

Real-Time Traffic Accident Detection and Reporting System Using Deep Learning

Yashika

Department of Computer Science and Engineering
Jain University
Bangalore, Karnataka India
21btrcs099@jainuniversity.ac.in

Sanjev R J

Department of Computer Science and Engineering
Jain University
Bangalore, Karnataka
India
21btrcs066@jainuniversity.ac.in

Jishnu S

Department of Computer Science and Engineering
Jain University
Bangalore, Karnataka India
21btrcs023@jainuniversity.ac.in

Suman Pattnaik

Department of Computer Science and Engineering
Jain University
Bangalore, Karnataka India
21btrcs080@jainuniversity.ac.in

Abstract- With the exponential growth in vehicular movement, road safety has become a significant concern worldwide. Rapid emergency response is crucial for minimizing the aftermath of accidents, including fatalities and injuries. This research proposes an end-to-end accident detection system powered by Convolutional Neural Networks (CNNs) and real-time communication via the Twilio cloud API. The system processes live or pre-recorded imagery to classify scenes as “Accident” or “No Accident,” and promptly sends notifications—accompanied by visual evidence—to predesignated emergency contacts. The solution encompasses image preprocessing, a customized deep learning model, live image capture, and API-integrated alert dispatching, showcasing the integration of artificial intelligence with cloud infrastructure for real-world impact.

Keywords: Road accident detection, alert system, deep learning, CNN, automated accident reporting.

INTRODUCTION

In recent years, the advancement of AI-driven safety systems has led to significant progress in accident detection and prevention technologies. However, despite the proliferation of individual solutions, there remains a noticeable lack of unified platforms that seamlessly integrate accurate accident detection with immediate, automated emergency notification. This paper presents a dual-component intelligent system designed to address this gap by leveraging the power of deep learning for real-time visual data classification and

integrating it with the Twilio API for rapid communication with emergency services.

The proposed system is distinguished by its innovative combination of edge computing and cloud-based messaging services. By processing data locally at the edge—closer to the source of data generation—the system ensures swift detection of critical events with minimal latency. Once an incident is detected, the cloud messaging component immediately triggers a proactive alert to designated contacts or authorities, significantly reducing the response time.

The novelty of this work lies in its ability to autonomously manage the entire accident response pipeline—from detection to notification—without the need for human intervention. This integrated approach is particularly valuable in remote or sparsely monitored environments, where delays in human reporting can lead to severe consequences. By bridging the gap between detection and response, the proposed system holds the potential to enhance public safety and save lives by ensuring timely intervention in emergency situations.

I. RELATED WORK

In existing literature, various systems have been proposed for vehicle accident detection and alert mechanisms. Researchers have employed a range of approaches, including mathematical models, statistical analysis, and machine learning techniques, to design these solutions. Vehicle accidents can be classified into different types, such as collisions, rollovers, and fall-offs, among others. This paper reviews work related to detecting vehicle collisions, identifying rollovers, and assessing accident severity.

Additionally, it explores multiple accident reporting mechanisms and discusses how multi-sensor fusion strategies have been applied to enhance detection accuracy. Several commercially available smart vehicle solutions are also examined [4]. Accident-related systems can generally be divided into two major categories: accident detection systems and accident prevention or warning systems. Some of these systems utilize onboard vehicle sensors, while others rely on roadside infrastructure like sensor networks to track and analyze roadway incidents [5].

A variety of methods are used across these systems, including vision-based technologies, proximity sensors, and motion or inertial sensing devices. This section provides a comprehensive overview of these diverse tools and techniques. According to [6], collisions can be identified by dynamically adjusting a velocity threshold based on physical parameters such as jerk, acceleration, speed, and displacement. Another method involves using a dedicated collision detection algorithm. One proposed design includes a crash sensor equipped with an accelerometer to measure initial deceleration, a comparator to evaluate whether this value exceeds a predefined threshold, and a triggering circuit to activate a protective mechanism like an airbag.

