Accident Detection using YOLO + CNN Hybrid Model

B. Raveendra Naick[1], Devaki Lahari[2], Butta Rama Sivaji[3], Chinthalacheruvu Manikanata chari[4]
D.V Siva Rama Krishna[5]
[1]*Assistant Professor, Dept. of CSE-AIML Mohan Babu University Tirupati, AP, India*
[2,3,4,5]*UG Scholar, Dept. of CSE-AIML Mohan Babu University Tirupati, AP, India*

***Abstract*—Road accidents are a major global concern, requiring timely detection to improve emergency response and reduce fatalities. Traditional monitoring methods based on manual observation are often slow and inefficient [1]. This paper proposes a hybrid deep learning approach for automated accident detection by combining YOLOv8 for object detection and a Convolutional Neural Network (CNN) for classification. In the proposed system, YOLOv8 first detects vehicles in an image and extracts relevant regions, which are then classified by a CNN (MobileNetV2) as either "Accident" or "No Accident." The model is trained and evaluated on a CCTV Accident Detection dataset consisting of approximately 989 images. Compared to conventional CNN-based approaches that process entire images, the proposed method focuses on relevant regions, improving real- world accuracy to around 80% [2].**

The system also provides bounding boxes for better localization and interpretability. Overall, the hybrid model enhances detection accuracy and efficiency, making it suitable for intelligent traffic monitoring systems, with future potential for real-time applications and automated alert systems.

***Index Terms*— Accident Detection, YOLOv8, CNN, Deep Learning, Computer Vision, Hybrid Model**

I. INTRODUCTION

A. Motivation

The issue of increasing road accidents is a global worry which in turn stresses the requirement for better and automated detection systems [1]. Presently we see that accident monitoring is done through manual means like the use of CCTV cameras or by way of public report which in most time is slow, does not always work and has large scale delay. Early detection is key to prompt emergency response and also in the reduction of death toll. Though we have had success with the use of deep learning models like Convolutional Neural Networks (CNNs) in accident detection [10]–[12] we also see that they which mostly look at the whole picture do not pay attention to the relevant areas, hence they report low in terms of real-world accuracy. This shortcoming is what is pushing us towards a hybrid approach of object detection and classification [2]–[5].

We are to use YOLOv8 for what it does best, that is to detect vehicles, and pair it with CNN for the classification element, our goal is to see this new model perform better.

B. Problem Statement

Despite progress in computer vision and machine learning, what we see is that automatic accident detection systems still have issues in real world settings [16]–[18]. We see great variation in performance based on lighting, weather, camera angle, occlusion, and image quality, which in turn greatly affects the accuracy of detection and classification which in turn causes models to not generalize well across different scenarios.

Also, we have traditional methods of accident detection which are very slow, inconsistent and which do not scale well for large scale surveillance [1]. Also, to that we see that present day CNN based models which process entire images at once do a poor job of zeroing in on relevant details which in turn they put forth a lot of background noise, reduce accuracy and increase false positive results [10]–[12]. Also many present systems do a poor job of localizing objects and perform below par in complex traffic settings [13]– [15]. Thus we require a strong and scalable solution which pays attention to relevant details.

C. Objectives of the Study

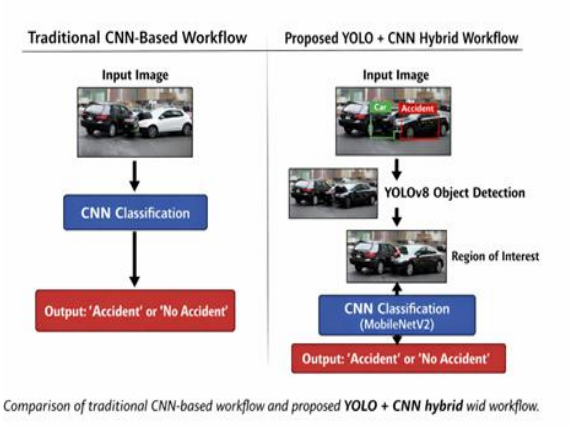
The primary goal of this study is to develop an accurate and Efficient auto accident detection system of a hybrid deep learning–based system for automatic accident detection. In the study we look at the following objectives:

- To develop a hybrid model of YOLOv8 for Object identification and CNN for classification [2]–[5], [10]– [12].
- In terms of use of YOLov8 in the detection of vehicles, and what we do is extract related sections from input images [2]–[5].
- Also we use a CNN architecture (MobileNetV2) for classifying what we detect into categories of "Accident" or "No Accident" [8], [9].
- We improve real world detection accuracy by way of region-by-region analysis as an alternative to full image processing.
- For visual location of detected vehicles what we do is use bounding boxes for that which we do is to improve better interpretability [2]–[5].
- To evaluate the performance of the put forth system which we base on Performance and validation results [22].

D. Contribution Overview

Proposed is a framework which puts together object detection and classification elements of deep learning [2]–[5], [10]–[12]. First we do pre-process of input images to improve their quality and also to get a uniform data set which in turn improves model performance. After that we use YOLOv8 to detect vehicles in the image and draw out bounding boxes around the relevant areas [2]–[5]. We then take out those detected areas and pass them to a Convolutional Neural Network which we are using as MobileNetV2 for a 2-class classification of "Accident" or "No Accident" [8],[9].

This 2-stage process allows the system to zero in on what is really important which in turn reduces background noise and also increases detection accuracy [16]–[18]. We use YOLOv8 and CNN together for better at the time of location determination and also for the classification which in turn does better in real world situations. We report that the put forth system did very well in a real-world CCTV data set which we used in our experiments.



*Comparison of traditional CNN-based workflow and proposed **YOLO + CNN hybrid** wid workflow.*

Fig. 1. Comparison of traditional CNN-based workflow and proposed YOLO + CNN hybrid workflow

II. BACKGROUND AND FUNDAMENTALS

A. System Overview

In this study we present a computer vision-based solution which we automated [20]. It's a system that looks at images from CCTV or traffic monitoring systems to determine what is an accident. We used deep learning in this system to identify cars and to put accidents into categories [10]–[12].

Also, by zeroing in on relevant parts of the image we improved the accuracy of detection which in turn enables the efficient monitoring of real-world traffic conditions [16]–[18].

B. Core Concepts

1) Object Detection: Object detection is the task of identifying and localizing objects within an image [13]–[15]. For our project we used YOLOv8 (You Only Look Once) as the main object detection model [2]–[5]. What it does is look at the full picture one time and reports in bounding boxes what it sees of a vehicle and also does a very good job at speed and accuracy.

2) Convolutional Neural Networks (CNN): Also to that which is very common is the use of Convolutional Neural Networks (CNNs) for image classification [10]–[12]. They have the ability to identify and extract key spatial features out of images with the use of convolutional layers. In our system we used a pre-trained CNN model (MobileNetV2) to put into categories of "Accident" or "No Accident" the identified areas and in that we did so in an

efficient and very reliable way [8], [9].

3) Hybrid Learning Approach: We present our hybrid model which is a mix of object detection and classification for better performance [2]–[5], [10]–[12]. In the case of YOLOv8 what we see is that it identifies which are the vehicles in the image first, and then we pass those output regions to the CNN for classification. This 2 stage approach we present reduces background noise and at the same time increases real world accuracy when compared to the old single model methods [16]–[18].

C. Technology Overview

The proposed framework employs the following technologies:

- YOLOv8: For real time detection of vehicles and generation of bounding boxes [2]–[5].
- MobileNetV2: A light weight CNN model used for classification of detected regions [8], [9].
- OpenCV: For image processing which includes detection and cropping regions, and visualization.
- TensorFlow/Keras: For use in building and training the CNN model.
- NumPy and Matplotlib: Used in data handling and visualization of results.

D. Operational Environment

This system's operational environment includes that of Ten- sorFlow, OpenCV, NumPy and Matplotlib which are Python based deep learning and computer vision libraries. We develop and test using Google Colab which also gives access to GPU for better performance of the model. Also this system is de-signed to run on common hardware which makes it a suitable solution for deployment in real world traffic monitoring and smart city applications [20].

