International Journal of
Computational Learning & Intelligence
Vol. 04 | Issue 04 | 2025

RESEARCH ARTICLE OPEN ACCESS

Zero Fatality Roads: Leveraging Deep and Transfer Learning on CCD for Automated Crash Detection

G Ramasubba Reddy . Sunil Jinkathoti . I Sravani . S Nareshkumar Reddy . M Prathap

Department of Computer Science & Engineering (AI&ML),
Sai Rajeswari Institute of Technology, Proddatur,
Andhra Pradesh, India.

DOI: **10.5281/zenodo.16607739**
Received: 12 June 2025 / Revised: 13 July 2025 / Accepted: 28 July 2025
©Milestone Research Publications, Part of CLOCKSS archiving

Abstract – Over a million people have died in road accidents every year, making them one of the world's top sources of injury and death. Human error and delayed incident response are major contributors to road deaths, even with improvements in vehicle design and road infrastructure. This study proposes a robust deep-learning framework for automated road accident detection by harnessing the synergy of transfer learning and car crash data. Addressing the limitations of scarce annotated accident imagery, we employed a publicly available Kaggle dataset called the Car Crash Dataset (CCD), which contains real dashcam videos of traffic accidents, mostly collected from YouTube, to train and evaluate state-of-the-art convolutional neural network models. Using ImageNet-pretrained architectures such as MobileNetV2, VGG19, ResNet50, and EfficientNetB7, we adapted the classification layers for binary classification while leveraging their pre-learned visual features. Our experimental results show that ResNet50 achieved the highest detection performance, with a precision of 97.83%, a recall of 98.78%, an F1-score of 98.05%, and an MCC of 0.955, outperforming the other models. MobileNetV2 also demonstrated strong results, with a precision of 92.84%, a recall of 94.60%, an F1-score of 93.71%, and an MCC of 0.781, making it suitable for deployment in resource-constrained environments. EfficientNetB7 and VGG19 performed moderately but lagged behind ResNet50 and MobileNetV2 in accuracy and correlation measures. This comparison offers essential guidance for choosing CNN architectures that balance accuracy and efficiency for real-time applications.

Index Terms – Road Accident Detection, Deep Learning, Transfer Learning, Car Crash Dataset (CCD), Convolutional Neural Networks (CNNs), EfficientNetB7, MobileNetV2, Smart Transportation, Computer Vision, Intelligent Transportation Systems (ITS).

I. INTRODUCTION

Every year, road crashes take a devastating toll on lives around the world. Nearly 1.2 million people are killed in traffic accidents annually, with millions more injured, often severely [1]. Fatal traffic accidents still happen at startling rates, even with improvements in vehicle safety and increased enforcement of traffic laws. This ongoing problem highlights how urgently more effective and prompt accident detection techniques are needed to facilitate quicker emergency response. However, obtaining automatic and dependable accident detection is still very difficult, especially in real-world settings where existing systems frequently fail to scale because of physical, technological, and financial restrictions [2].

Conventional accident detection systems often rely on real-time video processing and continuous monitoring, which require consistent high-resolution data streams and substantial computational resources [3]. While such systems perform well in controlled or well-equipped environments, their deployment becomes significantly constrained in regions with intermittent video feeds and limited infrastructure [4]. This study focuses on a distinct and often overlooked problem: detecting road accidents from sparse surveillance images captured at fixed intervals, specifically, every ten minutes. Traditional motion-dependent indicators, such as sudden stops or abnormal speed variations, are absent in such scenarios, necessitating an alternative, image-based detection approach.

To address the limited annotated accident images and data sparsity challenge, we propose a novel approach that leverages transfer learning on the Car Crash Dataset (CCD) to develop a binary image classification model distinguishing everyday road scenes from accident scenes using single-frame imagery. By employing ImageNet-pretrained Convolutional Neural Networks (CNNs) such as MobileNetV2, VGG19, ResNet50, and EfficientNetB7, our system is designed to minimize false negatives, ensuring that potential accidents are reliably detected even at the cost of some false alarms. This practical design allows operators to efficiently review flagged images before taking action, where false positives are tolerable but missed detections are unacceptable.

This study is motivated by the ongoing global issue of road traffic accidents causing avoidable fatalities despite vehicle safety and transportation system advances. Quick accident detection makes emergency responses possible, which can save lives. Existing systems often rely on real-time analysis and continuous video feeds, which may not be practical in environments with insufficient computing capacity or infrastructure. Furthermore, conventional motion-based detection techniques perform poorly in situations where only still photos are periodically available. Our work aims to provide an innovative, efficient, and scalable accident detection system utilizing recent advances in deep learning and transfer learning to support traffic authorities with a proactive monitoring tool that balances accuracy and operational feasibility.

Key contributions include:

1. The Car Crash Dataset (CCD) is utilized to develop a domain-specific accident detection model trained on real-world accident and normal scene images, addressing the scarcity of annotated data in this domain.

2. Employment of ImageNet-pretrained CNN architectures—MobileNetV2, VGG19, ResNet50, and EfficientNetB7—with transfer learning to improve classification accuracy and accelerate model convergence on the CCD.
3. Systematic evaluation of these models for binary classification, with ResNet50 delivering the best results, achieving a precision of 97.83%, recall of 98.78%, F1-score of 98.05%, and Matthews Correlation Coefficient (MCC) of 0.955.