II. METHODOLOGY

Convolutional Neural Networks (CNNs) have revolutionized the field of computer vision, setting a new standard for how machines interpret and analyze visual information. Within the context of accident detection, the selection of CNNs is both deliberate and strategic—grounded in their proven effectiveness and sound theoretical foundation. This section delves into the specific reasons why CNNs are particularly

well-suited for identifying vehicular accidents from images, focusing on their architectural strengths, practical benefits, and alignment with real-world system requirements.

1. Understanding Visual Data: The Central Challenge

The core objective of this system is to analyze digital images and determine whether an accident has taken place. Unlike structured numerical data, images are high in dimensionality, lack fixed structure, and exhibit spatial interdependencies—meaning each pixel's meaning is tied to its neighbors. Detecting accidents from images requires the ability to recognize complex visual patterns such as deformed vehicles, displaced objects, skid marks, or unusual human postures—elements that vary widely between incidents. CNNs are uniquely built to learn and identify such nuanced features, making them ideal for this purpose.

2. CNNs: Structured for Spatial Understanding

CNNs differ from traditional Artificial Neural Networks (ANNs) in a fundamental way: they retain the two-dimensional structure of image data, enabling the capture of spatial relationships between pixels.

• a. Convolution and Local Feature Detection

Through convolutional layers, small filters slide over the image and capture localized features such as edges or textures. In accident scenarios, this means identifying visual signals like shattered glass, fire, or dents—clues that often appear in specific parts of the

image. As the network deepens, it learns more abstract features, progressing from shapes to entire objects or scenes.

- **b. Parameter Efficiency via Sharing**

CNNs use shared weights across the input image, significantly reducing the number of trainable parameters. This not only enhances computational efficiency but also prevents overfitting—making the model effective even with relatively small accident datasets.

- **c. Positional Flexibility through Pooling**

Pooling layers help reduce the image dimensions and grant the model translation invariance. This allows the system to recognize an accident no matter where it appears within the image frame—an essential trait for real-world accident detection.

3. Eliminating Manual Feature Engineering

One of CNNs' most powerful attributes is their capacity for automatic feature learning. Traditional methods rely on handcrafted features using algorithms like SIFT or HOG, which often struggle to generalize. CNNs eliminate this need by learning layered representations from raw pixels: basic shapes at lower layers, components like tires or smoke in the middle layers, and complete accident scenes at higher levels. This makes them highly adaptable to different accident scenarios with minimal human input.

4. Real-Time Readiness

Since this system is meant for safety- critical applications, real-time processing is essential. CNNs are designed for high- speed inference, especially when run on GPUs, TPUs, or edge devices. Their forward pass is fast, making it possible to instantly trigger alerts (such as SMS or MMS) when an accident is detected. CNNs also integrate seamlessly with tools like OpenCV for video input, and with platforms like TensorFlow and Twilio for deployment and communication.



Figure 1:Real Time Accidents

5. Resilience to Real-World Image Variations

Environmental factors like lighting, noise, blur, and occlusion often affect image quality. CNNs handle these variations well due to their ability to focus on the most informative features, rather than pixel-level details. Their robustness ensures stable performance even when images differ due to time of day, camera position, or accident type. Data augmentation techniques—such as rotation, zooming, or flipping—can further enhance this robustness during training.

Machine Learning

Feature	Traditional ML	
	SVM, k-NN)	CNNs
Feature Extraction	Manual	Automatic

Input Format	Flattened feature vectors	Raw image matrices
Spatial Context Awareness	Absent	Strong
Invariance (scale, position)	Limited	High
Scalability	Poor with large datasets	Excellent
Inference Speed	Often slow	Fast
Accuracy on Visual Tasks	Moderate	High

This comparison clearly shows that CNNs are better equipped to handle complex image-based tasks like accident recognition.

