Enhancing Smart City Safety: Deep Learning Approaches for Automatic Vehicle Accident Recognition

Ahad AlNemari, Shahad AlOtaibi, Majd Jada, Aeshah AlHarthi,
Sara AlThuwaybi, Wojoud AlNemari, Nadan Marran, Abdulmajeed Alsufyani*
Department of Computer Science-Computers and Information Technology College, Taif University, Taif, Saudi Arabia

***Abstract*—Traffic accidents have significant societal impacts due to the substantial human and material losses they cause. Recently, numerous AI-based traffic surveillance technologies, such as Saher, have been implemented to improve traffic safety in Saudi Arabia. The prompt detection of vehicle accidents is crucial for enhancing the response time of accident management systems, thereby reducing the number of injuries resulting from collisions. This study evaluates various deep learning algorithms to determine the most effective method for detecting and classifying car accidents. Multiple deep-learning models were trained and tested using an extensive dataset of car accident images, allowing for the accurate identification and classification of different types of accidents. Among the six pre-trained models analyzed, ResNet-101 achieved the highest accuracy, with a classification rate of 93%. For accident detection, YOLOv5 attained a mean Average Precision (mAP) of 97.8%, indicating superior performance compared to YOLOv8 and YOLOv9, and highlighting its capability to effectively detect accidents in video footage. The research's primary goal is to enhance urban safety by enabling rapid accident detection, which supports timely emergency responses, minimizes fatalities, and contributes to the development of safer and more resilient smart cities.**

***Keywords*—Accident detection; deep learning algorithms; ResNet-101; traffic safety; YOLOv5; YOLOv9**

I. Introduction

Traffic accidents pose a significant global public health issue, with an annual toll of approximately 1.35 million deaths or disabilities. Each day, around 3,700 lives are lost in fatal accidents, impacting vulnerable road users such as cyclists, motorcyclists, and pedestrians the most. This issue is not confined to specific regions but affects countries worldwide. For instance, Saudi Arabia faces considerable challenges, with cities like Riyadh reporting around 4,000 daily traffic-related injuries [1]. In contrast, Jeddah, Makkah, and Taif report 2,500, 800, and 200 daily injuries, respectively. These statistics highlight the urgent need for enhanced road safety measures and effective strategies to reduce traffic accidents, save lives, and mitigate associated social and economic repercussions. A review of existing traffic management and accident detection systems indicates that while several technological solutions have been implemented, gaps remain in their real-time detection accuracy and rapid emergency response capabilities.

A. Modern Car Accident Mitigation Technologies in Saudi Arabia

In response to the global goal of reducing traffic accident fatalities, Saudi Arabia has set ambitious targets within its Vision 2030 framework, aiming to halve road accident fatalities. This initiative involves utilizing modern technologies to identify and manage high-risk areas, known as "black spots." Key measures implemented include the "Saher" automated monitoring system, active enforcement by traffic officers, and advancements in traffic management systems. These efforts have resulted in a significant 35% reduction in road accident fatalities over the past five years [2]. The number of deaths decreased from 9,311 in 2016 to 6,651 in 2021 as reported by the Global Health Organization. However, despite these advancements, current systems still struggle with automatic detection and classification of accidents from real-time visual data, limiting their effectiveness in minimizing response times.

Machine learning (ML), a subset of artificial intelligence (AI), involves creating algorithms and models that enable computers to learn from data and make predictions or decisions without explicit programming. ML's ability to handle complex data and solve intricate problems has led to its widespread use. Core types of ML algorithms include:

- Supervised Learning: Uses labeled datasets to train models to map inputs to outputs. It is commonly applied in classification and regression tasks.
- Unsupervised Learning: Involves training models on unlabeled data to identify patterns and relationships. Techniques include clustering and association.
- Semi-Supervised Learning: Combines labeled and unlabeled data to improve model accuracy, often used when labeled data is scarce [3].
- Reinforcement Learning: Focuses on training agents to make decisions by rewarding or punishing actions, aiming to maximize cumulative rewards.

ML methods such as Bayesian networks, decision trees, neural networks, and support vector machines are employed across various fields, including autonomous vehicles, audio and image recognition, natural language processing, and healthcare diagnostics.

*Corresponding author.

Deep learning, a branch of machine learning, involves the use of deep neural networks with multiple layers to model complex data representations. Inspired by the human brain's neural network structure, deep learning excels in tasks like speech and image recognition. Key models include:

- Convolutional Neural Networks (CNNs): Effective for image analysis and feature extraction, used in image classification and object detection.
- Recurrent Neural Networks (RNNs): Suitable for sequential data, including time series and natural language text. Variants like Long Short-Term Memory (LSTM) and Gated Recurrent Units (GRUs) address long-term dependencies.
- Generative Models: Such as Generative Adversarial Networks (GANs) and Variational Autoencoders (VAEs), are used to create new data samples by learning from existing data distributions.

Deep learning's ability to automatically extract hierarchical features from data has revolutionized fields such as computer vision and natural language processing.

Computer vision is a field of AI focused on enabling computers to interpret and understand visual information from images or videos. Techniques and applications include:

- Image Classification: Assigning labels to images using CNNs, which have shown significant advancements in complex recognition tasks.
- Object Detection and Recognition: Identifying and localizing objects in images, enhanced by methods like R-CNN, YOLO, and SSD.
- Image Segmentation: Dividing images into segments to understand object boundaries and relationships, with methods like FCNs and U-Net.
- Face Recognition: Identifying and verifying individuals based on facial features, employing techniques like FaceNet and DeepFace.
- Video Analysis: Analyzing video data for tasks such as activity detection and tracking, using RNNs and 3D CNNs.
- Augmented Reality (AR) and Virtual Reality (VR): Enabling immersive experiences through object tracking and scene understanding.
- Medical Imaging: Assisting in diagnosis through tasks like tumor detection and image segmentation.

B. Problem Statement

Delayed responses to traffic accidents can lead to increased fatalities, severe complications for the injured, and significant economic burdens. Inefficient emergency response times impede the timely delivery of medical care and emergency services, exacerbating injury severity and reducing survival chances. There is an urgent need for effective solutions to improve emergency response systems and enhance public safety on roadways. Current automated detection methods are limited in their ability to provide rapid and accurate identification of accidents in real-time, highlighting a critical gap that this research aims to address.

C. Objectives and Contribution

This research aims to evaluate various models for detecting and identifying traffic accidents or collisions using image datasets. The research focuses on analyzing visual data and leveraging deep learning techniques to develop effective models for accident detection. Key objectives include:

- Analyzing and processing visual data to accurately recognize and classify car accidents.
- Training and evaluating six deep learning models to detect and classify accidents, assessing their performance to determine the most effective approach.

By employing advanced image and video analysis techniques, this project seeks to enhance the accuracy and efficiency of traffic accident detection systems.

In summary, while significant efforts and technological advancements have been made to reduce traffic accidents, gaps remain in the real-time detection and accurate classification of accidents using visual data. This paper addresses this gap by evaluating and developing deep learning-based models for effective accident detection. The following sections will review related literature, present the methodologies employed, analyze results, and discuss how these approaches can improve emergency response systems and enhance overall road safety.

II. Literature Review

A. AI and IoT in Transportation Safety Systems

Recent systematic and bibliometric studies emphasize that the integration of machine learning (ML) and Internet of Things (IoT) technologies is driving innovation in smart transportation and other smart-city applications. Chahal et al. [40] conducted a comprehensive bibliometric review that highlighted healthcare, smart cities, energy, industrial IoT, security, agriculture, and transportation as the most prominent IoT application domains. Within transportation, their study specifically identified tasks such as traffic congestion prediction, road surveillance, parking, and accident detection as major areas where ML and IoT convergence has been most effective. They further categorized IoT data into real-time sensor streams, spatiotemporal data, and high-dimensional multimodal inputs, noting that these complex datasets often require deep learning or hybrid ML approaches for effective analysis. Importantly, the review concluded that deep learning typically outperforms traditional ML in vision- and data-intensive IoT applications, supporting the growing use of CNNs and object-detection models in transportation. It also emphasized that performance is commonly assessed with accuracy, precision, recall, F-measure, and mean Average Precision (mAP)—the same metrics applied in this paper to evaluate YOLO and CNN-based accident-detection models. Together, these insights place the present study within current research trends, showing alignment in application domain, data challenges, chosen methodologies, and evaluation practices, thereby reinforcing its novelty and practical contribution to smart transportation.

Complementing this perspective, Alshuaibi, Almaayah, and Ali [41] examined how ML can strengthen IoT systems by mitigating cybersecurity vulnerabilities and ensuring the reliability of connected infrastructures. While their focus was on the role of ML in protecting IoT environments against evolving threats, our work extends this principle to transportation by integrating IoT with advanced deep learning models such as YOLOv5 and ResNet-101, which achieved 97.8% mAP and 93% accuracy respectively for accident detection and classification. Both studies highlight the strength of AI-driven IoT frameworks in enhancing safety-critical applications, with theirs emphasizing security resilience and ours emphasizing rapid accident recognition to improve emergency response.

Similarly, Mousa and Shehab [42] conducted a structured risk analysis in the workstation domain, systematically classifying threats, vulnerabilities, and countermeasures through four stages: identifying key components, recognizing threats, pinpointing vulnerabilities, and determining effective controls. Although their study concentrated on cybersecurity—addressing risks such as malware, man-in-the-middle attacks, and unpatched software—the strength of their work lies in the rigorous methodology for proactively reducing risks in complex environments. Our research applies the same principle in the transportation domain, where deep learning models such as ResNet-101 and YOLOv5 were employed to detect and classify car accidents with high accuracy. In both cases, AI serves as the foundation for systematically identifying vulnerabilities and implementing rapid countermeasures, whether through cybersecurity defenses or accident detection systems that improve emergency response in smart cities. This parallel underscores how different domains can benefit from AI and IoT integration to enhance resilience, minimize risks, and contribute to safer, smarter environments.

B. Machine Learning Algorithms in Transportation Applications

This section will focus on references that use Machine Learning algorithms in transportation applications. Table I summarizes the combined references, presenting the algorithm name, the number of references for each application, the algorithm type, and the respective learning type.

