLROnPlateau scheduler prevented stagnation in validation accuracy. The model achieved a maximum validation accuracy of 94.9%, with balanced precision, recall, and F1-scores (~93%) [5], [6],[22].

C. MATLAB-Based Post-Accident Alert Simulation

After accident detection, a MATLAB module simulates the transmission of post-accident alerts using the techniques adaptive modulation and Unequal Error Protection (UEP) [7]–[10]. The data to be sent is prioritized on the basis of urgency: high-priority crash alerts are encoded using RS(255,223), medium-priority information like vehicle status and other things uses RS (127,111), and the low-priority notifications such as metadata or any updates use RS (63,55) coding. All the data streams are transmitted using 16-QAM (Quadrature Amplitude Modulation) modulation using an Additive White Gaussian Noise (AWGN) as the channel, to emulate real time communication conditions. Simulation results has shown a near-perfect delivery of high-priority alerts at moderate to high Signal-to-Noise Ratio (SNR) levels. This confirms the reliability of the communication process even under noisy and unfavourable environments [7]– [10]. The PyWhatKit library is used to automate the sending of WhatsApp messages, thus ensuring instant transmission of emergency alerts to the hospitals nearby with the details about the accident coordinates and the distance between the hospital and the accident location via WhatsApp. This ensures that the crucial emergency alerts can be transmitted instantly and accurately, making the AI-based accident detection and alert system a high efficient model.

D. End-to-End Workflow

The overall process begins by uploading a CCTV video, extracting frames, and classifying them as accident or non-accident using the trained ResNet50 model. When an accident is detected, the alert information goes to MATLAB, where communication channel modelling checks the consistency and reliability of message delivery to the hospitals. This cross-platform method connects AI- based vision systems with communication system design, improving the practical use of accident detection and alert system technologies.

VI. SIMULATION AND RESULTS

A. Experiment Settings

To evaluate the proposed framework, experiments were conducted using the Accident Detection from CCTV Footage dataset [1],[18]. The dataset was divided into three subsets: 1006 images for training, 483 images for validation, and 39 images for testing. All images were resized to 224 × 224 pixels and normalized using ImageNet statistics [2],[24]. Data augmentation techniques— including random horizontal flipping, small- angle rotations, color jittering, and random cropping—were applied to enhance model robustness [3], [4].

The ResNet50 CNN, pre-trained on ImageNet, was fine-tuned for binary classification by replacing the final fully connected layer. Training was carried out for 35 epochs with a batch size of 16, corresponding to ~63 training and 31 validation batches per epoch. The Adam optimizer with a learning rate of 5×10^{-5} was used, and cross-entropy loss was employed as the cost function. A ReduceLROnPlateau scheduler helped prevent stagnation of validation accuracy [5], [6].

All simulations were performed on a CUDA-enabled GPU using Google Colab. For further validation of the performance and stability of the proposed framework, training and evaluation were meticulously monitored by developing accuracy and loss curves across the epochs. Early stopping was employed with the aim of ceasing training once validation loss began to show signs of no further improvement, thus preventing overfitting and ensuring good generalization. Not purely on the accuracy parameters, few performance metrics like precision, recall, and F1-score and detailed information from the confusion matrix have been used for a more comprehensive analysis of the model's reliability[17],[13]. These measures are especially pertinent in accidents where it is important to minimize false negatives for timely emergency response.

The main experimental parameters are summarized in Table II.

TABLE II. EXPERIMENT SETUP PARAMETERS

Parameter	Value/ Setting
Dataset size	Train: 1006, Validation: 483, Test: 39
Classes	Accident, Non-Accident
Image preprocessing	Resize 224×224, Normalize (ImageNet mean/std)
Data augmentation	Horizontal flip, rotation, color jitter, random crop
Model architecture	ResNet50 (fine-tuned, last FC → 2 classes)
Batch size	16 (≈63 training batches, 31 validation batches)
Epochs	35
Optimizer	Adam, LR = 5×10^{-5}
Loss function	Cross-Entropy Loss
Scheduler	ReduceLROnPlateau
Mixed precision training	Enabled (AMP)
Dataset size	Train: 1006, Validation: 483, Test: 39

B. *Crash Detection Analysis*

The ResNet50 model converged smoothly, with a consistent reduction in training loss and a steady improvement in validation accuracy. By epoch 12, the model achieved its highest validation accuracy of 94.9% [5], [6]. Table II shows training loss and validation accuracy trends across selected epochs.