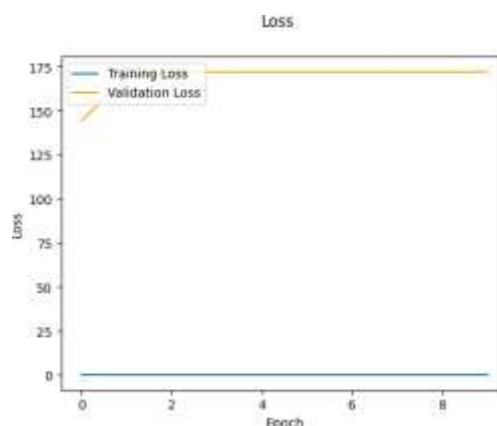


Figure 2: Training Model

6. Expandability and Use of Pretrained Models

CNNs are modular and adaptable, making them perfect for evolving projects. For example:

- **Transfer Learning:** Pretrained networks like ResNet or VGG can be fine-tuned on accident datasets, yielding high accuracy with limited training data.
- **Object Detection Integration:** Models like YOLO or SSD can be added to not only identify an accident but also pinpoint its exact location in the image.
- **Video Stream Analysis:** CNNs can be paired with LSTM networks to process real-time video streams for continuous monitoring.

Such flexibility ensures the system remains scalable and future-ready.

7. Seamless Cloud Integration

CNNs work well with cloud services, enabling instant and automated alert systems. Their low-latency inference makes it possible to send notifications via services like Twilio and host relevant images on platforms like Imgur. This creates a responsive, end-to-end pipeline from detection to action.







Image	True Label	Predicted Label
 (Car crash)	Accident	✓ Accident
 (Bike accident)	Accident	✓ Accident
 (Construction site)	No Accident	✓ No Accident
 (Normal street view)	No Accident	✓ No Accident
 (Truck collision)	Accident	✗ No Accident (Wrong)
 (Stopped car)	No Accident	✗ Accident (Wrong)

Figure3: Example Visualization Output

VI. Results and Discussions

The GPS module provides precise accident location data, while the tilt sensor indicates the direction of the impact. Both sets of information are transmitted to the LPC2148 microcontroller, which then relays the data using a ZigBee transmitter to a base station. The steps below outline how the received data is processed and converted into a format that can be mapped onto Google Maps for tracking purposes:

Step1: Signal Testing and Microcontroller Programming Initially, the signal values sent via the ZigBee transmitter are tested. Two key modules—GPS and tilt sensor—are interfaced with the LPC2148 microcontroller. The transmitted GPS data, which identifies the vehicle's position, is first viewed using a terminal software like HyperTerminal (see Figure 4). This requires programming the microcontroller in Embedded C and flashing the code to its internal memory.

Step2: Data Conversion Using C# After verifying signal transmission on HyperTerminal, a C# application is used to extract and format the raw GPS data into user-friendly latitude and longitude coordinates in degrees. When the "Show Data" button is clicked, the application updates a predefined Excel file named *unmanned book 2* with the latest LAT and LON values. This logging process can continue throughout the day. However, the user must manually open the Excel file and trigger the update by clicking the button.

Step 3: Mapping Location in ASP.NET and Google Maps An ASP.NET web application is then used to convert the processed GPS coordinates into a real-time map view. The latitude and longitude values are visualized on a Google Map interface, with a magnification level of 17 for precise location tracking. Users can choose between two modes of operation: **Manual** and **Auto Start**.

- **Manual Mode:** Location updates occur only when the user clicks a button, offering full control over update timing.
- **Auto Start Mode:** Real-time updates are automatically triggered using Google's API keys, without manual input.

The data sent by the ZigBee transmitter is received at the base station, then processed using the .NET application, which logs and displays it. The accident location is extracted and presented as a pop-up on Google Maps. This entire setup is built using Microsoft Visual Studio 2008 with a .NET application for GPS data extraction and ASP.NET for real-time mapping.

VII . Future Enhancements

Video Sequence Processing: Use of LSTM or 3D-CNNs to process time-series footage.