Due to geographical interdependence and changeable traffic patterns, traffic forecasting is difficult. To estimate traffic status, Cui et al. [4] introduced TGC-LSTM, a deep learning framework that models the traffic network as a graph. Based on network architecture, they define traffic graph convolution and explain its connection to spectral graph convolution. Interpretability is improved by L1 and L2 regularization. The model outperforms baselines on real-world datasets, according to experimental results. Graph convolution weight visualization reveals important route segments.

Luo et al. [5] discussed the use of deep convolutional neural networks (CNN) for traffic sign classification and the challenges of obtaining labeled training data. The authors propose a solution using generative adversarial networks (GANs) to synthesize traffic sign images. By inputting a traffic sign template and a background image into the generative network, the method generates realistic synthetic images. Experimental results demonstrate that the proposed method outperforms conventional image synthesis approaches. Additionally, when these synthesized images are added to the training data for a CNN model, improved accuracy in traffic sign classification is achieved. Similarly, our work tackles the data scarcity problem by incorporating semi-realistic images from video games and simulators to create a more diverse and extensive dataset, thereby enhancing the models' ability to generalize to a wider range of real-world accident scenarios.

Zhang et al. [6] proposed ST-ResNet, a deep-learning-based approach for forecasting crowd flow in cities. It takes into account things like weather, events, and traffic between regions. A special end-to-end structure that integrates spatiotemporal data is used by ST-ResNet. For temporal features, residual neural networks are used, and residual contraction units are used for spatial aspects. Based on data, the model dynamically combines the outputs from various networks and gives branches and regions varying weights. It also considers outside variables like the day of the week and the weather. Experimental results on Beijing and New York City crowd flows demonstrate that ST-ResNet works better than six other approaches. Their work effectively addresses traffic forecasting with time-series data, but we chose to focus on a different, equally critical challenge in smart city safety. Our research uses image-based deep learning models to enable real-time accident detection, which is the foundation for a rapid emergency response system.

Quddus et al. [7] examined intelligent transport systems (ITS) and the critical role of map-matching algorithms in combining location and road network data. These algorithms have evolved using Kalman filtering, probabilistic theory, and topological analysis. However, they struggle to meet navigation needs in complex, crowded city situations. The study conducted an in-depth literature review to identify current map-matching algorithm limitations. It also discussed future implications, including the European Galileo and EGNOS systems. The research stressed the importance of algorithm integrity, noting that the algorithms discussed are general and do not rely on future data for simplicity.

Hou et al. [8] tested a Regression Tree to predict short-term and long-term traffic flow in work zones. Regression trees and decision trees are comparable, but in a regression tree, the response variable is a numerical value rather than a binary class. The independent variables in a regression tree are used to fit the response variable in a regression. The squared regression error is then computed by a binary recursive partitioning process. The branch that has the variables with the lowest squared error total is selected. In the previously described study, the FF-NN or the RF performed better at traffic predictions than the regression tree.

As autonomous vehicles (AVs) become more widespread, training policies, mobility, and energy efficiency all increase, as Tran and Bae showed in this study [9]. AVs significantly reduce traffic congestion and improve traffic flow. Full-autonomy traffic surpasses other AV penetration rates, with 2.55 times less wait time and 1.38 times higher average speed compared to all human-driven cars. The leading autonomous vehicle experiment outperforms both human-driven and other AV trials. The paper introduces a deep reinforcement learning framework and proposed PPO hyper-parameters for modeling mixed-autonomy

traffic at different AV penetration rates. The approach becomes more effective with higher AV penetration. Future research focuses on employing the latest deep learning algorithms to enhance the effectiveness of multiple autonomous cars in multi-intersection networks. In another study, uncertainty was added to the detection of freeway incidents using the Bayesian deep learning (BDL) approach [10]. The Bayesian statistical framework and deep learning are combined in the BDL method. The BDL framework combines the capacity for perception and reasoning by integrating the advantages of deep learning in representation learning with the approach of Bayesian inference. This study proposed a Bayesian deep learning model with uncertainty quantification for detecting highway accidents. Experiments were conducted to examine the efficiency and reliability of the proposed model using a real-world dataset from the Portland area. The results obtained demonstrate that the suggested model can minimize the false alarm rate from 29.18% to 5.09% and boost detection from 87.83% to 90.53%. Table I summarizes all the research mentioned above.

TABLE I. MACHINE LEARNING ALGORITHMS IN TRANSPORTATION APPLICATIONS

No	Algorithm	Reference	Algorithm Type	Learning Type
1	GCN	[4]	GNN	Supervised Learning
2	cGAN	[5]	GAN	Unsupervised Learning
3	RNN	[6]	RNN	Supervised Learning
4	ITS using Decision Trees	[7]	Decision Trees	Supervised Learning
5	FF-NN	[8]	ANN	Supervised Learning
6	PPO	[9]	Deep Learning (PPO)	Reinforcement Learning
7	LSTM and Bayesian model	[10]	RNN	Supervised Learning

C. Machine Learning Algorithms in Car Accident Detection and Classification

Several scientific papers discussed and proposed solutions to the task of detecting and classifying car accidents. In the first study, the researchers applied the binary image classification technique to identify potential traffic accidents utilizing models that combine transfer learning with deep learning [11]. The model MobileNetV2 had been selected due to its speed in execution, while the model EfficientNetB1 was considered due to its high prediction quality. The model EfficientNetB1 achieved a 0.89 mAP, while the MobileNetV2 has a mAP of 0.88. The dataset consists of a set of images obtained from a traffic surveillance camera that was placed on a road in Finland. Synthetic photos were created since there was no real accident data available. Therefore, both synthetic and actual data were used to train the models. Our work builds on this by employing a significantly larger and more diverse dataset of 1568 images, sourced from a variety of real-world and simulated environments, to improve model generalization and robustness in varied real-world scenarios.

In another study, the proposed system utilizes computer vision and deep learning methods to improve traffic flow and safety [12]. It is composed of three models and an alert system integrated into a graphical user interface (GUI). The dataset involved 8000 annotated images and was obtained from several open sources. For real-time vehicle detection and tracking, the first model uses the DeepSORT tracker in conjunction with the YOLOv5 algorithm. It reached 99.2%, as a (mAP) rate. Also, the YOLOv5 algorithm is employed by the second model, which forms the basis of the system, to precisely identify and categorize the degree of accident severity. It achieved an 83.3% mAP. The third model uses ResNet152 transfer learning algorithm to classify post-collision vehicular fires. It attained a mAP rate of 98.955%. Our research provides a distinct contribution by conducting a rigorous comparative analysis of YOLOv5 against its newer versions, YOLOv8 and YOLOv9, specifically for the task of accident detection. This focused evaluation is essential for determining the most efficient and accurate model for a dedicated accident detection system.

Radu et al. [13] proposed a traffic accident detection system through a video sequence from a first-person perspective of the dash-cam, which is more beneficial for the automated driving systems. Their primary contribution was comparing various architectures using a dataset of videos from the dash-cam. They used ResNet-50 and EffcientNet-b4 and compared their performance to each other, where ResNet achieved an accuracy rate of 78.58%, while EffcientNet-b4 achieved an accuracy rate of 76.76%. For the video classification, their architecture was based on ResNet and Long Short-Term Memory (LSTM). Furthermore, they demonstrated how incorporating a depth estimate module may increase the performance of the suggested classifier. Our study validates the performance of models like ResNet-101 and EfficientNetB7 on a significantly more diverse and extensive dataset from various angles and environments, making our findings more applicable to a wider range of real-world surveillance systems.

In another research paper, deep learning algorithms, VGG19 and VGG16, were employed to help insurance companies detect car accidents, evaluate the location of the accident and its degree of severity [14]. In order to improve the system accuracy, they apply a combination of transfer learning and L2 regularization to the VGG models. The accuracy of the VGG19 and VGG16 in damage detection reached 95.22% and 94.56% respectively. In damage localization, the accuracy of VGG19 reached 76.48% and 74.39% of VGG16. Our research focuses on the broader and more challenging task of general accident detection. Our VGG16 model achieves a high accuracy of 92% on a larger and more diverse dataset, demonstrating its robust ability to detect and classify a wider range of accident scenarios beyond just car damage.

Pathik et al. [15] highlighted the significance of addressing road accidents through the implementation of an accident detection and rescue system that is based on IoT and AI. It was built to help shorten delays in rescuing people in case of accidents. The system consists of sensors, GPS and GSM modules, controllers, a camera, and Raspberry Pi. The workflow of the intelligent system is composed of three phases. In the first stage, data related to accidents is collected through IoT equipment. In the second stage, deep learning-based models, such as ResNet-50 and InceptionResNetV2, are used to confirm the output of IoT. In the third stage, after confirming the

detection of the accident, information is sent to the rescue team and the family. Due to the integration of AI and IoT, the model has low false positives during testing and zero false positives during training. Building upon this concept of a multi-component system, our work contributes a detailed comparative analysis of six different deep learning models for both classification and detection tasks, providing a comprehensive performance evaluation that is crucial for selecting the optimal model for rapid accident detection.

In another study, they designed a network consisting of two streams for accident detection called Attention R-CNN [16]. The appearance stream is responsible for object detection using Faster-RCNN with modifications to maximize efficiency. Features extracted from the appearance stream's backbone are transformed into characteristic features via the characteristic stream, which is subsequently used for computing object characteristic properties. While the Attention R-CNN model is effective, our research provides a more comprehensive comparison of multiple single-stage and classification-based models, which is crucial for identifying the most efficient and accurate solution for real-world deployment.

Zantalis et al. [17] emphasized the integration of IOT and ML approaches in smart transportation, addressing a range of applications including accident prevention and detection, streetlights, route optimization, road anomalies, infrastructure, and parking. The research draws attention to the growing interest in route planning, parking, and accident detection as important ITS research topics. The research also highlighted a possible gap in ML coverage of applications for smart parking and lighting technologies. Its overall goal is to give an in-depth examination of current developments in various sectors and to point out potential topics for further study. Our project provides a concrete and detailed implementation that supports their findings. We offer a practical solution for real-time accident detection that leverages deep learning on visual data, thereby addressing a critical need they identified in their review of the field. Table II summarizes all the research mentioned above.