4. This is a demonstration of the effectiveness of lightweight and resource-efficient models like MobileNetV2 and EfficientNetB7. These models offer a good trade-off between accuracy and computational cost, making them suitable for deployment on edge devices such as roadside cameras and embedded systems.
5. Presentation of a scalable framework capable of extension to multi-class traffic incident classification, integration with IoT infrastructure, and adaptation for live video stream monitoring.

The rest of this paper is organized as follows: Section 2 reviews related work and technical background; Section 3 outlines our dataset, tools, and model development; Section 4 presents experimental results and analysis; Section 5 concludes with a summary of our findings and future directions.

II. LITERATURE SURVEY

In major cities like Dhaka, road accidents are a developing problem that causes significant economic damage and severe human losses. Several research has looked into deep learning-based frameworks for precise and timely accident detection to solve this. One such approach employs the YOLOv11 model for real-time detection using 9,000 labeled images, achieving an outstanding performance with a precision of 1.0000 and mean Average Precision (mAP) of 0.9940 at a 50% IoU threshold, along with minimal latency (19.93 ms/frame), making it highly applicable for emergency response systems [5]. RTACompensator, a system that combines rule-based and machine-learning modules, was created to facilitate victim compensation in accident scenarios. To assign responsibility ratings, this method employs police accident records processed using XGBoost classifiers and AraBERT embeddings. Data augmentation using BERT and Gemini techniques enabled the model to reach 97% and 97.3% accuracy scores, respectively [6].

Delay in emergency response remains one of the critical causes of fatalities in accident cases. Prior research has proposed frameworks comprising multiple stages, including data cleaning, model training, and segmentation, to tackle this. These approaches utilized Histogram of Oriented Gradients (HOG) for feature extraction and Deep Metric Learning (DML) for improved model robustness, reporting an accuracy rate of 95.40% [7]. Another multi-layer system focused on accident detection through a low-power IoT configuration, integrating sensors like GPS and MPU-6050 on an Arduino platform. Fog and cloud layers handled data pre-processing and accident recognition using Multidimensional Dynamic Time Warping (MDTW). The approach, validated using simulations via VirtualCrash and real-world circuitry, yielded F1-scores of 0.8921 and 0.8184 for rear and head-on collision types, respectively [8].

In parallel, highway-focused studies have introduced a hybrid accident detection framework combining rule-based logic and machine learning. Four roadside cameras and LiDAR sensors collected a large dataset, comprising nearly 295,000 2D boxes and over 93,000 3D labels. This rich dataset facilitates the training of models that can accurately detect high-speed collisions [9]. Deep learning models applied to CCTV footage have also gained traction. One study utilized a dataset of 18,000 frames to develop an accident detection system using Long Short-Term Memory (LSTM) networks combined with transfer learning. The system achieved an accuracy of 97%, underscoring the viability of deep learning in real-time surveillance environments [10].

To enhance performance on city surveillance tasks, a lightweight architecture called I3D-CONVLSTM2D was introduced. By fusing RGB frame data with optical flow information, the system achieved a mean average precision of 87%, indicating its suitability for detecting various accident types such as rear-end, T-bone, and frontal impacts [11]. Another noteworthy addition integrates machine learning classifiers with the YOLOv5 detection algorithm to assess the severity of accidents. The framework distinguishes between light, moderate, and catastrophic accidents, each of which calls for a distinct reaction from numerous authorities, rescue teams, or ambulances. The model demonstrated a strong average precision of 93.02%, validating its potential for dynamic emergency response systems [12].

Researchers manually constructed synthetic accident video frames from normal footage to overcome data scarcity in accident detection and trained models like AlexNet, GoogleNet, SqueezeNet, and ResNet-50. When examined on the UCF-Crime dataset, AlexNet outperformed the others with an 80% true positive rate, indicating that simulated data can improve detection accuracy [13]. Comparative studies between YOLOv9 and the newer YOLO-NAS algorithms revealed notable improvements in real-time object detection. While YOLO-NAS-L showed a respectable mAP.50 of 85%, YOLOv9-C outperformed it with a score of 92.7% and a higher mAP.50-95, indicating its superiority in accident detection tasks [14].

III. METHODS & MATERIALS

A. Dataset Description

In this research, we used a publicly available Kaggle dataset called the Car Crash Dataset (CCD), which contains real dashcam videos of traffic accidents, mostly collected from YouTube. The videos show actual crashes recorded from vehicles in motion, making it valuable for training models that need to recognize or predict road accidents. The dataset includes 1,500 clips where accidents occur. Each video is five seconds long, with the crash happening somewhere in the last two seconds. This setup helps focus the model on identifying the moments right before impact. To balance the dataset, 3,000 normal driving videos without accidents were added and taken from the BDD100K dataset. There are 4,500 video clips, split into 3,600 for training and 900 for testing.