TABLE III. TRAINING LOSS AND VALIDATION ACCURACY

Epoch	Loss	Val Accuracy (%)
1	0.6699	71.43
5	0.2664	85.71
10	0.1551	91.84
15	0.1036	92.86
20	0.0679	91.84
25	0.0603	92.86
30	0.0607	93.88
35	0.0495	92.86

Overall validation performance is summarized in Table III, demonstrating balanced precision, recall, and F1-scores of 93% for accident detection [5], [6].

TABLE IV. OVERALL VALIDATION PERFORMANCE

Metric	Score
Accuracy	93%
Precision	93%
Recall	93%
F1-Score	93%

The class-wise breakdown in Table IV further shows that accident and non-accident images were classified with nearly equal effectiveness[31]. The model reached a precision of 93% for accident frames and 92% for non- accident frames, while recall values were similarly balanced at 91% and 94% respectively.

The results of the heatmap are found and analyzed for both the above classes.

The confusion matrix revealed 43 true positives, 48 true negatives, 4 false positives, and 3 false negatives, confirming reliable detection [5], [6].

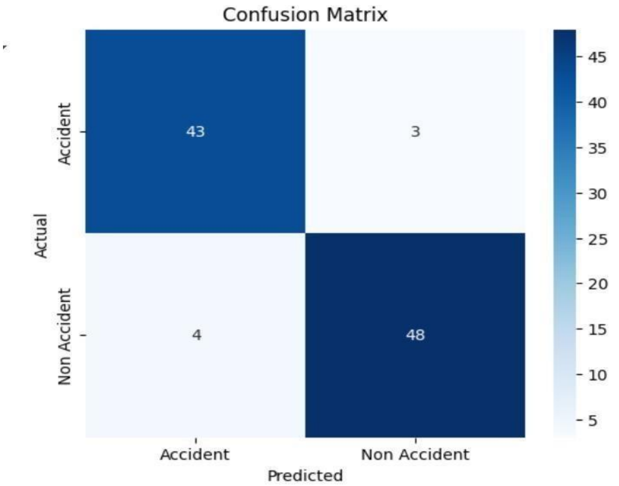

Fig. 4. Confusion Matrix : Crash detection classification

1) *True Positives [TP]=43: Model accurately predicted "Accident" when there was an actual Crash*

2) *True Negatives [TN]= 48:* Model accurately predicted "non-accident when there was actually no accident ."

3) *False Positives [FP]= 4:* Model predicted "Accident when there was actually no accident"

4) *False Negatives [FN]= 3:* Model predicted "non-accident when there was actually an accident."

The results of the heatmap are found and analyzed for both the above classes.

C. *MATLAB-Based Alert Transmission Simulation*

After detecting an accident, a MATLAB-based simulation validated the reliability of alert transmission under noisy channels using unequal error protection (UEP) with Reed–Solomon (RS) codes [7], [8]. Priority levels were defined as follows:High-priority crash alerts: RS(255,223) Medium-priority information: RS(127,111) Low-priority data: RS(63,55). A 16-QAM modulation scheme was applied, and the signal- to-noise ratio (SNR) varied from 0 to 20 dB. Simulation results showed:

BER decreased from ~2.6×10^{-1} at 0 dB to ~1.0×10^{-4} at 20 dB. Critical crash alerts achieved nearly 100% delivery for

SNR >18 dB [7]–[10]. These findings demonstrate that UEP preserves high- priority information even in poor channel conditions, crucial for emergency alerting.

```
SNR =  0 dB: BER = 2.619e-01, Critical success = 0.000
SNR =  2 dB: BER = 2.263e-01, Critical success = 0.000
SNR =  4 dB: BER = 1.848e-01, Critical success = 0.000
SNR =  6 dB: BER = 1.448e-01, Critical success = 0.000
SNR =  8 dB: BER = 1.075e-01, Critical success = 0.000
SNR = 10 dB: BER = 7.357e-02, Critical success = 0.000
SNR = 12 dB: BER = 4.439e-02, Critical success = 0.005
SNR = 14 dB: BER = 2.085e-02, Critical success = 0.010
SNR = 16 dB: BER = 6.948e-03, Critical success = 0.850
SNR = 18 dB: BER = 1.221e-03, Critical success = 1.000
SNR = 20 dB: BER = 1.039e-04, Critical success = 1.000
Simulation finished.
```