Although previous studies have made significant progress in applying machine learning and computer vision techniques for accident detection, they also present notable limitations. Several works relied on synthetic or limited datasets, which may not generalize well to diverse real-world environments. Others achieved high accuracy but lacked scalability or applicability in real-time scenarios, particularly under poor lighting or complex traffic conditions. Moreover, many studies focused on single models, making them less robust in handling different accident types and severities. IoT-based approaches showed promise in reducing false positives, but they require additional infrastructure, which limits widespread adoption. These limitations highlight the need for a more comprehensive approach that leverages multiple deep learning models and diverse datasets to enhance robustness, accuracy, and real-time applicability. This research aims to address these gaps by evaluating and comparing several deep learning architectures for accident detection, ultimately contributing to more reliable and efficient emergency response systems.

TABLE II. MACHINE LEARNING ALGORITHMS IN CAR ACCIDENT DETECTION AND CLASSIFICATION

No	Algorithm	Reference	Algorithm Type	Learning Type
1	MobileNetV2	[11]	CNN	Supervised learning
2	EfficientNetB1	[11, 13]	CNN	Supervised learning
3	YOLOv5	[12]	Object Detection	Supervised learning
4	ResNet	[12, 13]	CNN	Supervised learning
5	VGG	[14]	CNN	Supervised learning
6	InceptionResNetV2	[15]	CNN	Supervised learning
7	Faster-RCNN	[16]	Object Detection	Supervised learning

III. PROPOSED SYSTEM

The proposed system for detecting and classifying car accidents includes the following phases:

1) Data collection: Relevant data is gathered from open sources, including CCTV images of accidents, traffic images, and semi-realistic images from video games.

2) Data preprocessing: The data is split into training, testing, and validation sets, and cleaned to prepare it for analysis.

3) Data labeling: The collected data is labeled to indicate whether it represents an accident for classification and to assign bounding boxes for detection. This manual annotation is performed using Roboflow.

4) Training phase:

a) Car accident classification: Pre-trained models (e.g., VGG, ResNet, EfficientNet) are fine-tuned on the dataset. We modify the final layers to match the accident classes, freezing original weights while updating the new layers. Model performance is evaluated using accuracy, precision, and recall by applying it to a different test set. Fig. 1 illustrates the proposed system for classifying car accidents.

b) Car accident detection: A detection head is added to the pre-trained model to predict bounding box coordinates and class labels. Both the pre-trained model and detection head's weights are updated. Performance is assessed using metrics like mean Average Precision (mAP). We conducted an experimental evaluation comparing the performance of YOLOv5, YOLOv8, and YOLOv9 as object detection models. The proposed system for detecting car accidents is demonstrated in Fig. 2.

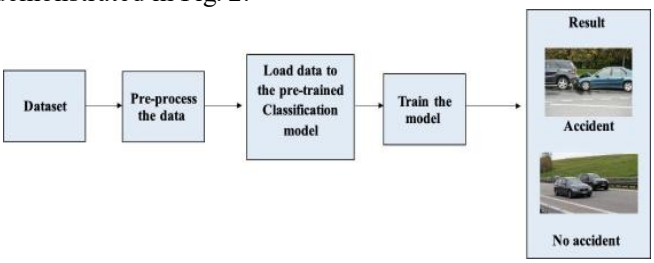


Fig. 1. The workflow of the system for car accident classification.

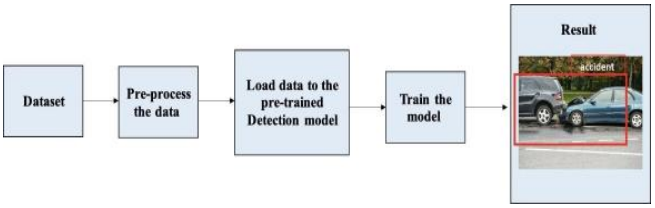


Fig. 2. The workflow of the system for car accident detection.

IV. Materials and Methods

A. Datasets

Sources: We took the initiative to create our dataset by collecting videos of car traffic and accidents in different countries from YouTube. These videos were converted into individual frames using the CV2 Python library. Additionally, we collected images from the extensive Mapillary Vistas Dataset [18]. It includes photos taken in different seasons, weather conditions, and daytime hours. To further augment our dataset, we incorporated data from Kaggle [19]. In order to enhance the diversity of the dataset, we also captured semi-realistic images from video games and car crash simulators [20]. This diverse collection helped us maintain a balanced distribution of data for both classes, "accident" and "no-accident", preventing over-fitting. Our dataset comprises a total of 1568 images. Each image had a size of 1280 by 720 pixels. Samples from our dataset for accident and non-accident images are illustrated in Fig. 3.

1) Dataset splitting: Our dataset was then divided into three main segments: training data, testing data, and a validation set. The training data is utilized to train the model while the testing data is reserved for evaluating the model's performance during testing. The model is adjusted, and its performance is assessed during training using the validation set. It facilitates the determination of the model's optimal hyper-parameters.

B. Algorithms and Models

The deep learning algorithms that have been pre-trained on large-scale datasets are useful for a variety of tasks. By leveraging the learned representations, pre-trained models offer transferable knowledge and efficient solutions for new tasks, enhancing generalization and reducing the need for extensive training from scratch. In this project, the following models were included:

1) VGG16

a) Overview: VGG16 is a deep learning model for computer vision and image recognition, consisting of 16 convolutional layers. It was developed by Simonyan and Zisserman [21]. Simonyan is known for its success in achieving a top 5 test accuracy of 92.7% on the ImageNet dataset.

b) Architecture: The model utilizes small 3x3 convolutional filters in successive layers, with ReLU activation functions, and includes three fully connected layers. Despite its simplicity and uniform structure, VGG16 is a complex network with approximately 138 million parameters.

2) ResNet-101:

a) Overview: ResNet, proposed by He et al. [22], in 2015 revolutionized deep learning by introducing residual connections (skip connections) that addressed the vanishing gradient problem, enabling the training of very deep neural networks with improved accuracy.

b) Architecture: ResNet comes in multiple versions, including ResNet-18, ResNet-34, ResNet-50, ResNet-101, and more, differing in the number of layers. ResNet-50 and later models use a bottleneck design for efficiency, while ResNet-101, with 101 layers, is known for its depth and effectiveness in complex recognition tasks.

(a) Accident image dataset.

(b) No Accident image dataset

(c) Rainy weather

(d) Semi-realistic image

Fig. 3. Images from the dataset.

3) EfficientNet

a) Overview: EfficientNet is a CNN architecture known for its exceptional efficiency and accuracy in computer vision tasks like image classification and object detection. It includes

seven models (EfficientNet-B0 to B7), each optimized for different performance and resource requirements.

b) Architecture: The architecture of EfficientNet begins with the EfficientNet-B0 model, designed using Neural Architecture Search (NAS). It features Mobile Inverted Bottleneck Convolution (MBConv) layers, which are optimized for efficiency, especially on mobile and resource-constrained devices.

4) YOLOv5:

a) Overview: YOLOv5, introduced in 2020 [23], is a leading single-stage object detection algorithm known for its efficiency and accuracy. It is an evolution of the YOLO (You Only Look Once) series, initially launched in 2016 [24], and is designed to be faster and more flexible. YOLOv5 is available in four variants (small, medium, large, and extra-large), offering a balance between accuracy, speed, and computational requirements.

b) Architecture: YOLOv5 utilizes a lightweight architecture based on Scaled-YOLOv4. It employs a single-stage object detection framework that integrates anchor boxes and anchor clustering for predicting bounding boxes and class probabilities. The model design involves selecting a deep neural network backbone (e.g., ResNet or EfficientNet) for feature extraction, followed by additional convolutional layers for feature fusion and multi-scale object detection. YOLOv5's architecture prioritizes simplicity and deployment flexibility.

5) YOLOv8:

a) Overview: YOLOv8, released by Glenn Jocher in January 2023 [25], maintains high accuracy while offering faster inference speeds and introduces developer-friendly features like an easy-to-use command-line interface (CLI) [26].

b) Architecture: YOLOv8 is built on a deep convolutional neural network (CNN) architecture, consisting of a backbone (CSPNet) for feature extraction and a head for prediction. It also features an anchor-free design, which reduces box prediction numbers and improves Non-Maximum Suppression (NMS) speed [27].

6) YOLOv9:

a) Overview: YOLOv9 is the latest iteration of the YOLO (You Only Look Once) real-time object detection system [28], building on the legacy of previous versions. It incorporates advanced deep learning techniques to achieve superior accuracy, speed, and efficiency in object detection tasks.

b) Architecture: YOLOv9 introduces a novel architectural design by integrating the Generalized ELAN (GELAN) architecture with Programmable Gradient Information (PGI). This combination enhances the model's performance, making it faster and more accurate for real-time object detection.

C. Software and Libraries

1) Programming language: Python was employed for model implementation due to its extensive support for machine learning and data analysis libraries [29].

2) Deep learning frameworks:

a) PyTorch: Used for training detection models. PyTorch was chosen for its dynamic computation graph and user-friendly interface [30].

b) Tensorflow: Used for training classification models. And it was selected due to its simplicity and popularity [31].

3) Development environment and platforms:

a) Google Colab: Leveraged for its cloud-based environment, which offers accelerated computation and ease of use without the need for local installations. Google Colab was instrumental in efficiently training Python-based models [32].

b) Kaggle: Datasets were obtained from Kaggle, a leading platform for data science and machine learning competitions, offering a variety of datasets, code, and resources crucial for training, and evaluating models [33].

c) Mapillary: It is a collection of street-level pictures that has been annotated into 66 object categories and includes 25000 high-resolution images from all over the world [18].

d) Roboflow: This platform was employed for object detection and classification tasks. Roboflow assists in developing computer vision models and provides functionalities for data augmentation and model training [34].

4) Libraries and tools:

a) NumPy: Utilized for numerical computations and handling data arrays.

b) OpenCV: Used for image processing tasks, such as resizing and augmenting images, that helps in developing the field of computer vision (CV) [35].

c) CV2: It is used for image and video processing tasks, such as reading and writing images, applying filters, detecting edges, and performing image transformations.

d) Torch: Widely used for tasks like computer vision, natural language processing, and reinforcement learning.

e) Torchvision: Provide tools specifically for computer vision tasks.

f) Matplotlib: Used for creating static, interactive, and animated visualizations in Python.

