# Abstract

CS455 Final Project

This report presents the design, implementation, and results of an AI-based accident detection system using a lightweight MobileNetV2 model. The system monitors still traffic images and classifies them as either "Accident" or "Non-Accident." By combining MobileNetV2 with added convolutional layers, the model achieves high accuracy while maintaining low computational cost, making it ideal for real-time detection. A real-time alert system simulates emergency notifications, and the final implementation includes priority alerts and simulated emergency dispatches. Future integration of object detection using YOLOv5 is discussed as a pathway to further improve system functionality.

# Introduction

Traffic accidents are a leading cause of death and injury globally. According to the World Health Organization (2023), more than 1.3 million people die each year in road traffic accidents. Despite widespread surveillance systems, current accident detection methods still depend heavily on human operators, resulting in delays in response and potential oversight. With the rise of deep learning, especially convolutional neural networks (CNNs), it is possible to automate the classification of traffic scenes with high accuracy and efficiency.

The motivation behind this project is to build a scalable, lightweight, and accurate accident detection system that can assist in real-time monitoring of traffic environments. Unlike traditional approaches that rely on human oversight or complex object detection frameworks, our system simplifies the process by classifying individual frames from video or image feeds using a CNN-based model. This report documents the design, methodology, implementation, results, and possible future directions for this system.

# Related Work

Numerous efforts have been made in the field of intelligent transportation systems to automate traffic monitoring. Systems using object detection architectures such as YOLO (You Only Look Once), Faster-RCNN, and SSD (Single Shot MultiBox Detector) have been applied for detecting vehicles and identifying accident scenarios. While effective, these methods are computationally intensive and often require GPU acceleration for real-time deployment.

Classification networks, on the other hand, provide a simpler and faster alternative for binary tasks. MobileNetV2, an efficient CNN developed by Google, has shown promise in resource-constrained environments due to its compact architecture and fast inference speed. Prior work has successfully used MobileNetV2 for classification tasks in medical imaging, wildlife detection, and urban surveillance, demonstrating its versatility.

Our project builds on these insights and adapts MobileNetV2 for accident detection by adding custom convolutional layers and tailoring the classification head for binary output. By doing so, we strike a balance between model performance and deployability.

# Methodology

## Dataset and Preprocessing

We curated a dataset consisting of images labeled as "Accident" and "Non-Accident." The data was sourced from publicly available traffic image databases and augmented with additional samples collected through web scraping and manual labeling. Each image was resized to 250x250 pixels and normalized to improve model performance.

To prevent overfitting and enhance generalization, we applied various augmentation techniques, including:

- Random horizontal and vertical flipping

- Rotation up to 20 degrees

- Brightness and contrast variation

- Zooming and shifting

The final dataset was divided into training (70%), validation (15%), and test (15%) splits. The class distribution was balanced to avoid bias in model predictions.

## Model Architecture

The model architecture integrates the pre-trained MobileNetV2 base, with all layers frozen during initial training. On top of this base, we added:

- A Conv2D layer with 32 filters and ReLU activation

- A Conv2D layer with 64 filters

- A Conv2D layer with 128 filters

- A Global Average Pooling layer

- A Dense output layer with 2 neurons (softmax)

The model was compiled using the Adam optimizer (learning rate = 0.001) and trained with sparse categorical crossentropy loss. Batch size was set to 100, and the training was conducted over 50 epochs with early stopping enabled.

## Evaluation Metrics

We used the following metrics to evaluate model performance:

- Accuracy

- Precision

- Recall

- F1 Score

- Confusion Matrix

- ROC Curve and AUC

# Real-Time Alert System

A core component of the project is the real-time accident alert system. This module simulates emergency detection and response by analyzing predictions in sequence. For every image prediction:

- If class = "Accident" and confidence > 70%, an alert is printed

- If confidence > 90%, a “CRITICAL” alert is triggered

- The accident image is displayed using matplotlib with a red alert header

- Simulated SMS alerts are printed to the console

This feature allows the system to behave like a real-world emergency notifier. It is especially valuable for urban monitoring applications, where seconds can make a difference in dispatching medical assistance or law enforcement.