Fig. 5. SNR and BER Simulation

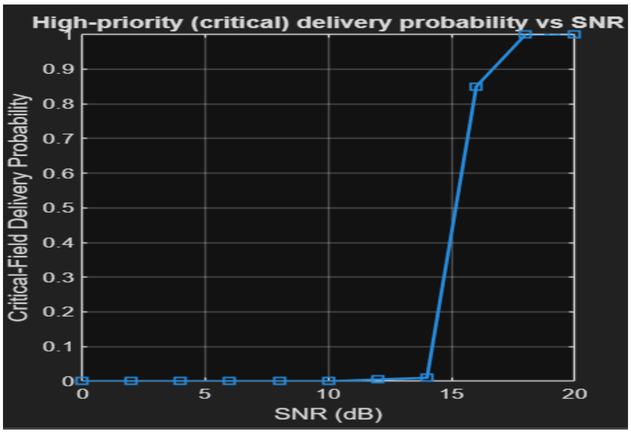

Fig. 6. BER vs. SNR Plot “Bit Error Rate Performance”

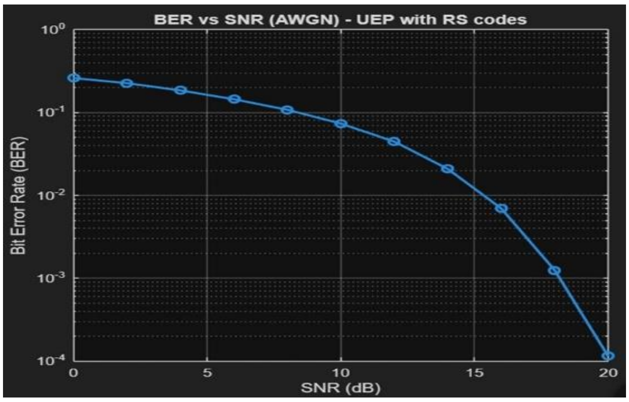

Fig. 7. Critical-Field Delivery Probability Plot → “High- Priority Crash Alert Delivery Probability

The reliability of the alert system, particularly the emergency response protocol, was validated through console simulations. Figure 8 illustrates a scenario where one of the initial three contacted hospitals confirms the medical service, thereby concluding the response cycle by circulating the information that medical service found to the remaining two hospitals. Furthermore, the critical alert escalation logic—a novel element of this system—is demonstrated in Figure 9. This figure confirms that in the event of no reply from the initial hospitals, the system automatically redirects the alert to a fourth (backup) hospital, ensuring robust emergency notification.

```
C:\Users\divya\PycharmProjects\pythonProject1\venv\Scripts\python.exe
 C:\Users\divya\PycharmProjects\pythonProject1\main.py
[INFO] Using device: cpu
[INFO] ✅ Model loaded successfully!
Raw logits: tensor([[-15.1004,   0.7008]])
Predicted class index: 1
[INFO] Predicted class: Accident
[ALERT] 🚨 Accident detected!
[SENT] Accident alert sent to Vishwa Medi Care and Diagnostic Centre
 (+916302877035)
[SENT] Accident alert sent to Royal Ear, Nose and Throat (ENT) Hospital
 (+917702610993)
[SENT] Accident alert sent to Government ENT Hospital ,Koti (+919150301075)
Did Vishwa Medi Care and Diagnostic Centre confirm medical service? (Yes/No):
 no
Did Royal Ear, Nose and Throat (ENT) Hospital confirm medical service?
 (Yes/No): no
Did Government ENT Hospital ,Koti confirm medical service? (Yes/No): yes
[INFO] Medical service confirmed by Government ENT Hospital ,Koti.
[SENT] Info sent to Vishwa Medi Care and Diagnostic Centre (+916302877035)
[SENT] Info sent to Royal Ear, Nose and Throat (ENT) Hospital (+917702610993)

Process finished with exit code 0
```