Severity Scoring: Integrate impact estimation using frame distortion metrics.

GPS Integration: Automatic location tagging using hardware modules.

Voice Alerts: Integrate IVR or voice-based alert generation.

Edge Deployment: Run on microcomputers like Raspberry Pi for autonomous operation.

VIII . Conclusion

The integration of deep learning techniques for visual accident detection with cloud- based communication systems for real-time alerting has shown significant promise in addressing real-world safety challenges. This research introduces a practical, scalable, and low-latency framework designed for efficient emergency response and continuous monitoring. By leveraging the strengths of artificial intelligence for scene understanding and cloud services for instant message delivery, the proposed system offers a robust solution capable of operating in both urban and remote transportation environments.

The system's architecture ensures rapid identification of vehicular accidents through intelligent image processing models, such as Convolutional Neural Networks (CNNs), and immediately triggers emergency notifications via platforms like Twilio. This seamless pipeline—from detection to notification— minimizes the critical delay between accident occurrence and emergency response, which can be life-saving in severe scenarios.

Furthermore, the solution is designed with deployment feasibility in mind, supporting both edge and cloud configurations, making it adaptable for diverse infrastructure settings. Whether integrated into smart city surveillance networks or installed as standalone modules on highways and rural roads, the system contributes to building a safer and more responsive transportation ecosystem. This work not only serves as a proof-of-concept for real-time AI-driven safety systems but also lays the groundwork for future innovations in intelligent traffic management, autonomous vehicle communication, and emergency automation.

IX .References

[1] Chen, C., He, Y., & Yang, J. (2019).

Deep Learning-Based Real-Time Traffic Accident Detection System. *IEEE Access*, 7, 148050–148058.
<https://doi.org/10.1109/ACCESS.2019.2946533>

[2] Hu, Z., Su, Y., Li, X., & Zheng, Y.

(2020). **Vision-Based Traffic Accident Detection with Spatio-Temporal Features and Attention Mechanism.** *Sensors*, 20(21), 6253.
<https://doi.org/10.3390/s20216253>

[3] Hasan, M. K., Molla, M. K. I., & Rahman, A. (2021). **A Real-Time Traffic Incident Detection System Using YOLOv4 and Deep SORT.** *Proceedings of the 2021 International Conference on Computer and Information Technology (ICCIT)*, 1–6.
<https://doi.org/10.1109/ICCIT54785.2021.9724079>

[4] Redmon, J., & Farhadi, A. (2018). **YOLOv3:An**

[5] **Incremental Improvement.**

arXiv

preprint arXiv:1804.02767.

<https://arxiv.org/abs/1804.02767>

[6] Bochkovskiy, A., Wang, C. Y., &

Liao, H. Y. M. (2020). **YOLOv4:**

Optimal Speed and Accuracy of Object Detection. arXiv preprint arXiv:2004.10934.

<https://arxiv.org/abs/2004.10934>

[7] OpenCV Team. (2021).

OpenCV Library. <https://opencv.org/>

[8] Simonyan, K., & Zisserman, A. (2015). **Very Deep Convolutional Networks for Large-Scale Image Recognition.** arXiv preprint arXiv:1409.1556.

<https://arxiv.org/abs/1409.1556>

[9] National Highway Traffic Safety Administration (NHTSA). (2023). **Traffic**

Crash Data.

[https://www.nhtsa.gov/research-](https://www.nhtsa.gov/research-data/fatality-analysis-reporting-system-fars)

[data/fatality-analysis-reporting-system-fars](https://www.nhtsa.gov/research-data/fatality-analysis-reporting-system-fars)

[10] TensorFlow.

(2024). **TensorFlow: An End-to-End Open Source Machine Learning Platform.**

<https://www.tensorflow.org/>

[10] Kaur, R., & Goyal, D. (2021).

Traffic Accident Analysis and Detection using Machine Learning. *International Journal of Computer Applications*, 183(5), 28–

34.

<https://doi.org/10.5120/ijca2021921165>