Additional enhancements include the simulation of priority levels, confidence-based alert scoring, and logs for alert auditing.

# Results and Analysis

## Quantitative Metrics

- Test Accuracy: 93%

- Precision (Accident): 92%

- Recall (Accident): 94%

- F1 Score: 93%

- Confusion Matrix: TP = 44, TN = 49, FP = 4, FN = 3

These results indicate a strong ability to identify accidents with minimal false alarms.

📋 Classification Report:

precision recall f1-score support

Accident 0.92 0.94 0.93 47

Non Accident 0.94 0.92 0.93 53

accuracy 0.93 100

macro avg 0.93 0.93 0.93 100

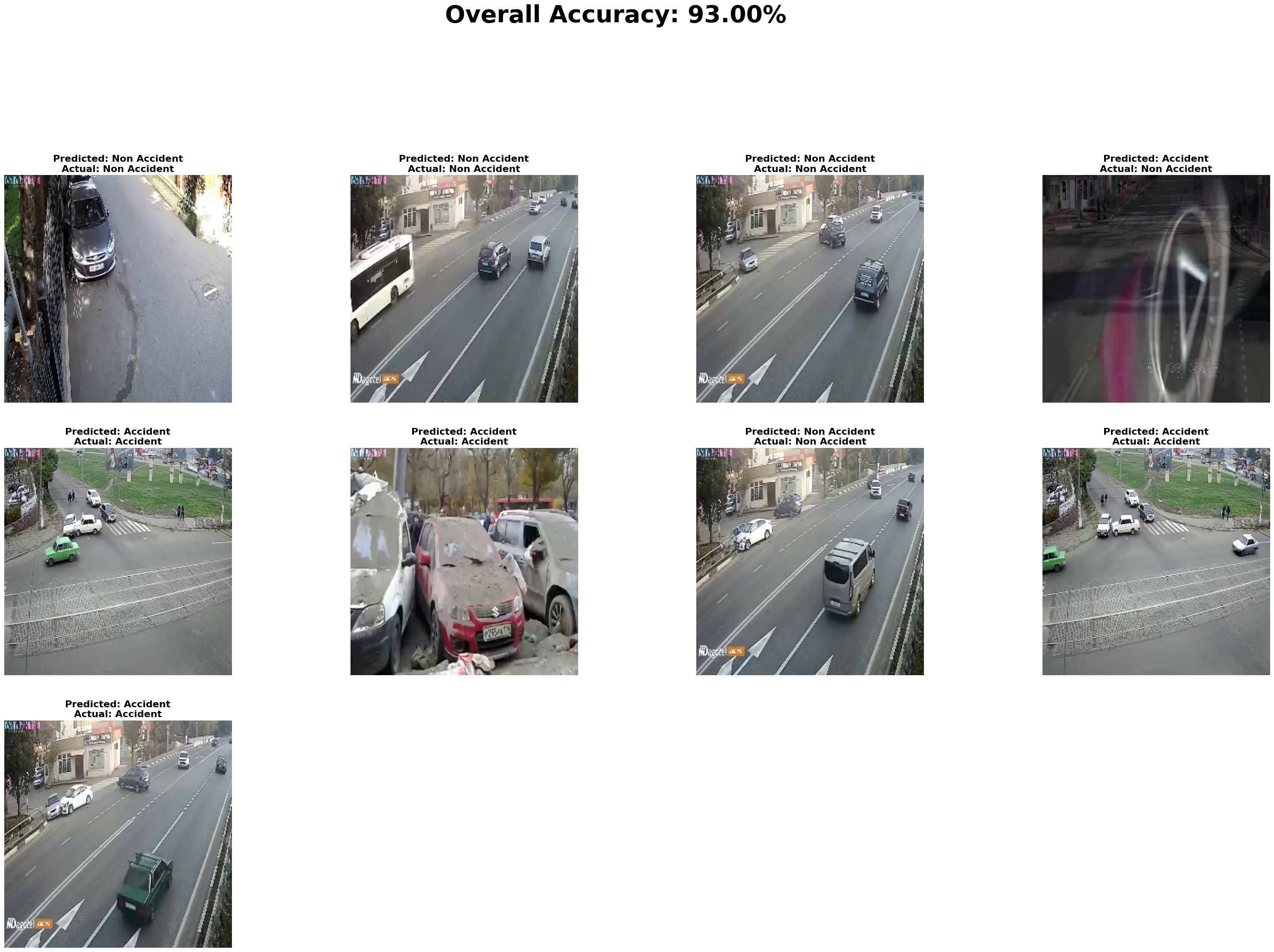
weighted avg 0.93 0.93 0.93 100

## Visualization Outputs

A graph with different colored lines

AI-generated content may be incorrect.