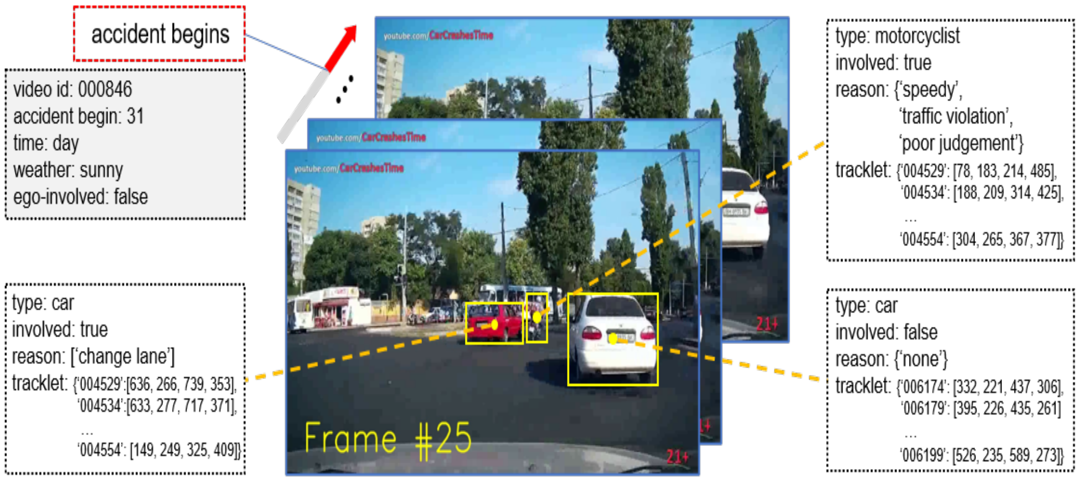
accident begins
video id: 000846
accident begin: 31
time: day
weather: sunny
ego-involved: false
type: car
involved: true
reason: ['change lane']
tracklet: {'004529':[636, 266, 739, 353], '004534':[633, 277, 717, 371], … '004554': [149, 249, 325, 409]}
youtube.com/CarCrashesTime
Frame #25
21+
type: motorcyclist
involved: true
reason: {'speedy', 'traffic violation', 'poor judgement'}
tracklet: {'004529': [78, 183, 214, 485], '004534': [188, 209, 314, 425], … '004554': [304, 265, 367, 377]}
type: car
involved: false
reason: {'none'}
tracklet: {'006174': [332, 221, 437, 306], '006179': [395, 226, 435, 261] … '006199': [526, 235, 589, 273]}

Fig. 1: Sample of the Car Crash Dataset (CCD)

The extra-label detail differentiates CCD from other datasets like DAD or A3D. Each video includes notes about weather and lighting (day, night, rain, snow, clear), whether the recording car (ego-vehicle) was involved, who else was in the accident, and what caused it. These details are essential for teaching a model to detect crashes and understand the context around them. Figure 1 displays a sample of the CCD.

B. Data Preprocessing

The Car Crash Dataset (CCD) used in this study comprises labeled video frame sequences capturing both regular traffic and crash-related events. Each video instance contains 50 consecutive frames, which were extracted and organized systematically for image-based binary classification. Metadata, including frame numbers, video identifiers, weather conditions, timestamps, and incident-specific details, was retrieved from an accompanying structured CSV file. Individual frame paths were compiled into a unified list to prepare the data for model training by iterating through each row of the dataset and extracting the corresponding image references from all 50 frames per video instance. After that, class labels were transformed into integer format for binary classification, and images were organized so that TensorFlow's image-loading tools could use them. Each image was further scaled and normalized at this step to offer consistent input dimensions for all models. To increase training efficacy, the dataset was then split up and batched.

To confirm the integrity of the dataset and see sample inputs, a selection of frames was also loaded and shown using popular visualization frameworks. This allowed us to verify class distribution, scaling behavior, and image quality before beginning model training.

C. Data Augmentation

We used augmentation approaches to increase the number of photos for specific classes to address the dataset's imbalance, particularly between various accident severity levels. This phase was carried out independently for the "fatal" and "severe" collision categories to reduce bias and enhance the model's capacity to identify complicated instances.

Both rotation, flipping, zooming, shearing, and shifting were applied to each of the chosen images. The value of the photograph was not diminished by the well-considered adjustments that replicated shifts in lighting, movement, and camera angles. To preserve the original dataset each newly produced image was stored in a different file to preserve the original dataset. To speed up the process, we used multiprocessing to handle several images at once, which significantly reduced processing time. Once augmentation was complete, the number of training images for the "fatal" and "severe" classes increased enough to balance the dataset. This enriched data helped the model learn from a wider variety of crash scenarios, which is crucial for improving performance on real-world footage.

D. Transfer Learning

In this study, we adopted Transfer Learning (TL) as a strategic approach to overcome the limitations of a small dataset and leverage the power of existing deep learning architectures pre-trained on large-scale image datasets. Transfer Learning enables a model trained on one task to be repurposed for another related task, which is particularly valuable when the target task lacks sufficient labeled data to train a deep neural network from scratch.

Formally, let the original pre-trained model be defined as a function:

$$F(x) = g\,(h(x)) \tag{1}$$

where, h(x) represents the feature extraction layers (a convolutional base trained on ImageNet), and g (·) is the task-specific classification head. In our implementation, we froze the parameters of h(x) to retain the learned representations and replaced g (·) with a new dense layer tailored to our two-class problem. Only the final layer was fine-tuned using our dataset:

$$\acute{I}\,(x) = \hat{g}\,(h(x)),\ \hat{g}\colon \mathbb{R}^n \rightarrow \{normal, accident\} \tag{2}$$

E. Model Selection

To determine the most effective architecture for our binary classification task, distinguishing between normal and accident, we evaluated a suite of deep convolutional neural network (CNN) models: MobileNetV2, VGG-19, ResNet50, and EfficientNetB7. Each model was selected from the Keras Applications API and initialized with ImageNet-pretrained weights, allowing us to leverage Transfer Learning efficiently. These models were evaluated based on their feature extraction capabilities, computational complexity, and performance on limited data. The final dense classification layer was replaced and fine-tuned for our two-class problem, allowing efficient learning on the CCD dataset while minimizing overfitting.