D. Computational Resources

To effectively train and evaluate the object detection models, we utilized the following:

a) MacOS: M1 pro chip, CPU: 10 core GPU: 14 core, Memory: 16G

b) Windows: CPU: Intel core i7, GPU: NVIDIA GetForce MX330 2GB, Memory: 16G

E. Performance Metrics

To evaluate the performance of our models, we used the following metrics:

1) Detection models: Mean Average Precision (mAP) was used as the primary metric to assess the accuracy of object detection models. We also considered Precision and Recall.

2) Classification models: For classification models, we used Accuracy, Precision, and Recall [36].

V. RESULTS

The findings of the detection and classification models used in this research to precisely identify and detect car accidents are presented in this section. It provides a comprehensive analysis of the models' performance and effectiveness in classifying and detecting car accidents in real-world scenarios. The following subsections outline the evaluation metrics, the classification model's performance in identifying accident-related images, and the object detection model's effectiveness in localizing the involved vehicles. The results demonstrate the models' efficacy in automating the detection and classification process, contributing to more efficient accident response and improved road safety.

A. Classification

The models presented were trained using the TensorFlow deep learning framework. The categorical classes targeted by the model were "Accident" and "No-Accident". The training process utilized the Adam optimizer [37], a popular gradient-based optimization algorithm, to update the model's parameters. To evaluate the performance of our models, we employed standard evaluation metrics including accuracy, recall, and precision, and visualized these metrics using plots.

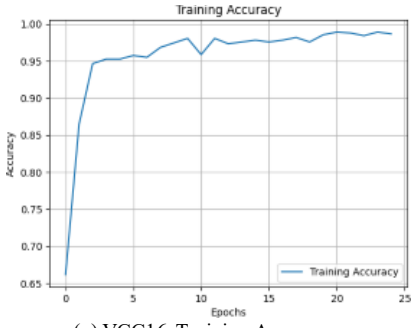
In the subsequent section, we will present the results obtained from the evaluation of classification models. The different versions of VGG16, EfficientNetB7, and ResNet101 were conducted with a consistent configuration, employing a batch size of 100, and 25 epochs for training. The proposed deep learning models were trained using TensorFlow and implemented within the Google Colab environment. As part of the training process, all training and test images were uniformly resized to dimensions of 224 x 224. Table III summarizes the results achieved in classification models with a detailed illustration of the training parameters.

TABLE III. CAR ACCIDENT CLASSIFICATION: OPTIMAL EXPERIMENTAL RESULTS

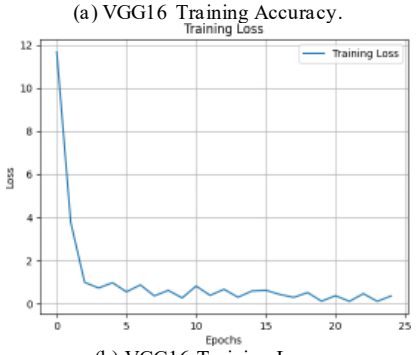
Algorithm	Epoch/ Iteration	Batch Size	Accuracy	Recall	Precision
ResNet101	25	100	93%	96%	96%
EfficientNetB7	25	100	91%	96%	96%
VGG16	25	100	92%	98%	96%

The VGG16 model, trained with 25 epochs, exhibited an impressive accuracy of 0.92. The accuracy and loss results for the training and validation are shown in Fig. 4. With a precision and a recall of 0.96 and 0.98, respectively shown in Fig. 7(a) and 8(a) depicts the confusion matrix for the VGG16 model during the training and testing phases.

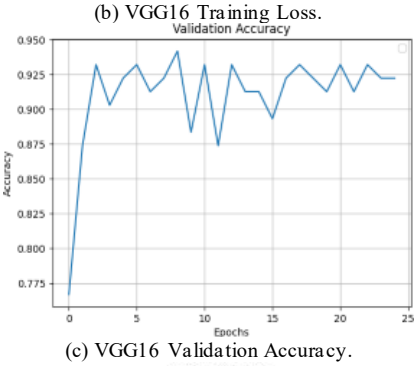
Similarly, the EfficientNetB7 model achieved an accuracy of 0.91 after 25 epochs of training. The accuracy and loss trends for both training and validation iterations are illustrated in Fig. 5. With a precision and a recall of 0.96 and 0.96, respectively shown in Fig. 7(b). The confusion matrix for the EfficientNetB7 model is shown in Fig. 8(b).



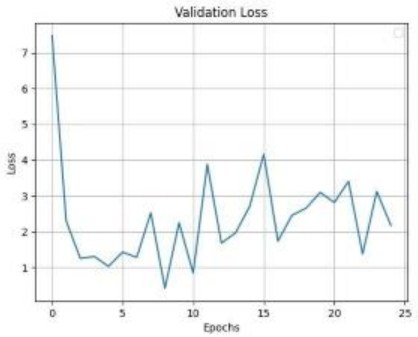
(a) VGG16 Training Accuracy.



(b) VGG16 Training Loss.



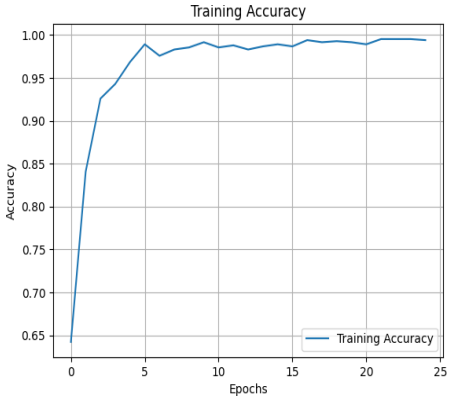
(c) VGG16 Validation Accuracy.



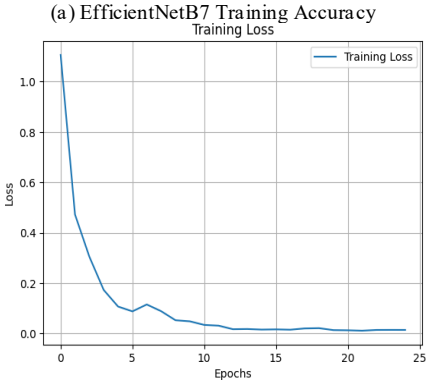
(d) VGG16 Validation Loss.

Fig. 4. Visualizing accuracy and loss of training and validation for VGG16.

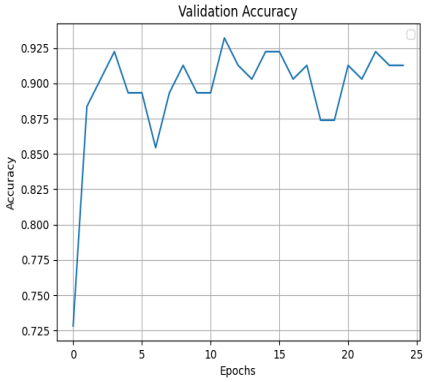
Lastly, the ResNet101 model attained accuracy of 0.93 after 25 epochs of training. The accuracy and loss curves for training and validation iterations are plotted in Fig. 6. With a precision and a recall of 0.96 and 0.96 respectively shown in Fig. 7(c). The confusion matrix for the ResNet101 model is presented in Fig. 8(c).



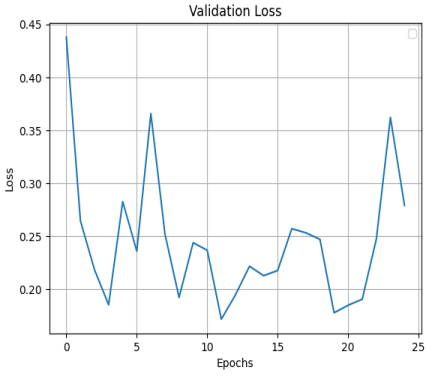
(a) EfficientNetB7 Training Accuracy



(b) EfficientNetB7 Training Loss

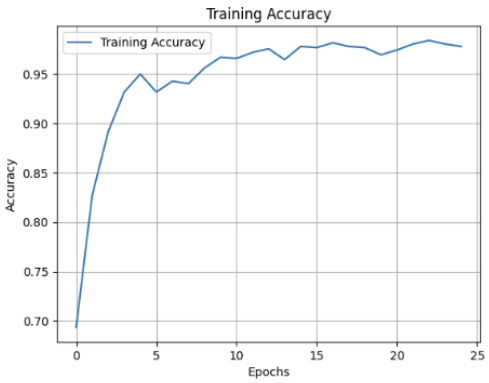


(c) EfficientNetB7 Validation Accuracy

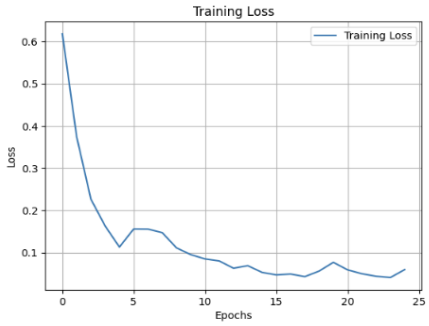


(d) EfficientNetB7 Validation Loss

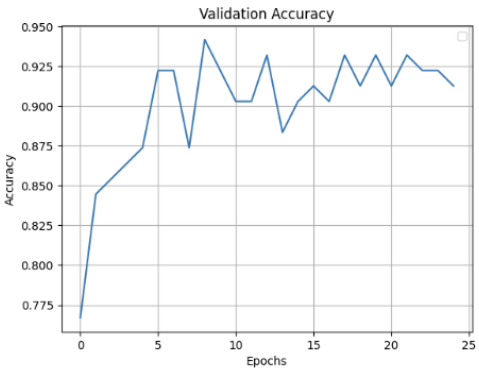
Fig. 5. Visualizing accuracy and loss of training and validation for EfficientNetB7.