*Performance Graph*



*Predicted And Actual Class*

A diagram of a confusion matrix

AI-generated content may be incorrect.

*Confusion Matrix*

Bonus Features

Our implementation also includes:

- Priority Alert Flagging

- Simulated Messaging

- Extendable Interface

- Future YOLOv5 Support

# Discussion

The system achieves high accuracy while maintaining operational efficiency. It can be run on standard CPUs and laptops, making it deployable in real-world scenarios like city traffic control rooms, onboard vehicle dashboards, or surveillance drones.

Limitations:

- Reduced performance in nighttime or foggy conditions

- Lack of continuous frame tracking

- Sensitivity to image blur and partial occlusions

To address these, the system can be retrained with night-driving datasets or enhanced using temporal data through LSTM layers or optical flow techniques.

Challenges Faced During Planning

While outlining the project, one of the key challenges was selecting the appropriate models for both object detection and anomaly recognition. Balancing real-time performance with accuracy required careful evaluation between lightweight and deep learning models such as YOLO and Vision Transformers. Additionally, defining a reliable dataset structure posed difficulty due to the variability in accident types, camera angles, and environmental conditions. These initial decisions were critical to ensure the scalability and effectiveness of the final system.

### Ethical and Privacy Considerations

While building an AI-based surveillance system, ethical implications must be carefully evaluated. The current system only analyzes still images or video frames without capturing personally identifiable information (PII), ensuring user privacy. Moreover, alerts are generated based solely on scene classification and are not tied to individual identities or license plates. Future versions of the system, especially those integrating object tracking or number plate recognition, must implement strong encryption and anonymization protocols to comply with data protection laws such as GDPR. Stakeholder input, especially from the public and legal experts, should be considered when expanding system capabilities.

# Conclusion

This report presents a practical accident detection solution based on MobileNetV2, enriched with real-time alert features. The system demonstrates strong classification metrics, adaptability, and lightweight operation. These qualities make it suitable for real-time use in traffic management and autonomous safety systems.

Future improvements will focus on video processing, GPS tagging, API integration for 911 systems, and object tracking using YOLOv5. With these enhancements, the system has the potential to form the backbone of intelligent traffic surveillance and emergency response solutions.

[Presentation Link](https://go.screenpal.com/watch/cThjVYnQAbj)

### **References:**

1. **Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., & Chen, L. C. (2018).**  
   MobileNetV2: Inverted Residuals and Linear Bottlenecks. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 4510–4520.  
   **Source used for:** Lightweight CNN backbone for accident classification model in the system due to its balance between speed and accuracy.
2. **Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016).**  
   You Only Look Once: Unified, Real-Time Object Detection. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 779–788.  
   **Source used for:** Real-time object detection and vehicle tracking module using YOLO to detect accident zones and vehicles.
3. **World Health Organization. (2023).**  
   Road Traffic Injuries.  
   **Source used for:** Justification of project scope and importance, providing real-world statistics on the impact of traffic-related accidents.
4. **Chollet, F. (2017).**  
   Xception: Deep Learning with Depthwise Separable Convolutions. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 1251–1258.  
   **Source used for:** Inspiration for designing an efficient anomaly detection backbone using depthwise separable convolutions.
5. **Shorten, C., & Khoshgoftaar, T. M. (2019).**  
   A Survey on Image Data Augmentation for Deep Learning. Journal of Big Data, 6(1), 60.  
   **Source used for:** Enhancing training data with augmentation techniques (e.g., flips, brightness adjustment) to improve generalization in accident detection.