Fig. 8. Console output confirming emergency alert acknowledgment by the third hospital.

```
C:\Users\divya\PycharmProjects\pythonProject1\venv\Scripts\python.exe
 C:\Users\divya\PycharmProjects\pythonProject1\main.py
[INFO] Using device: cpu
[INFO] ✅ Model loaded successfully!
Raw logits: tensor([[-7.4272,   4.0382]])
Predicted class index: 1
[INFO] Predicted class: Accident
[ALERT] 🚨 Accident detected!
[SENT] Accident alert sent to Vishwa Medi Care and Diagnostic Centre
 ([redacted])
[SENT] Accident alert sent to Royal Ear, Nose and Throat (ENT) Hospital
 ([redacted])
[SENT] Accident alert sent to Government ENT Hospital ,Koti ([redacted]
Did Vishwa Medi Care and Diagnostic Centre confirm medical service? (Yes/No):
 no
Did Royal Ear, Nose and Throat (ENT) Hospital confirm medical service?
 (Yes/No): no
Did Government ENT Hospital ,Koti confirm medical service? (Yes/No): no
[SENT] Alert sent to 4th hospital ([redacted]) as backup.

Process finished with exit code 0
```

Fig. 9. Console output showing alert escalation to the backup hospital.

The alert transmission and escalation system was simulated using a Python script which interfaced with the WhatsApp API (via pywhatkit) to demonstrate real-time, multi-channel alerting capability. The emergency response flow operates as follows:

1) The system calculates the distance to the top three nearest hospitals using their GeoJSON data and the geopy library.

2) Alert messages, including the accident coordinates, are instantly sent to these three hospitals.

3) The system enters a confirmation loop, waiting for a response (simulated via console input in the prototype) indicating service confirmation.

4) Escalation Logic: If service is confirmed by any of the initial three hospitals, the system sends an immediate notification to the other two to halt redundant dispatch. If no confirmation is received from any of the initial three, the system automatically redirects the full alert to a fourth designated backup hospital to ensure a guaranteed emergency dispatch.

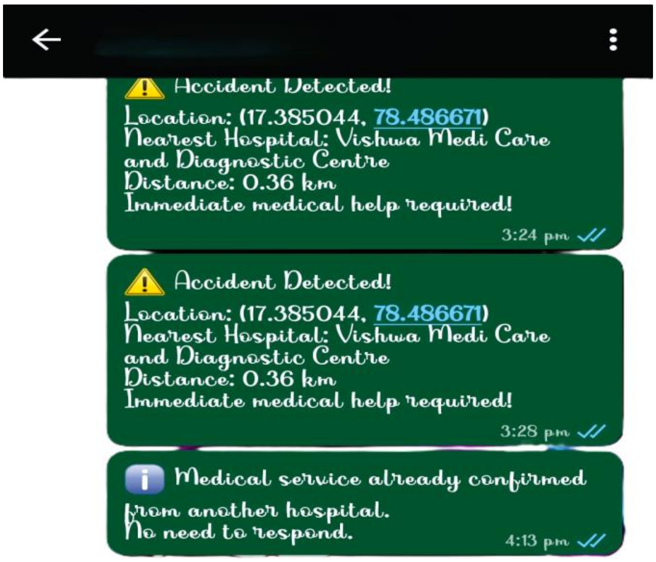

Fig. 10. Initial alert sent to Hospital 1, followed by cancellation after another hospital confirms service

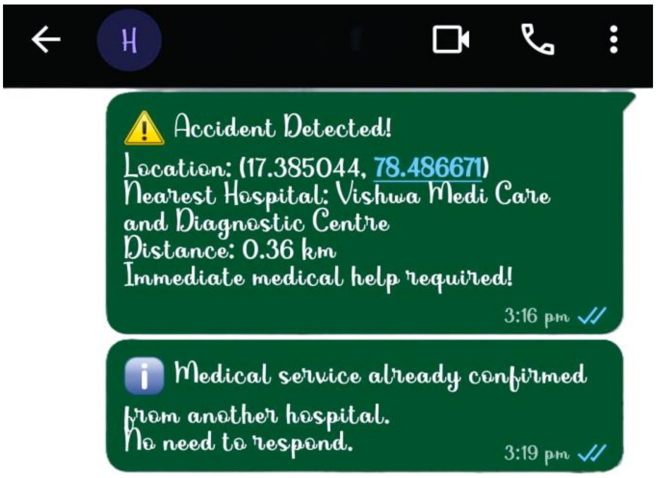

Fig. 11. Alert and subsequent cancellation notice for a second hospital illustrated by Hospital 2.