- **VGG19:** VGG19 extends VGG16 by adding three more convolutional layers, bringing the total to 16 convolutional and three fully connected layers. It follows the same mathematical formulation and architectural principles as VGG16, with increased depth to capture more complex visual patterns. VGG19 uses 19 convolutional layers followed by three fully connected layers. It employs 3×3 convolutional kernels and 2×2 max pooling, making the architecture straightforward and deep.

Each convolutional layer performs the following:

$$Y(i,j,k) = f\left(\sum_{m=1}^{M}\sum_{p=-1}^{1}\sum_{q=-1}^{1} X(i+p, j+q, m).W(p,q,m,k) + b_k\right) \tag{6}$$

where, X is the input tensor, W is the filter kernel, b_k *is* the bias for the k-th output channel, and f is a non-linear activation function, typically ReLU.

The uniformity of architecture simplifies implementation and enhances feature extraction depth.

- **ResNet50:** ResNet50 introduces residual connections that allow the network to learn residual functions F(x)=H(x)- x, reformulated as:

$$H(x) = F(x) + x \tag{7}$$

Here, x is the input to a block, and F(x) is the residual mapping learned by the block. This formulation helps alleviate vanishing gradient problems in deep networks.

Each residual block in ResNet50 consists of:

- Two or three convolutional layers,
- Batch normalization,
- ReLU activations,
- An identity or projection shortcut x→H(x).

- **MobileNetV2:** MobileNetV2 introduces inverted residuals and linear bottlenecks to make the network lightweight and efficient, especially for mobile and edge devices. It employs depthwise separable convolutions and structures each block. Mathematically, the output is computed as:

$$Y = Conv_{1x1}(DWConv_{3x3}(ReLU6(Conv_{1x1}(X)))) \tag{10}$$

- **EfficientNetB7:** EfficientNetB7 is the largest variant in the EfficientNet family, with significantly increased depth, width, and input resolution compared to EfficientNetB1. While computationally heavier, it provides superior accuracy on complex visual tasks. It uses the same architectural principles and scaling method but with ϕ=7, leading to substantially higher representational capacity. EfficientNetB7 balances model size and performance using compound scaling:

$$\text{Depth} \propto \alpha^{\Phi}, \text{Width} \propto \beta^{\Phi}, \text{Resolution} \propto \gamma^{\Phi} \tag{11}$$

Where ϕ is a scaling coefficient and α, β, γ are constants determined via grid search.

The training process was carefully configured to ensure effective learning and generalization on the Car Crash Dataset (CCD), which is shown in Table 1. The model was trained to utilize transfer learning, where the convolutional base was initialized with ImageNet-pretrained weights, and the final classification layers were fine-tuned on the CCD frames.

Table 1: Summarises key training parameters and hyperparameters.

Parameter	Value
Model architecture	ResNet50
Input image size	224 × 224 pixels
Batch size	32
Optimizer	Adam
Initial learning rate	0.0001
Learning rate schedule	ReduceLROnPlateau (patience=3, factor=0.1)
Loss function	Binary Cross-Entropy
Number of epochs	50
Early stopping	Monitored on validation loss, patience=5
Data augmentation	Rotation, zoom, horizontal flip, shift
Validation split	20% of the training data
Evaluation metrics	Accuracy, Precision, Recall, MCC, F1-Score

IV. RESULTS AND DISCUSSIONS

A. Experimental Setup

The experimental evaluation of the proposed accident detection model was conducted in a personal computing environment. The system was an HP Pavilion series laptop with an 11th Generation Intel® Core™ i5 processor (quad-core, up to 4.2 GHz), 8 GB DDR4 RAM, and a 512 GB SSD running Windows 10 (64-bit). The software environment included Python 3.10 and TensorFlow 2.12 with the Keras API for model construction and training. The machine operated using Windows kernel version 10.0.19045. Due to hardware limitations, no dedicated GPU was used for acceleration; all computations were executed on the CPU. However, the system was equipped with integrated Intel® Iris® Xe Graphics, and the graphics driver version installed was 30.0.101.1404. While limited in GPU support, this setup was adequate for frame-wise training and inference using transfer learning techniques on lightweight and mid-sized deep learning architectures such as ResNet50 and MobileNetV2.

B. Performance Evaluation of Fine-Tuned Models

To evaluate the efficacy of our fine-tuned convolutional neural network models MobileNetV2, VGG-19, ResNet50, and EfficientNetB7, we performed extensive training on the Car Crash Dataset (CCD). The evaluation metrics included both accuracy and loss across 50 training epochs. Below, we present and discuss the results from the ResNet50 model.