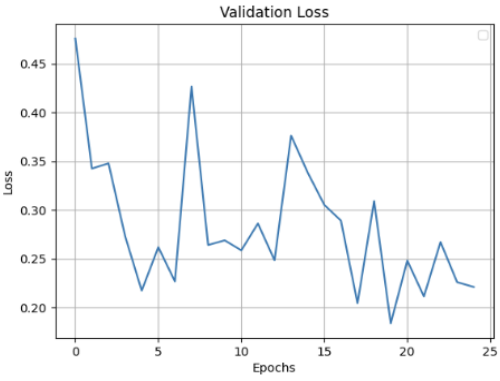
(a) ResNet101 Training Accuracy



(b) ResNet101 Training Loss

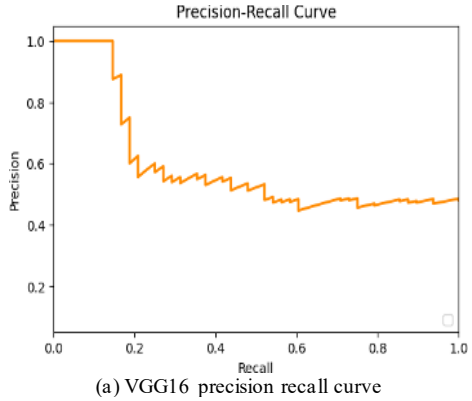


(c) ResNet101 Validation Accuracy

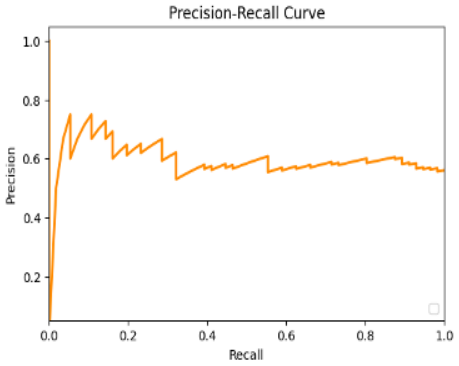


(d) ResNet101 Validation Loss

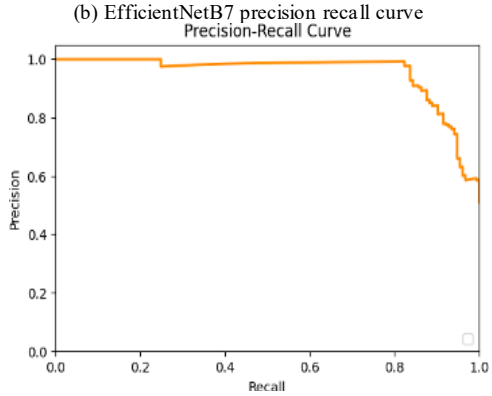
Fig. 6. Visualizing accuracy and loss of training and validation for ResNet101.



(a) VGG16 precision recall curve

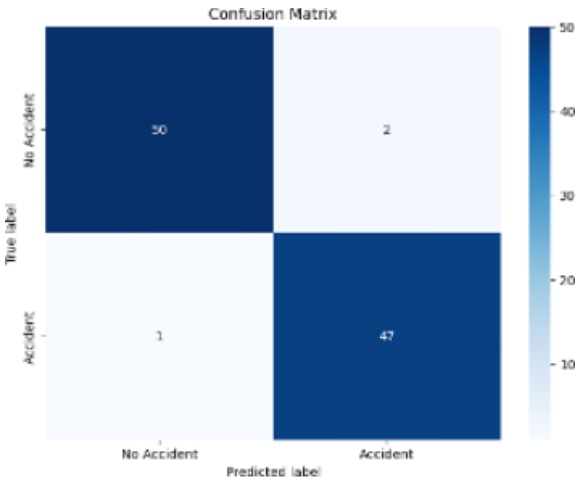


(b) EfficientNetB7 precision recall curve

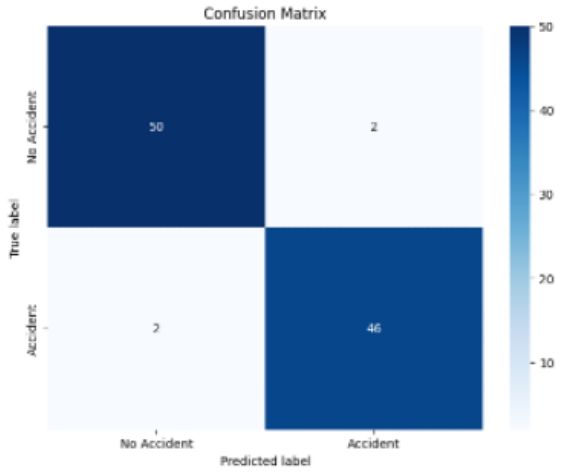


(c) ResNet101 precision recall curve

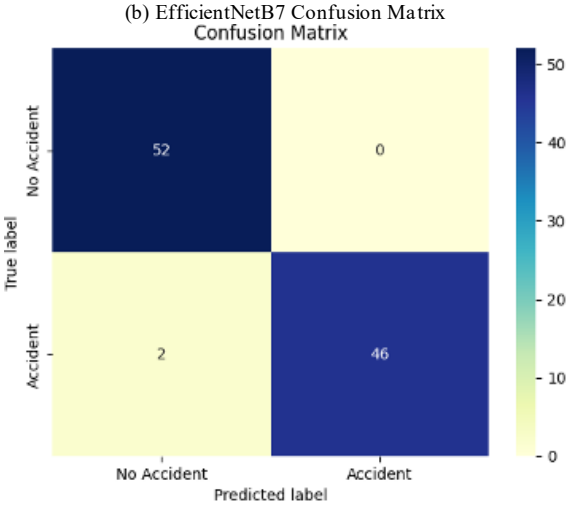
Fig. 7. Visualizing precision recall curve for VGG16, EfficientNetB7 and ResNet101.



(a) VGG16 Confusion Matrix



(b) EfficientNetB7 Confusion Matrix



(c) ResNet101 Confusion Matrix

Fig. 8. Visualizing confusion matrix for VGG16, EfficientNetB7 and ResNet101.

B. Detection

In the subsequent section, we will present the results obtained from the evaluation of detection models. The different versions of YOLO were conducted with a consistent configuration, employing a batch size of 16 and 100 epochs for training. The proposed deep learning models were trained using Pytorch and implemented within the Google Colab environment. As part of the training process, all training and test images were uniformly resized to dimensions of 640 x 640. summarization of the results achieved in detection models with a detailed illustration of the training parameters is shown in Table IV.

TABLE IV. CAR ACCIDENT DETECTION OPTIMAL EXPERIMENTAL RESULTS

Algorithm	Epoch/ Iteration	Batch Size	mAP	Recall	Precision
YOLOv5	100	16	97.8%	96.6%	95.7%
YOLOv8	100	16	95.3%	90.8%	94%
YOLOv9	100	16	95.7%	93.4%	94.1%

The YOLOv5 model, trained with 100 epochs, exhibited an impressive mean Average Precision (mAP) of 0.978. The mAP and loss results for 100 of the training and testing iterations shown in Fig. 9. With a precision and a recall of 0.957 and 0.966 respectively shown in Fig. 10(a) and (b). Fig. 11 depicts the confusion matrix for the YOLOV5 model during the training and testing phases.

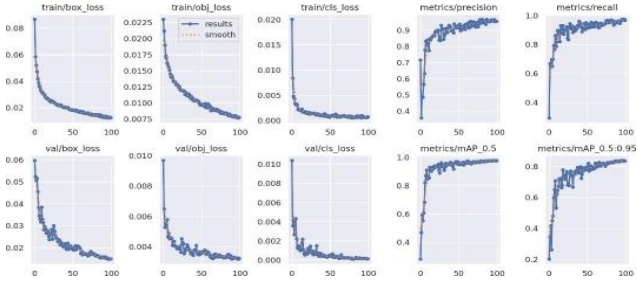
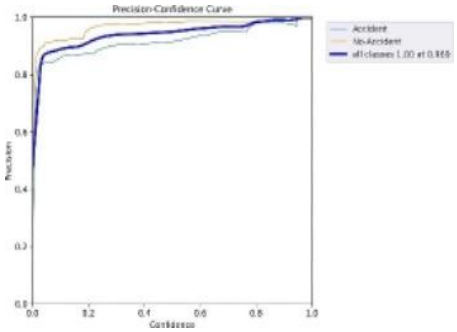
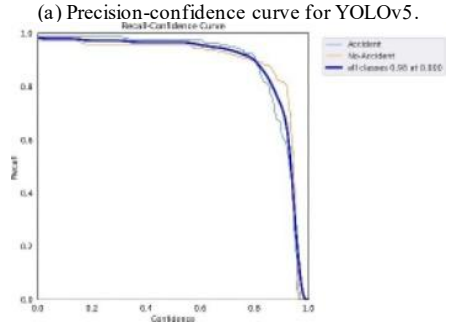


Fig. 9. mAP, training loss and validation loss for YOLOv5.



(a) Precision-confidence curve for YOLOv5.



(b) Recall-confidence curve for YOLOv5

Fig. 10. Visualizing the precision-confidence and recall-confidence curves of YOLOv5.

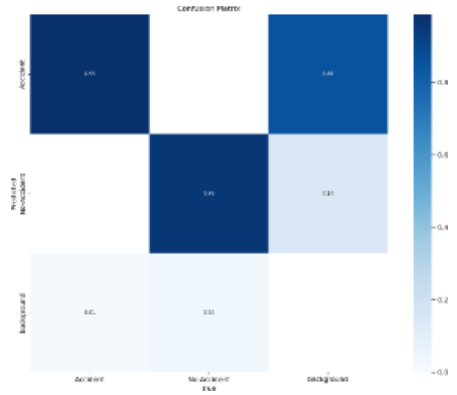


Fig. 11. Confusion matrix for YOLOv5.

The evaluation of YOLOv8 and YOLOv9 models yielded noteworthy findings regarding their mean Average Precision (mAP) values. YOLOv8 demonstrated a commendable mAP score of 0.953 in Fig. 12. Meanwhile, YOLOv9 demonstrated a notably elevated mean Average Precision (mAP) of 0.957 as illustrated in Fig. 15. Additionally, YOLOv9 showcased superior recall performance in Fig. 16(b) with a score of 0.934, whereas YOLOv8 achieved a recall value of 0.908 in Fig. 13(b). In terms of precision, YOLOv9 achieved a score of 0.941 as shown here in Fig. 16(a), while YOLOv8 attained a precision value of 0.94 in Fig. 13(a). The confusion matrices for YOLOv8 and YOLOv9 are presented in Fig. 14 and 17, respectively, providing a comprehensive visual representation of the model's performance.