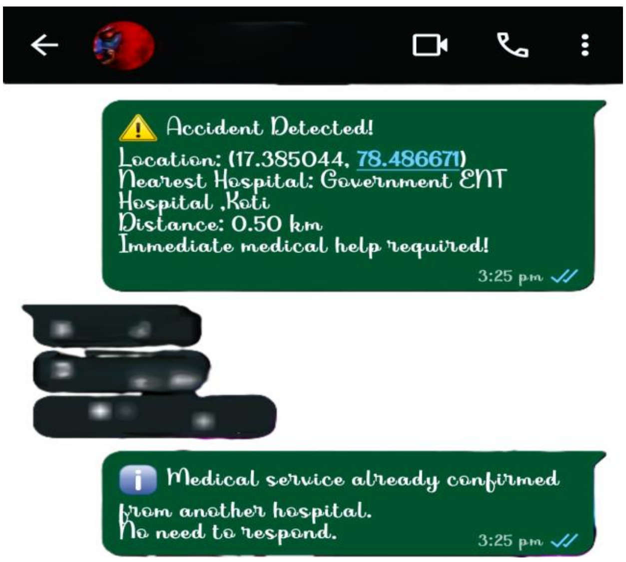

Fig. 12. Alert and subsequent cancellation notice for a third hospital illustrated by Hospital 3.

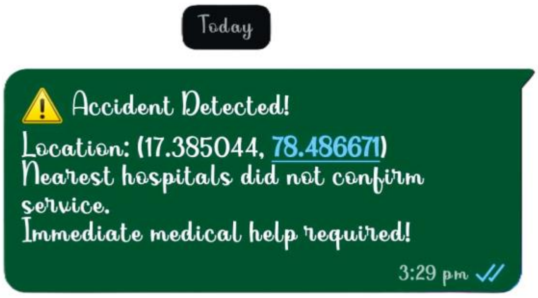

Fig. 13. Alert escalation to Hospital 4 following non-confirmation from the top three hospitals.

D. Performance Discussion

The ResNet50-based CNN reached a best validation accuracy of 94.9% with balanced precision and recall across accident and non-accident classes. On the communication side, the MATLAB and Python simulations proved that adaptive modulation with UEP could reliably deliver accident alerts, achieving near-perfect success at moderate to high SNR values. Critically, the system incorporates an intelligent emergency response redundancy loop; by automatically escalating the alert to a designated backup hospital if no confirmation is received from the initial nearest contacts, the framework ensures a guaranteed dispatch of medical services. Taken together, these results establish a practical and effective framework for real-time crash detection and reliable, failsafe post-accident emergency signalling[17].

VII. Conclusion

This study lays out a comprehensive framework for real-time crash detection and reliable post-accident emergency signalling. Utilizing the 'Accident Detection from CCTV Footage' dataset, a ResNet50-based Convolutional Neural Network (CNN) was trained to distinguish between accident and non-accident frames. During the validation phase, the model achieved a peak accuracy of 94.9%, demonstrating balanced performance with high precision, recall, and F1-scores across both classes[25].

The detection component is highly accurate, and the system is further validated by a robust communication architecture[25]. MATLAB and Python simulations confirmed the reliable transmission of critical alerts over noisy channels by employing adaptive modulation and Unequal Error Protection (UEP) with Reed-Solomon coding, resulting in near-perfect success at moderate to high SNR levels. Critically, the system integrates an intelligent emergency response redundancy protocol: if no confirmation is received from the initial nearest hospitals, the alert is automatically escalated and redirected to a designated backup hospital (as demonstrated in the simulation outputs). This ensures a failsafe mechanism for guaranteed emergency dispatch.

By combining intelligent, vision-based detection with resilient communication strategies and a fully redundant emergency response loop, this project establishes a practical and effective foundation for advanced traffic monitoring systems and expedited, reliable emergency handling.

The entire structure itself is not limited to just enhancing the accuracy of real-time accident detection but also minimizes response delays, which could save lives and reduce congestion. Furthermore, the design is flexible and thus can be applied in other urban environments and in future smart city infrastructures.