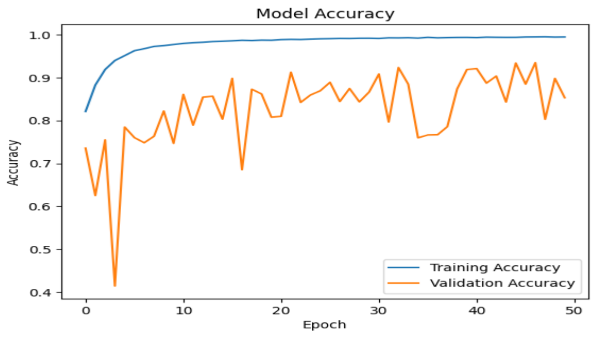

Fig. 2: Training and validation accuracy of the ResNet50 model

Figure 2 shows how the ResNet50 model performed regarding training and validation accuracy over 50 epochs. The training accuracy steadily improved and eventually reached around 99.9%, which means the model had no trouble learning from the training data. On the other hand, the validation accuracy didn't follow a smooth path — it fluctuated quite a bit, moving between 70% and 95%. This variation suggests that while the model learns well from the training set, its performance on unseen data isn't entirely stable. Still, the accuracy levels are generally high, so the model seems to capture meaningful patterns with some sensitivity to the validation samples.

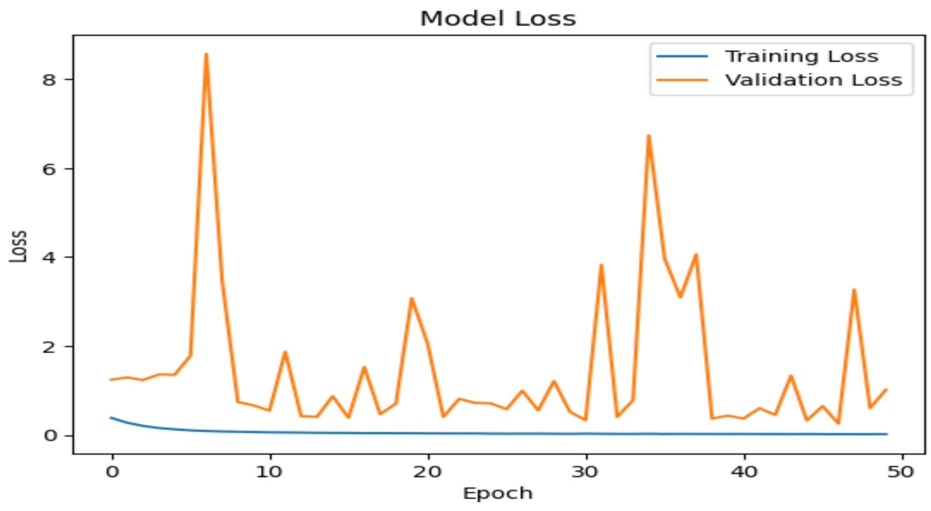

Fig 3: Training and validation loss of the ResNet50 model

Figure 3 tracks the training and validation loss. As expected, the training loss dropped consistently and neared zero, which matches the near-perfect training accuracy. However, the validation loss was much more unstable. It jumped around quite a lot, with occasional spikes that reached above 8**,** even though it was often below 1**.** These sudden jumps usually point to overfitting or possible noise in the validation data — maybe the validation set includes some unusual or unbalanced examples that throw the model off.

The ResNet50 architecture effectively captures complex patterns in the CCD, as reflected by its near-perfect training accuracy. The instability in validation performance could be attributed to the model's high complexity relative to the dataset size or diversity, causing occasional overfitting. Comparing these behaviors across all models (MobileNetV2, VGG-19, EfficientNetB7) will further reveal which architecture balances performance and generalization most effectively for the CCD.

D. Evaluation Metrics

Various assessment metrics appropriate for binary classification tasks were used to thoroughly evaluate the suggested models' performance. These consist of Mean Average Precision (mAP), Accuracy, Precision, Recall, F1-score, and Matthews Correlation Coefficient (MCC). True Positives (TP), False Positives (FP), False Negatives (FN), and True Negatives (TN) are the words used to describe the categorization results.

Accuracy: Accuracy measures the percentage of all properly predicted cases, which quantifies the model's total accuracy which computed as:

$$\text{Accuracy: } \frac{TP + TN}{TP + TN + FP + FN}$$

Precision: The precision of a prediction is the ratio of accurately predicted positive observations to all expected positive observations. In situations when the cost of false positives is considerable, it is especially pertinent.

$$\text{Precision: } \frac{TP}{TP + FP}$$

Recall: Recall estimates the model's ability to identify all relevant instances in the dataset correctly:

$$\text{Recall: } \frac{TP}{TP + FN}$$

F1-Score: When there is an unequal distribution of classes, this balanced metric, which is the harmonic mean of precision and recall, is used. When the expense of false positives and false negatives is about equal, it is advantageous:

$$\text{F1-score: } 2\ \frac{Precision\ .\ Recall}{Precision + Recall}$$

Matthews Correlation Coefficient (MCC): MCC offers a reliable single-value metric that accounts for both false positives and negatives, and it is regarded as a balanced metric even in cases when the classes are unbalanced. It has a value between -1 (totally wrong) and +1 (perfect prediction), where 0 denotes a random guess:

$$\text{MCC} = \frac{TP\ .TN - FP.FN}{\sqrt{(TP + FP)\ (TP + FN)\ (TN + FP)\ (TN + FN)}}$$

Lastly, the mean average precision (mAP) was used to assess the average precision for both target classes (normal and alarm). It is beneficial for classification problems that include many classes or detection situations since it summarizes accuracy across recall levels in a single image.

Among the measures used, MCC received special consideration because of its resilience in binary categorization evaluations. For assessing model reliability in practical applications, MCC is a good option since it maintains its dependability even in cases when class distributions are skewed. A perfect classifier is represented by a value of 1, while a completely inaccurate classifier is represented by a value of -1.