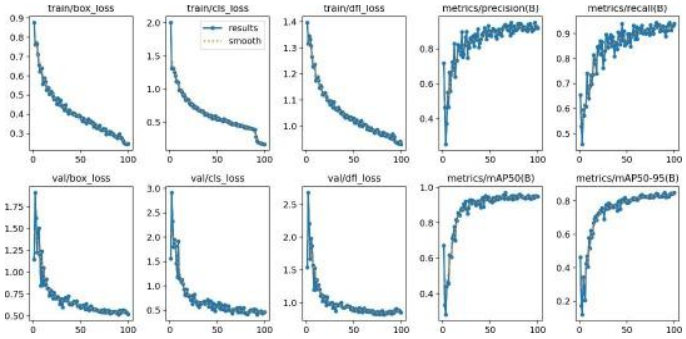
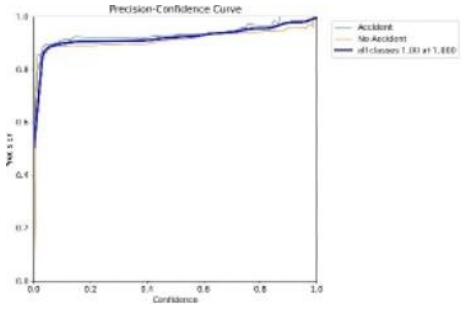
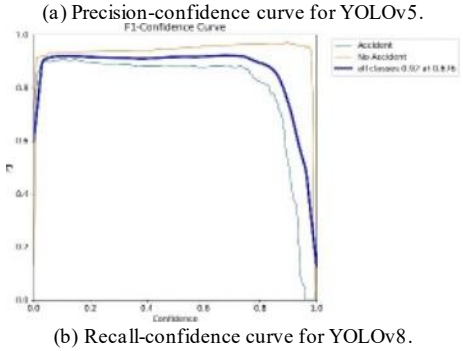


Fig. 12. mAP, training loss and Validation loss for YOLOv8.



(a) Precision-confidence curve for YOLOv5.



(b) Recall-confidence curve for YOLOv8.

Fig. 13. Visualizing the precision-confidence and recall-confidence curves of YOLOv8.

The superior performance of our YOLOv5 model compared to the newer YOLOv8 and YOLOv9 versions is a noteworthy finding. While YOLOv8 and YOLOv9 incorporate more advanced architectural features, such as the anchor-free design and the Generalized ELAN (GELAN) architecture, our results suggest that these innovations did not provide a significant advantage on our specific dataset. We hypothesize that the

simpler, well-optimized framework of YOLOv5 proved to be exceptionally effective at learning the nuanced features of our mixed dataset. This indicates that while newer models offer theoretical improvements, an earlier, well-optimized model can still achieve state-of-the-art performance, especially when its architecture is well-suited to the characteristics of the training data.

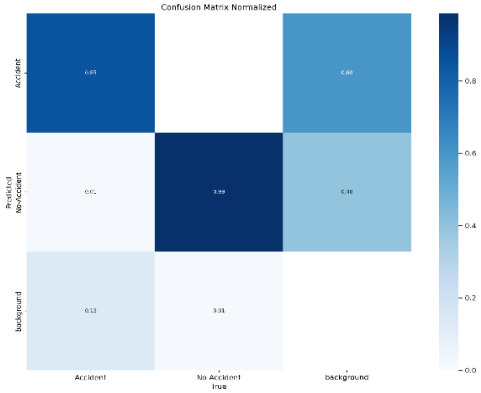


Fig. 14. Confusion matrix for YOLOv8.

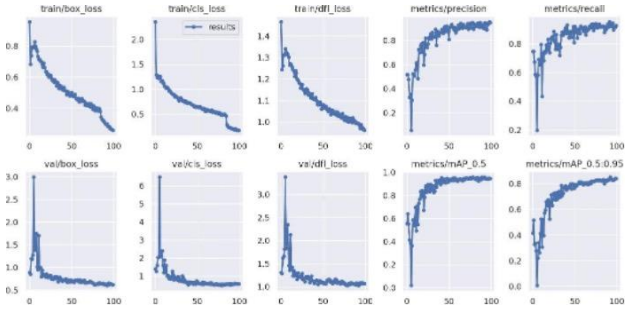
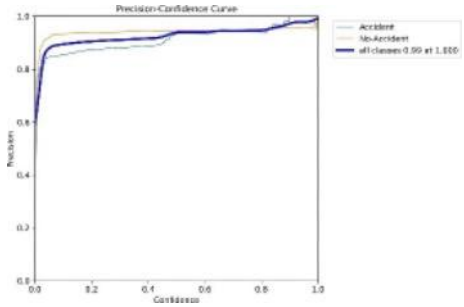
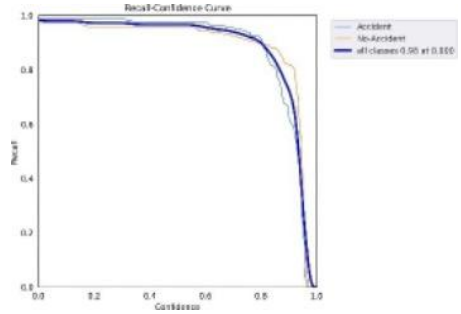


Fig. 15. mAP, training loss and validation loss for YOLOv9.



(a) Precision-confidence curve for YOLOv9.



(b) Recall-confidence curve for YOLOv9

Fig. 16. Visualizing the precision-confidence and recall-confidence curves of YOLOv9.

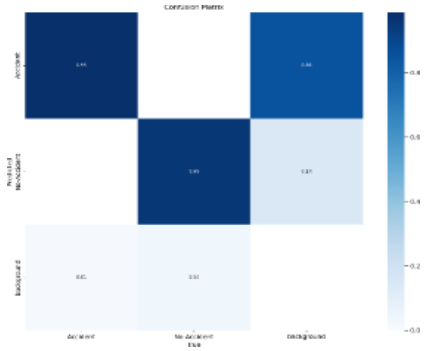


Fig. 17. Confusion matrix for YOLOv9.

VI. Discussion

The results obtained from our experiments demonstrate the effectiveness of the proposed system. A key contribution of this work is the comprehensive comparative analysis of multiple state-of-the-art deep learning models on a diverse, custom-built dataset. This approach not only identifies the most effective model for the task but also provides crucial insights into the performance trade-offs between model architecture, training time, and dataset characteristics, which is often a missing element in prior research.

According to study findings and prior research, traffic accidents rank among the top causes of death and disability worldwide. This highlights the necessity of improving urban safety through rapid accident detection and quick response. Our proposed framework contributes to solving these problems and achieving the safety of roads and communities. We trained and evaluated different deep learning models for car accident classification and detection on an extensive dataset of labeled images for "accident" and "no-accident" to assess their performance. The results obtained from our experiments demonstrate the effectiveness of the proposed system. In this section, analogous frameworks that have been conducted will be comprehensively compared.

A. Classification

In this section, we evaluate the performance of our proposed classification models:

ResNet-101, EfficientNetB7, and VGG16. The evaluation is carried out using precision, recall, and accuracy metrics which provide a comprehensive understanding of the models' classification capabilities. ResNet-101 achieved a precision and recall of 96%, with an accuracy of 93%. EfficientNetB7 exhibited a similar precision and recall of 96%, but with a slightly lower accuracy of 91%. VGG16 demonstrated a precision of 96% and a higher recall of 98%, with an accuracy of 92%. Based on these results, all three models showcase high precision values, indicating their ability to accurately classify positive instances. However, VGG16 stands out with a higher recall, suggesting its proficiency in correctly identifying positive instances while minimizing false negatives. When considering accuracy, ResNet-101 achieved the highest value of 93%, closely followed by VGG16 at 92% and EfficientNetB7 at 91%. The chart in Fig. 18 presents a comparative analysis of classification models, highlighting their performance in terms of Precision, Recall, and Accuracy metrics.

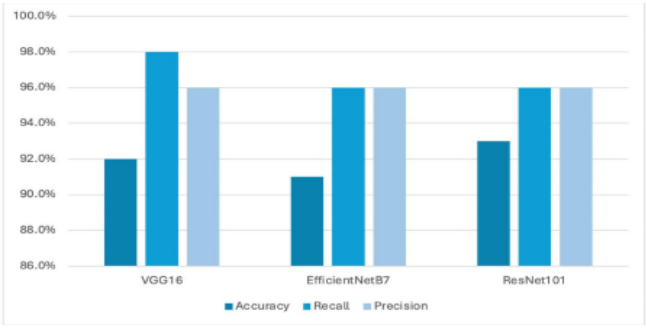


Fig. 18. Comparative analysis of classification models.

To assess the performance of our proposed model, we conducted a comparative analysis with a previously published paper [15]. The referenced paper employed the ResNet architecture for training on a dataset consisting of 500 images. They utilized ResNet-50 with a batch size of 64, a learning rate of 0.0001, and trained the model for 50 epochs. Stochastic Gradient Descent (SGD) was applied as an optimizer. The achieved accuracy was an impressive 99.3%. In our study, we trained our proposed model throughout 25 epochs. Our model attained an accuracy of 93%. This 6.3% decrease in accuracy can be attributed to several critical factors. The study's smaller, more specialized dataset likely led to a higher accuracy score due to its focused nature, while our model's slightly lower score reflects its ability to generalize to a broader range of scenarios and variations present in our diverse dataset. Additionally, the longer training duration (50 epochs) in the cited study may have allowed the model to fine-tune its parameters more thoroughly. Our comprehensive evaluation highlights the trade-offs between dataset size, diversity, training time, and overall model performance, underscoring the value of our approach on a more robust dataset.

In [11], the performance of EfficientNetB1 was evaluated using a dataset comprising 164 images. The model was trained for 300 epochs, employing a batch size of 32. The results of the study revealed that the EfficientNetB1 model achieved a commendable mAP of 89%. Contrastingly, our proposed EfficientNetB7 model surpassed the mAP reported in the study by achieving an accuracy of 91%, indicating an improvement of 2%. This is a significant finding given that our model was trained with a much shorter duration of only 25 epochs. This indicates that our model, leveraging a more advanced architecture and a larger, more diverse dataset, demonstrates enhanced performance and superior efficiency, making it a more viable solution for real-world applications where training time is a critical factor.