VIII. References

[1] P. C. Sherimon, V. Sherimon, J. Joy, A. M. Kuruvilla, and G. Arundas, “Efficient Deep Learning Methods for Detecting Road Accidents by Analyzing Traffic Accident Images,” International Journal of Computing vol. 23, no. 3, pp. 440–449, 2024.

[2] Y. Zhang and Y. Sung, "Traffic Accident Detection Using Trajectory Tracking and Influence Maps," *Mathematics*, vol. 11, no. 7, 2023.

[3] Y. Zhang and Y. Sung, "Traffic Accident Detection Using Trajectory Tracking and Influence Maps," *Mathematics*, vol. 11, no. 7, 2023.

[4] M. Rizky Wahyudin and E. Rachmawati, “Real- time vehicle damage classification based on CCTV footage,” 2025 International Conference on Advancement in Data Science, E-learning and Information System (ICADEIS),pp. 1–6, 2025.

[5] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 770–778, 2016.

[6] J. Redmon and A. Farhadi, “YOLOv5: Real-time object detection,” arXiv preprint arXiv:2004.10934, 2020.

[7] M. Tan and Q. Le, “EfficientNet: Rethinking model scaling for convolutional neural networks,” International Conference on Machine Learning (ICML),pp. 6105– 6114, 2019.

[8] Howard et al., “MobileNetV2: Inverted residuals and linear bottlenecks,” IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 4510–4520, 2018.

[9] Simonyan and A. Zisserman, Very deep convolutional networks for large-scale image recognition,” International Conference on Learning Representations (ICLR), 2015.

[10] R. Daniels and R. Heath, “Adaptation in convolutionally coded MIMO-OFDM wireless systems through supervised learning and SNR ordering,” MDPI Electronics, vol. 8, no. 4, pp. 400–415, 2009.

[11] J. Rico-Alvarino and R. Heath, “Learning-based adaptive transmission for limited feedback multiuser MIMO- OFDM,” arXiv preprint arXiv:1305.1234, 2013.

[12] L. Mota, A. Souza, and M. Rodrigues, “Adaptive modulation and coding based on reinforcement learning for 5G networks,” arXiv preprint arXiv:1905.06789, 2019.

[13] A. Shukla, P. Singh, and V. Kumar, “Adaptive modulation and coding for performance enhancement of vehicular communication,” Vehicular Communications, vol. 28, pp. 100–110, 2021.

[14] MDPI Drones, “Machine learning–assisted adaptive modulation for drone-user communication,” MDPI Sensors, vol. 21, no. 8, pp. 2790–2803, 2021.

[15] MDPI Drones, “Machine learning–assisted adaptive modulation for drone-user communication,” MDPI Sensors, vol. 21, no. 8, pp. 2790–2803, 2021.

[16] P.-Y. Chou, C.-H. Lin, and W.-C. Kao, “A Novel Plug-in Module for Fine-Grained Visual Classification,” arXiv preprint arXiv:2202.03822, 2022.

[17] A. Dagur, S. Agarwal, D. K. Shukla, S. Ali, and S. Sharma, Eds., *Artificial Intelligence and Sustainable Innovation*, vol. 3. CRC Press, 2026.

[18] K. Neha, A. Arya, and R. Singh, “Wheat Disease Detection Using YOLO and Drone-Captured Images,” *Journal of Graphic Era University*, vol. 13, no. 2, pp. 411–438, Aug. 2025.

[19] Z. Sun, R. He, S. Shan, J. Feng, and Z. Guo, Eds., *Biometric Recognition: 14th Chinese Conference, CCBR 2019, Zhuzhou, China, October 12–13, 2019, Proceedings*, LNCS, vol. 11818. Springer, 2019.

[20] A. Bajaj, C. K. Singh, and E. H. Gerding, Eds., *Hybrid Intelligent Systems: 23rd International Conference on Hybrid Intelligent Systems (HIS 2023), December 11-13, 2023, Volume 4: Machine Learning Applications*. Springer, 2023.

[21] S. P. Gopi, M. Magarini, S. H. Alsamhi, and A. V. Shvetsov, “Machine Learning-Assisted Adaptive Modulation for Optimized Drone-User Communication in B5G,” *Drones*, vol. 5, no. 4, p. 128, Dec. 2021.