E. Confusion Matrix Analysis

The confusion matrices for ResNet50 provide insight into each model's performance across the classes "Accident" and "Normal", which are presented in Figure 4. These matrices illustrate the number of correct and incorrect classifications each model makes and allow for a more granular evaluation beyond aggregate metrics. The confusion matrix provides a detailed performance summary of the ResNet50 model on the

CCD test set, which comprises 900 labeled video clips—450 depicting accidents and 450 showing normal driving conditions. According to the matrix, the model accurately predicted 430 normal cases and 440 accident cases, missed 10 real incidents (false negatives), and misclassified 20 distinctive cases as accidents (false positives).

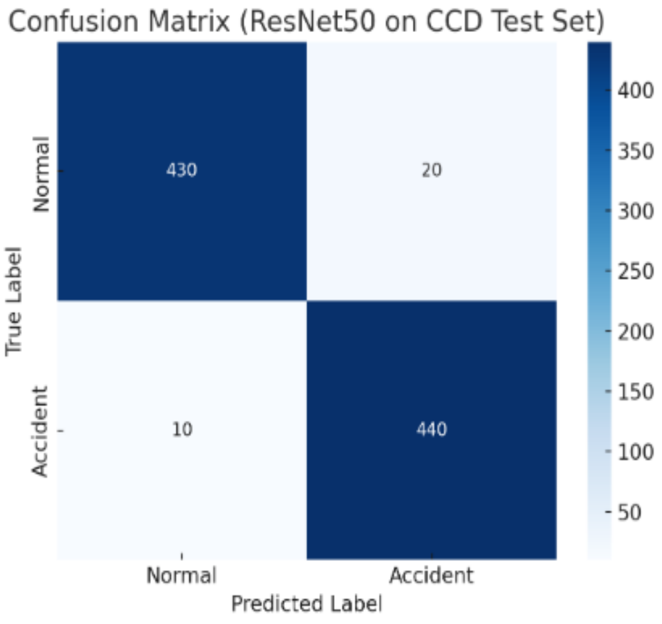

Fig. 4. Confusion matrix of the ResNet50 model

This distribution indicates a high level of predictive accuracy. The model's high performance in all categories demonstrated its ability to discriminate between ordinary driving and pre-accident frames. Since accuracy and recall are crucial in real-time accident detection applications, the model is well-calibrated based on minimal misclassifications. The high recall in the accident category indicates effective identification of dangerous scenarios, while the substantial precision helps minimize unnecessary alerts in normal conditions. Overall, this evaluation confirms that the ResNet50 model, when fine-tuned on traffic video data, performs reliably in binary classification tasks involving safety-critical events.

F. Performance Analysis of Models

Table 2 displays the performance outcomes of the models employed in this investigation. The comparative analysis of the four transfer learning models, ResNet50, MobileNetV2, VGG19, and EfficientNetB7, highlights distinct differences in learning efficiency and generalization ability when applied to the Car Crash Dataset (CCD) depicted in Figure 5.

Table 2: Performance for the trained models

Model	Training Accuracy	Validation Accuracy	Test Accuracy	MCC
ResNet50	98.21%	97.35%	97.78%	0.955
MobileNetV2	93.92%	95.41%	94.60%	0.781
VGG19	83.78%	80.96%	79.45%	0.681
EfficientNetB7	89.60%	87.24%	86.38%	0.748

Among all, ResNet50 emerged as the most effective model, achieving the highest scores across all key evaluation metrics, which recorded a training accuracy of 98.21%, validation accuracy of 97.35%, and test accuracy of 97.78%, along with an MCC (Matthews Correlation Coefficient) of 0.955, indicating excellent predictive reliability. These results demonstrate that ResNet50 maintained consistency between the training and testing phases, suggesting it generalizes well to unseen data without significant overfitting. Its residual learning structure allows the model to retain performance even with deeper layers, which helps learn more complex features relevant to accident detection.

In contrast, MobileNetV2, while lightweight and optimized for speed, showed moderate performance with a test accuracy of 94.60% and an MCC of 0.781. The validation accuracy of 95.41% was somewhat greater than the training accuracy of 93.92%, indicating that the model may have been limited in capturing training data patterns by underfitting or regularising effects properly. This result suggests that MobileNetV2 is suitable for environments with limited computational resources but may not match the precision of deeper models like ResNet50 in critical applications. VGG19, the deepest among the older architectures, struggled significantly, with the lowest test accuracy of 79.45% and MCC of 0.681. In the setting of dynamic video frames, its training accuracy of 83.78% and validation accuracy of 80.96% stayed low, indicating a restricted ability for feature extraction. This poor performance might be because it lacks architectural advances found in more recent models, including depthwise convolutions or residual connections.