In another study [14], the VGG16 model was trained using a dataset consisting of 1150 images depicting damaged cars. The accuracy achieved in damage detection task was reported to be 94.56%. Notably, the study also focused on damage localization, where the VGG16 model achieved an accuracy of 74.39%. In comparison, our proposed system demonstrated encouraging results, achieving an accuracy of 92% in accident detection. This indicates that our model performs at a similar level of effectiveness as the VGG16 model utilized in the cited study, albeit with a larger dataset. The differences in dataset composition, quality, and diversity could impact the model's performance. It is possible that the dataset used in the study [14] provided a more targeted representation of damaged car images, leading to higher accuracy. Our model's ability to achieve high accuracy on a larger, more diverse dataset highlights its robust performance and adaptability to a wider range of accident scenarios. Table V summarizes the comparison between our research and other studies.

TABLE V. COMPARISON TABLE OF CLASSIFICATION MODELS

Model	Accuracy/mAP	Epochs	Batch size	Dataset
ResNet-50 [15]	99.3%	50	64	500
ResNet-101	93%	25	100	1568
EfficientNetB1 [11]	89%	300	32	164
EfficientNetB7	91%	25	100	1568
VGG16 [14]	94.56%	-	-	1150
VGG16	92%	25	100	1568

B. Detection

The evaluation conclusions show that YOLOv5 exceeds both YOLOv8 and YOLOv9 in terms of mAP, recall, and precision. It achieved the highest mAP score of 97.8%, demonstrating that it can accurately detect objects. YOLOv8 and YOLOv9 achieved slightly lower mAP scores of 95.3% and 95.7%, respectively. In terms of recall, YOLOv5 scored 96.6%, outperforming YOLOv8 and YOLOv9. This indicates that YOLOv5 performs more effectively at capturing a greater proportion of relevant objects. Similarly, YOLOv5 achieved an outstanding precision score of 95.7%, which indicates its ability to reliably detect objects with a high level of confidence. YOLOv9 had a precision score of 94.1%, which was somewhat lower than YOLOv5, while YOLOv8 had a precision score of 94%. The models were trained on the same training setup including dataset size, epochs, batch size, and image size. we can make a more direct comparison of the training times. YOLOv5 took approximately 0.408 hours to train, YOLOv8 required around 0.924 hours, and YOLOv9 took approximately 2.027 hours

Overall, YOLOv5 achieved the highest mAP, recall, and precision scores among the three models while having the shortest training duration. YOLOv8, despite slightly lower mAP, recall, and precision scores compared to YOLOv5, requires longer training time, suggesting it may not significantly outperform YOLOv5 in performance. YOLOv9, on the other hand, achieved competitive performance metrics, particularly in recall and precision, but its long training time may be a concern in time-sensitive applications or resource-constrained environments. The chart in Fig. 19 presents a comparative analysis of detection models, highlighting their performance in terms of Precision, Recall, and mAP metrics.

According to YOLOv5, we performed a comparison study with prior research [12]. The cited research utilized the YOLOv5 model trained on 8000 images of vehicles extracted from CCTV footage to detect and track vehicles. The model was trained in 900 epochs, with a batch size of 16 achieving 99.2% mAP. Additionally, the YOLOv5 was employed to detect accident severity, and it achieved 83.3% mAP. In contrast, our

YOLOv5 model was trained on 1568 annotated images of car accidents collected from several sources. Our model attained 97.8% mAP by training it with 100 epochs. These comparative results highlight the superior performance of the YOLOv5 model. This is a significant finding that highlights the importance of a carefully curated dataset. Despite using less data and a shorter training time, our model's strong performance demonstrates that the quality and diversity of our dataset—which includes a wide range of real-world and simulated accident scenarios—is a more critical factor for achieving high-performance results than sheer data quantity. Furthermore, the efficient YOLOv5 architecture, which combines high accuracy with real-time inference speeds, proved to be particularly well-suited for our dataset and task, making it a viable choice for real-world applications.

Another study employed YOLOv8 to detect windshields as a subsystem of automated detection of drivers and passengers without seat belts [38]. The dataset includes 3289 video frame images of windshields labeled for use in the windshield detection. The model trained in 100 epochs with a batch size of 16 and achieved 95% mAP. Conversely, our YOLOv8 model was trained using 1568 images in 100 epochs with a batch size of 16, achieving an impressive 95.3% mAP in the more complex task of car accident detection. This is a significant finding that highlights the importance of a well-curated dataset. While the cited study focused on a single object within a larger system, our model's performance on a smaller, yet more diverse and challenging dataset demonstrates its robust generalizability across different detection tasks. This proves that a high-quality, representative dataset can be more effective than a larger but more specialized one, a crucial insight for real-world applications where data collection can be a major challenge. The comparison of our study with previous studies is summarized in Table VI.

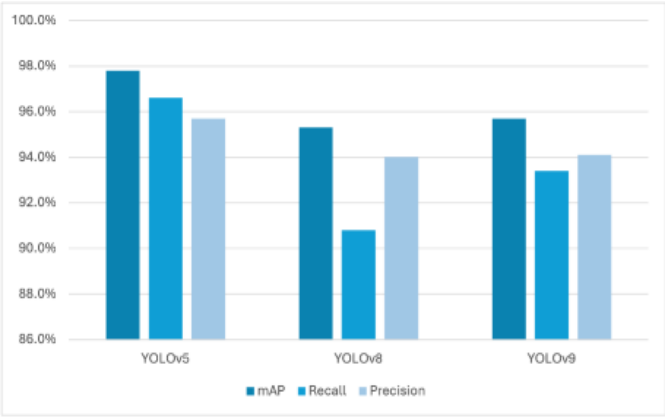


Fig. 19. Comparative analysis of detection models.

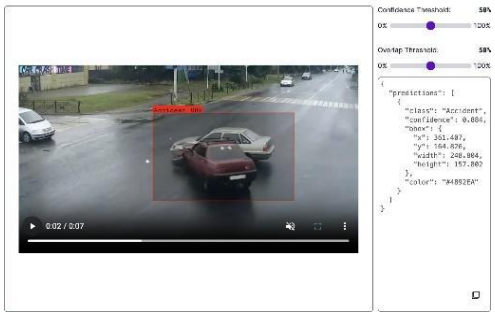
TABLE VI. COMPARISON TABLE OF DETECTION MODELS

Model	mAP	Epochs	Batch size	Dataset
YOLOv5 [12]	99.2%	900	16	8000
YOLOv5	97.8%	100	16	1568
YOLOv8 [38]	95%	100	16	3289
YOLOv8	95.3%	100	16	1568

C. Online Deployment

Considering the superior performance demonstrated by YOLOv5, a hosted video inference API developed by Roboflow is used to deploy the generated model [39]. This integration opens up a realm of possibilities for leveraging the robust capabilities of YOLOv5 in real-world applications, empowering users to harness its exceptional object detection and tracking capabilities for a wide range of video analysis tasks. The model demonstrates a remarkable ability to detect accidents within videos by meticulously analyzing individual frames and accurately assigning class labels to the objects present in the video.

Fig. 20(a) and (b) demonstrated the output of deploying the video on the hosted inference API. Each identified object is accompanied by a bounding box that precisely delineates its spatial positioning within the frames.



(a) Accident output.

(b) No-Accident output.

Fig. 20. YOLOv5 deployment on inference API.

D. Practical Implications, Limitations, and Future Direction

The findings of this research, particularly the superior performance demonstrated by YOLOv5, have significant practical implications for real-world deployment in smart cities. The model's reliability in accurately identifying and classifying car accidents is a critical first step in automating emergency response. The short training time for YOLOv5 is also a crucial factor, making it a viable solution for real-time applications and resource-constrained environments. In a smart city context, this model can be integrated with a camera and a notification system. This automated system for accident detection and classification can then swiftly notify medical facilities and security of any accidents. This would allow emergency services to react more quickly and efficiently, potentially saving lives and reducing the severity of injuries.

The findings of this research, while promising, are subject to certain limitations that warrant consideration for future work.

Firstly, the models' effectiveness may be hampered by factors such as low-quality footage or restricted camera coverage in real-world scenarios. Secondly, the computational resources required for training complex deep learning models can be a significant obstacle, as seen with our attempts to implement the Single Shot Detector (SSD) model. This limitation may affect the model's ability to reliably detect all types of accidents and can impact its effectiveness in time-sensitive applications.

The next logical step is to move from theoretical evaluation to practical implementation. This involves integrating the trained model with a camera and a notification system. Future research will also focus on expanding the dataset to include a wider variety of accident types, such as multi-vehicle collisions or accidents involving pedestrians, to improve the model's versatility. Finally, while our research focused on object detection and classification, exploring the integration of other data sources, such as real-time vehicle telemetry data, could provide a more comprehensive and accurate accident detection system in the future.

VII. Conclusions

To conclude, using different deep learning algorithms in the realm of accident detection and classification holds immense potential to enhance safety, and offers a proactive approach to accident detection, response, and prevention. In our project, we concentrated on employing diverse artificial intelligence algorithms, particularly algorithms within the realm of deep learning, to attain optimal outcomes aligned with the objectives of our project. We have employed six of the top computer vision models three of them specializing in classification task and the rest in object detection task are included to achieve our aim which is to successfully help detect and reduce accidents, thereby preserving more lives and decreasing the annual fatality rate. During the training phase, we observed satisfactory results across both classification and detection tasks. In the classification task, we obtained the following results: VGG16 model (accuracy of 0.92, precision 0.96, and recall 0.98). EfficientNetB7 model (accuracy of 0.91, precision 0.96, and recall 0.96). ResNet101 model (accuracy of 0.93, precision 0.96, and recall 0.96).

In the detection task we obtained the following results: YOLOv5 model (mAP of 0.978, precision 0.957, and recall 0.966). YOLOv8 model (mAP of 0.953, precision 0.94, and recall 0.908). YOLOv9 model (mAP 0.95, precision 0.941, and recall 0.934). According to classification results, The VGG16, EfficientNetB7, and ResNet101 models demonstrated high precision scores, with VGG16 exhibiting superior recall performance. In terms of accuracy, ResNet101 achieved the highest score among the models. And according to detection results, YOLOv5 exhibited superior performance with the highest mAP, recall, and precision scores among the three models, along with the shortest training duration.