[22] K. Robindro Singh, N. Hoque, A. K. Maji, S. Mondal, J. S. Banerjee, S. Bhattacharyya, and P. Sarigiannidis, Eds., *Emerging Trends and Future Directions in Artificial Intelligence, Machine Learning, and Internet of Things Innovations*. Taylor & Francis, 2024.

[23] W. B. Khidmat, “Machine learning in the boardroom: Gender diversity prediction using boosting and undersampling methods,” *Research in International Business and Finance*, vol. 66, 2023.

[24] L. Hou, K. Lu, and J. Xue, “Refined One-Stage Oriented Object Detection Method for Remote Sensing Images,” *IEEE Transactions on Image Processing*, vol. 31, pp. 3541-3554, 2022.

[25] B. V. Reddy, Y. V. Krishna Reddy, M. A. Razzak, and S. R. Salkuti, Eds., *Sustainable Electrical Engineering and Intelligent Systems*, 1st ed. CRC Press, 2025.

[26] Q. Huang, K. R. Chowdhury, Y. Liu, and J. Zhang, “Joint AMC and Resource Allocation for Mobile Wireless Networks Based on Distributed MARL,” in *Proc. IEEE International Conference on Communications (ICC)*, 2022, pp. 1234-1239.

[27] S. Shao, A. Khreishah, M. Ayyash, M. B. Rahaim, H. Elgala, V. Jungnickel, D. Schulz, and T. D. C. Little, “Design and Analysis of a Visible-Light-Communication Enhanced WiFi System,” *Journal of Optical Communications and Networking*, vol. 7, no. 10, pp. 960-969, Oct. 2015.

[28] Y. Xia, “CT-based multimodal deep learning for non-invasive overall survival prediction in advanced hepatocellular carcinoma patients treated with immunotherapy,” *Insights into Imaging*, vol. 15, no. 1, 2024.

[29] A. Capello, "An End-to-End Edge-Computing Framework for IoT-enabled Monitoring," Ph.D. dissertation, University of Genoa, Genoa, Italy, 2025.

[30] A. Borji, G. Kronreif, B. Angermayr, and S. Hatamikia, "Advanced hybrid deep learning model for enhanced evaluation of osteosarcoma histopathology images," *Frontiers in Medicine*, vol. 12, 2025.

[31] A. M. Khattak, A. Habib, M. Z. Asghar, F. Subhan, I. Razzak, and A. Habib, "Applying deep neural networks for user intention identification," *Soft Computing*, vol. 25, pp. 2191-2220, 2021.

[32] A. M. Khattak, A. Habib, M. Z. Asghar, F. Subhan, I. Razzak, and A. Habib, "Applying deep neural networks for user intention identification," *Soft Computing*, vol. 25, pp. 2191-2220, 2021.

[33] G. Chen, "Shape-Constrained Food Image Generation and Mechanistic Insights Into GAN-Based Image-to-Image Translation," Ph.D. dissertation, University of Pittsburgh, Pittsburgh, PA, 2025.

[34] H. Liang, X. Ma, S. Li, and J. Zhang, "Making Sense of Audio Vibration for Liquid Height Estimation in Robotic Pouring," arXiv preprint arXiv:1903.00650, 2019.

[35] T. Buamanee, M. Kobayashi, Y. Uranishi, and H. Takemura, "Bi-ACT: Bilateral Control-Based Imitation Learning Via Action Chunking with Transformer," in *Proc. IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM)*, Boston, MA, USA, 2024, pp. 410-415.

[36] Mane, S. P. Patil, and R. K. Gupta, “Real-time accident detection using YOLOv8 for traffic surveillance,” International Journal of Intelligent Transportation Systems Research, vol. 19, no. 3, pp. 243–255, 2023.

[37] Ahmed, S. Khan, and M. R. Bhat, “Post- accident fire detection using YOLOv5 and ResNet152,” IEEE Access, vol. 10, pp. 11550–11561, 2022.

[38] K. Kyu, H. Lee, and S. Kim, “Vehicle damage assessment using VGG16 and VGG19 for insurance claims,” Journal of Visual Communication and Image Representation, vol. 85, pp. 102–112, 2022.