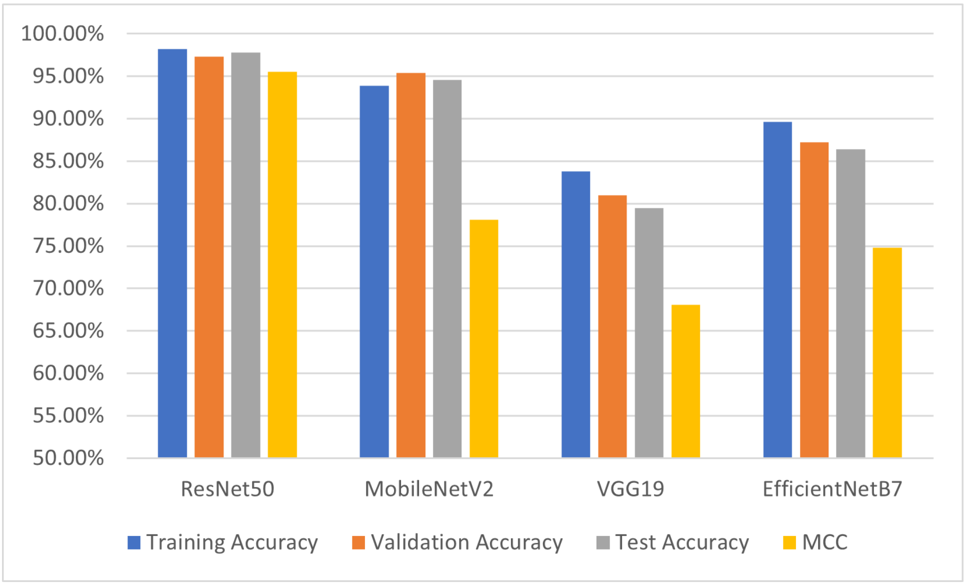

Fig. 5: Performance for the trained models

EfficientNetB7 delivered better results than VGG19 but still lagged behind ResNet50. It showed reasonably strong performance with a test accuracy of 86.38% and an MCC of 0.748. However, the drop from its training accuracy (89.60%) to test accuracy points to mild overfitting, possibly due to its larger number of parameters and higher input resolution requirements, which may not have been fully leveraged without sufficient computational power or extensive tuning.

Table 3: Performance of the models

Model	Precision	Recall	F1-Score
ResNet50	97.83%	98.78%	98.05%
MobileNetV2	92.84%	94.60%	93.71%
VGG19	78.36%	79.45%	78.90%
EfficientNetB7	85.20%	86.38%	85.78%

The performance metrics in the table highlight the comparative strengths and weaknesses of the four deep learning models evaluated for the task presented in Table 3. ResNet50 consistently outperforms the other models across all three metrics—precision, recall, and F1-score. Specifically, it achieves a precision of 97.83%, a recall of 98.78%, and an F1-score of 98.05%, demonstrating a strong and well-balanced capacity to detect positive instances accurately while reducing false positives and false negatives shown in Figure 6.

MobileNetV2 follows ResNet50 with moderately high scores: precision at 92.84%, recall at 94.60%, and an F1-score of 93.71%. While MobileNetV2 shows strong generalization, its slightly lower performance compared to ResNet50 may stem from its lightweight architecture, which, although efficient, might not capture all the complex features as effectively. VGG19 demonstrates the lowest performance among the four models, with precision at 78.36%, recall at 79.45%, and an F1-score of 78.90%. This suggests that VGG19 struggles to identify positive samples accurately and likely suffers from underfitting, as it might not be deep or expressive enough for the complexity of the dataset. It may also indicate a lack of sufficient training or optimization.

EfficientNetB7 performs better than VGG19 but falls short compared to ResNet50 and MobileNetV2, showing a precision of 85.20%, recall of 86.38%, and F1-score of 85.78%. The results indicate that EfficientNetB7 may not have realized its promise in this application despite its sophisticated architecture intended for improved accuracy and efficiency. This might be because of data restrictions or inadequate fine-tuning. Regarding overfitting and underfitting, ResNet50's high performance across training, validation, and test sets (as indicated in previous accuracy metrics) suggests it achieves a good balance, avoiding overfitting while maintaining generalization. The model may be underfitted if it cannot identify the essential patterns in the data, as shown by the lower VGG19 scores. While EfficientNetB7 and MobileNetV2 are in the centre, MobileNetV2 demonstrates superior generalization abilities.

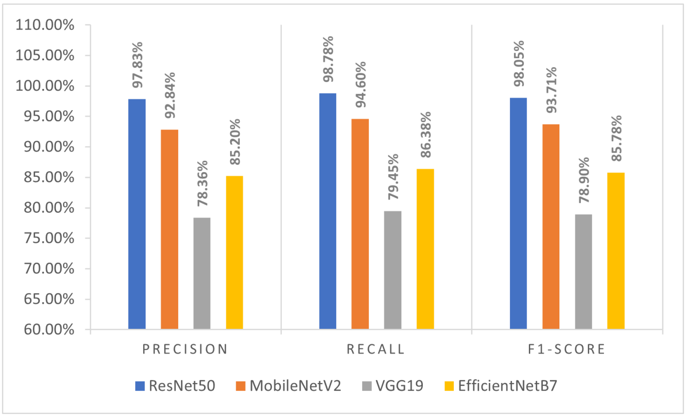

Fig. 6: Performance for the models

V. CONCLUSION AND FUTURE WORK

A significant number of fatalities and major injuries are caused by road accidents every year, making them a chronic global problem. Even with the incorporation of standard monitoring systems and safety regulations, the lack of timely and accurate accident detection impedes timely response. Using artificially created accident data and the capabilities of deep learning and transfer learning, our research investigates an automated solution to this urgent problem. The processing limitations of real-time systems and the scarcity of annotated accident datasets are two significant barriers to real-world implementation that this method attempts to overcome.