These results underscore the efficacy of our approach in accident detection and classification tasks. The robust performance metrics, including high accuracy rates and recall scores, alongside the promising generalization capabilities of our models, highlight their potential for real-world deployment and scalability. Overall, our findings emphasize the vi ability and effectiveness of leveraging deep learning algorithms to enhance safety measures and mitigate the impact of road accidents.

Acknowledgment

The authors extend their appreciation to the Deanship of Scientific Research and Graduate Studies at Taif University, Saudi Arabia, for funding this research.

Data Availability Statement

The data that support the findings of this study are available on request from the corresponding author.

Abbreviations

The following abbreviations are used in this manuscript:

mAP - mean average precision.

SGD - Stochastic Gradient Descent.

TLA - Three letter acronym.

LD - Linear dichroism.

References

[1] MOI, "The general department of traffic (GDT) 2023." 2024. Available: https://www.moi.gov.sa/wps/portal/Home/sectors/publicsecurity/traffic/trafficriyadh/contents/!ut/p/z1/jZFPT8MwDMW_yjj0HC_JtnK0OmhCJ43ACiUXZHUbGLGMlQjYt6eHXfizrj7Z0u_Z1nvCi0r4QB_8RJG3gV7b-cGPH8FqbYZaFvki04Auu3aFXUhIx-L-JwCYXgKiujNm4uS8kML30Wc5Gj2ZAaSzfAQWTXlz7pQCVP30cKQQeuqPA757_S014urUkdZFftntPApfb0NcfUVRxYbWa64TODSDhve0fB5Qk8CGOCRAdc3LVYjv7ZO-8wToP8DUjVqguJjbqZKA8jfwT1CdQDo8AF1RnTLrbVOWZfW5V8CWLZ59A24-PZ0!/dz/d5/L0lHSkovd0RNQUprQUVnQSEhLzROVkUvZW4!/.

[2] WHO, "Reducing road crash deaths in the Kingdom of Saudi Arabia 2023." Available: https://www.who.int/news/item/20-06-2023-reducing-road-crash-deaths-in-the-kingdom-of-saudi-arabia.

[3] M. Mohammed, M. B. Khan, and E. B. M. Bashier, Machine Learning: Algorithms and Applications. Boca Raton: CRC Press, 2016.

[4] Z. Cui, K. Henrickson, R. Ke, and Y. Wang, "Traffic graph convolutional recurrent neural network: A deep learning framework for network-scale traffic learning and forecasting," IEEE Trans. Intell. Transp. Syst., vol. 21, no. 11, pp. 4883–4894, November 2020.

[5] H. Luo, Q. Kong, and F. Wu, "Traffic sign image synthesis with generative adversarial networks," in 2018 24th Int. Conf. Pattern Recognit. (ICPR), Beijing, China, IEEE, 2018, pp. 2540–2545.

[6] J. Zhang, Y. Zheng, and D. Qi, "Deep spatio-temporal residual networks for citywide crowd flows prediction," *Proceedings of the 31st AAAI Conference on Artificial Intelligence (AAAI '17)*, San Francisco, CA, USA, Feb. 4–9, 2017, pp. 1655–1661

[7] M. A. Quddus, W. Y. Ochieng, and R. B. Noland, "Current map-matching algorithms for transport applications: state-of-the art and future research directions," Transp. Res. C Emerg. Technol., vol. 15, no. 5, pp. 312–328, October 2007.

[8] Y. Hou, P. Edara, and C. Sun, "Traffic flow forecasting for urban work zones," IEEE Trans. Intell. Transp. Syst., vol. 16, no. 4, pp. 1761–1770, August 2015.

[9] D. Q. Tran and S. H. Bae, "Proximal policy optimization through a deep reinforcement learning framework for multiple autonomous vehicles at a non-signalized intersection," Appl. Sci., vol. 10, no. 16, p. 5722, August 2020.

[10] A. O. Philip and R. K. Saravanaguru, "Multisource traffic incident reporting and evidence management in Internet of vehicles using machine learning and blockchain," Eng. Appl. Artif. Intell., vol. 117, p. 105630, January 2023.

[11] T. Tamagusko, M. G. Correia, M. A. Huynh, and A. Ferreira, "Deep learning applied to road accident detection with transfer learning and synthetic images," Transp. Res. Procedia, vol. 64, pp. 90–97, January 2022.

[12] M. I. B. Ahmed, R. Zaghdoud, M. S. Ahmed, R. Sendi, S. Alsharif, J. Alabdulkarim, B. A. A. Saad, R. Alsabt, A. Rahman, and G. Krishnasamy, "A real-time computer vision based approach to detection and classification of traffic incidents," Big Data Cogn. Comput., vol. 7, no. 1, p. 22, January 2023.

[13] V. Radu, M. Nan, M. Trăscău, D. T. Iancu, A. Ş. Ghiţă, and A. M. Florea, "Car crash detection in videos," in 2021 23rd Int. Conf. Control Syst. Comput. Sci. (CSCS), Bucharest, Romania, IEEE, 2021, pp. 127–132.

[14] P. M. Kyu and K. Woraratpanya, "Car damage detection and classification," in Proc. 11th Int. Conf. Advances Inf. Technol., Bangkok, Thailand, Association for Computing Machinery, 2020, pp. 1–6.

[15] N. Pathik, R. K. Gupta, Y. Sahu, A. Sharma, M. Masud, and M. Baz, "AI enabled accident detection and alert system using IoT and deep learning for smart cities," Sustainability, vol. 14, no. 13, p. 7701, May 2022.

[16] T. N. Le, S. Ono, A. Sugimoto, and H. Kawasaki, "Attention R-CNN for accident detection," in 2020 IEEE Intell. Vehicles Symp. (IV), Las Vegas, NV, USA, IEEE, 2020, pp. 313–320.

[17] F. Zantalis, G. Koulouras, S. Karabetsos, and D. Kandris, "A review of machine learning and IoT in smart transportation," Future Internet, vol. 11, no. 4, p. 94, March 2019.

[18] Mapillary Vistas, "Mapillary vistas dataset." Available: https://www.mapillary.com/dataset/vistas.

[19] Kaggle, "Accident detection from CCTV footage." 2020. Available: https://www.kaggle.com/datasets/ckay16/accident-detection-from-cctv-footage/data.

[20] "Car crash simulator." 2022.

[21] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," arXiv preprint arXiv:1409.1556, 2014.

[22] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in Proc. IEEE Conf. Comput. Vision Pattern Recognit. (CVPR), IEEE, 2016, pp. 770–778.

[23] G. Jocher, A. Stoken, J. Borovec, L. Changyu, A. Hogan, L. Diaconu, J. Poznanski, L. Yu, P. Rai, R. Ferriday, and T. Sullivan, "ultralytics/yolov5: v3. 0, Zenodo." 2020. Available: https://zenodo.org/records/3983579.

[24] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You Only Look Once: Unified, real-time object detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Las Vegas, NV, USA, Jun. 27–30, 2016, pp. 779–788.

[25] J. Solawetz, "What is YOLOv8? The ultimate guide." Available: https://blog.roboflow.com/what-is-yolov8/.

[26] Encord, "YOLOv8 for object detection explained [Practical example]." 2023. Available: https://medium.com/cord-tech/yolov8-for-object-detection-explained-practical-example-23920f77f66a.

[27] H. Sahota, "The history of YOLO object detection models from YOLOv1 to YOLOv8." Available: https://blogs.nvidia.com/blog/category/enterprise/deep-learning/.

[28] A. Acharya, "YOLOv9: SOTA object detection model explained." Available: https://encord.com/blog/yolov9-sota-machine-learning-object-dection-model/.

[29] Python.org 2023, "Python organiziton." Available: https://www.python.org/.

[30] N. Ketkar and J. Moolayil, "Introduction to PyTorch," in Deep Learning with Python: Learn Best Practices of Deep Learning Models with PyTorch, N. Ketkar and J. Moolayil, Eds. Berkeley, CA: Apress, 2021, pp. 27–91.

[31] TensorFlow, "An end-to-end platform for machine learning." Available: https://www.tensorflow.org/.

[32] Colab, "Google colaboratory." Available: https://colab.research.google.com.

[33] Kaggle, "Your machine learning and data science community." Available: https://www.kaggle.com.

[34] Roboflow, "Give your software the power to see objects in images and video." Available: https://roboflow.com.

[35] OpenCV, "Open computer vision library." Available: https://opencv.org.

[36] M. Hossin and M. N. Sulaiman, "A review on evaluation metrics for data classification evaluations," Int. J. Data Min. Knowl. Manag. Process, vol. 5, no. 2, pp. 1–11, March 2015.

[37] K. D. P. Adam and J. Ba, "A method for stochastic optimization," arXiv preprint arXiv:1412.6980, 2014.

[38] A. Sugiharto and R. Kusumaningrum, "Automated detection of driver and passenger without seat belt using YOLOv8," Int. J. Adv. Comput. Sci. Appl., vol. 14, no. 11, pp. 806–813, November 2023.

[39] Roboflow, "What is inference? Roboflow inference." Available: https://inference.roboflow.com/.

[40] A. Chahal, S. R. Addula, A. Jain, P. Gulia, N. S. Gill, and B. V. Dhandayuthapani, "Systematic analysis based on conflux of machine learning and internet of things using bibliometric analysis," J. Intell. Syst. Internet Things, vol. 13, no. 1, pp. 196–224, Jan. 2024, doi: 10.54216/JISIoT.130115.

[41] A. Alshuaibi, M. Almaayah, and A. Ali, "Machine learning for cybersecurity issues: A systematic review," J. Cyber Secur. Risk Audit., vol. 2025, no. 1, pp. 36–46, Jan. 2025.

[42] R. S. Mousa and R. Shehab, "Applying risk analysis for determining threats and countermeasures in workstation domain," J. Cyber Secur. Risk Audit., vol. 2025, no. 1, pp. 12–21, Jan. 2025, doi: 10.63180/jcsra.thestap.2025.1.2.