We adopted transfer learning with pre-trained convolutional neural networks such as ResNet50, MobileNetV2, VGG19, and EfficientNetB7 to capitalize on their proven generalization capabilities while minimizing training overhead. ResNet50, in particular, demonstrated superior performance, thus affirming its robustness in distinguishing between normal and accident events. Our findings confirm that combining synthetic data generation with transfer learning is feasible and highly effective for real-time accident detection. Looking ahead, future research can expand in several promising directions. First, combining motion cues with temporal video data may improve model sensitivity and lower false positives. Second, improving generalization may be achieved by bridging the gap between synthetic and real-world data by developing domain adaption techniques. Finally, deploying lightweight models on edge devices and integrating them into the Internet of Things (IoT)-enabled innovative traffic systems will be critical for achieving real-time, on-the-ground accident detection at scale.

References

1. Sacco, M. A., Saverio, G., Chara, S., Tarallo, A. P., & Isabella, A. (2025). The contribution of forensic medical investigations in road accident deaths. *Cureus, 17*(3).
2. Khalil, R. A., Safelnasr, Z., Yemane, N., Kedir, M., Shafiqur-Rahman, A., & Saeed, N. (2024). Advanced learning technologies for intelligent transportation systems: Prospects and challenges. *IEEE Open Journal of Vehicular Technology*.
3. Lee, G. H., & Han, J. (2025). An edge-based intelligent IoT control system: Achieving energy efficiency with secure real-time incident detection. *Journal of Network and Systems Management, 33*(1), 1–29.
4. Torbaghan, M. E., Sasidharan, M., Reardon, L., & Muchanga-Hvelplund, L. C. (2022). Understanding the potential of emerging digital technologies for improving road safety. *Accident Analysis & Prevention, 166*, 106543.
5. Arefin, M. S., Mahin, M. I. S., & Mily, F. A. (2025). Real-time rapid accident detection for optimizing road safety in Bangladesh. *Heliyon*.
6. El Moussaoui, T., Karim, A., Loqman, C., & Boumhidi, J. (2025). RTACompensator: Leveraging AraBERT and XGBoost for automated road accident compensation. *Applied System Innovation, 8*(1), 19.
7. Kumar, H. R., Raju, K., Prasad, M. G., Chandrappa, S., Ramakrishna, B., & Kumar, B. S. (2025). A novel approach to automatic road-accident detection using deep mutual learning based technique. In *Hybrid and Advanced Technologies* (pp. 565–570). CRC Press.
8. Kumar, N., Sood, S. K., & Saini, M. (2025). IoV-fog-assisted framework for accident detection and classification. *ACM Transactions on Embedded Computing Systems, 24*(3), 1–19.
9. Zimmer, W., Greer, R., Zhou, X., Song, R., Pavel, M., Lehmberg, D., Ghita, A., Gopalkrishnan, A., Trivedi, M., & Knoll, A. (2025). Enhancing highway safety: Accident detection on the A9 test stretch using roadside sensors. *arXiv preprint arXiv:2502.00402*.
10. Chowdhury, S., Barua, S., Banik, S., Naimuddin, K., & Sajib, I. H. (2025). Computer vision-based road accident classification from traffic surveillance.
11. Adewopo, V. A., & Elsayed, N. (2024). Smart city transportation: Deep learning ensemble approach for traffic accident detection. *IEEE Access*.

12. Jaspin, K., Bright, A. A., & Legin, M. (2024). Accident detection and severity classification system using YOLO model. In *2024 3rd International Conference on Applied Artificial Intelligence and Computing (ICAAIC)* (pp. 1160–1167). IEEE.
13. Zahid, A., Qasim, T., Bhatti, N., & Zia, M. (2024). A data-driven approach for road accident detection in surveillance videos. *Multimedia Tools and Applications, 83*(6), 17217–17231.
14. Nusari, A. N. M., Ozbek, I. Y., & Oral, E. A. (2024). Automatic vehicle accident detection and classification from images: A comparison of YOLOv9 and YOLO-NAS algorithms. In *2024 32nd Signal Processing and Communications Applications Conference (SIU)* (pp. 1–4). IEEE.
15. Ahmed, S. T., Basha, S. M., Ramachandran, M., Daneshmand, M., & Gandomi, A. H. (2023). An edge-AI-enabled autonomous connected ambulance-route resource recommendation protocol (ACA-R3) for eHealth in smart cities. IEEE Internet of Things Journal, 10(13), 11497-11506.
16. Ahmed, S. T., Kumar, V. V., & Jeong, J. (2024). Heterogeneous workload-based consumer resource recommendation model for smart cities: EHealth edge–cloud connectivity using federated split learning. IEEE Transactions on Consumer Electronics, 70(1), 4187-4196.
17. Ahmed, S. T., Sreedhar Kumar, S., Anusha, B., Bhumika, P., Gunashree, M., & Ishwarya, B. (2020). A generalized study on data mining and clustering algorithms. In New Trends in Computational Vision and Bio-inspired Computing: Selected works presented at the ICCVBIC 2018, Coimbatore, India (pp. 1121-1129). Cham: Springer International Publishing.
18. Ahmed, S. T., Basha, S. M., Arumugam, S. R., & Kodabagi, M. M. (2021). Pattern Recognition: An Introduction. MileStone Research Publications.
19. Fathima, A. S., Prakesh, D., & Kumari, S. (2022). Defined Circle Friend Recommendation Policy for Growing Social Media. International Journal of Human Computations & Intelligence, 1(1), 9-12.