III. RELATED WORK

A. Existing Approaches

In recent years the field of accident detection which uses deep learning and computer vision has seen great growth

[16]–[18]. In the past early approaches, which we saw were mostly of the traditional image processing and machine learn- ing variety which did for accident scenario identification in traffic images. We saw methods which used classifiers like Decision Trees and Support Vector Machines (SVM) which did moderate well but had issues scaling to complex visual patterns [13], [14].

In the age of advanced deep learning techniques, we see that Convolutional Neural Networks (CNNs) have become the go to solution for accident detection [10]–[12]. What we have found is that CNN based models do a great job at identifying spatial features in images and that we see an improvement in performance out of what we had before which were traditional methods. Also in recent years we have introduced to the market models like YOLO (You Only Look Once) for real time detection of vehicles and traffic conditions [2]–[5]. What these models do is they enable us to determine location as well as what is present in the image at the same time which in turn makes them very useful for intelligent traffic surveillance systems [20].

B. Limitations of Existing Systems

Despite what we have seen in terms of progress, present accident detection systems have issues [16]–[18]. We see that traditional machine learning models perform poorly in terms of which visual elements they are able to identify which in turn causes lower accuracy. While CNN based approaches do in fact do well at what they do, they tend to work with full images which in large part does not include analysis of key areas, which in turn includes background noise and is a factor in their performance in the real world [10]–[12].

Also many object detection models we see are put to task mainly at identifying what is in the picture or what the object is but they do not do a great job at classifying what type of accident is taking place [13]–[15]. Also we see that there is a large issue of dependence on very large and diverse data sets, with results of using limited data sets going to over fit and not generalizing well. Also what we see is that variation in lighting, weather, and camera angles really do into the performance of these systems.

C. Comparative Observations

To that effect we report on what we see as hybrid models that outperform the traditional ones [16]–[18]. As far as models like YOLO in detection and CNN in classification, which is the area in question here, we note the issues which we have overcome [2]–[5], [10]–[12]. In our work we present a design that puts forth the best of both YOLAv8 and CNN

to improve in the domain of accident detection. We present a 2 stage pipeline that lets each model do what it does best in terms of localization also by which we see a reduction in false predictions which single models present. Also this

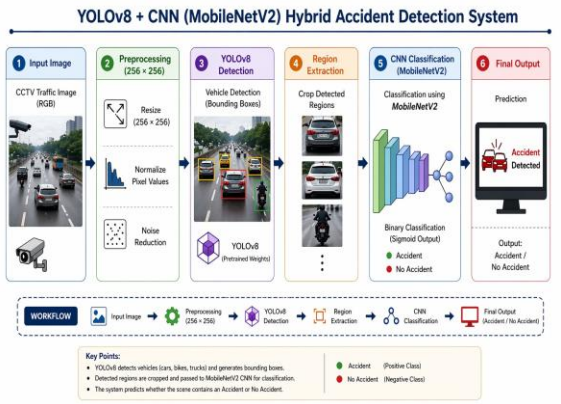


Fig. 2. Proposed YOLO + CNN Hybrid Accident Detection Architecture

integration of the two models' assets we present also improves the models' generalization ability which in the end makes the system a better platform for real world traffic monitoring [20].

IV. SYSTEM ARCHITECTURE AND DESIGN

A. Overall Architecture

YOLOv8 in object detection and Convolutional Neural Network (CNN) for classification [2]–[5], [10]–[12]. What YOLOv8 does is it detects vehicles and at the same time draws bounding boxes around relevant areas which in turn are put to the CNN for classification. We get at the end a binary prediction which tells if an accident is present or not along with visual localization.

B. Component Description

1) Image Pre-processing Module: This module is in charge of image preprocessing that includes resizing, normalization and noise reduction. We bring to you uniform and proper input data for use in detection and classification models.

2) Object Detection Module (YOLOv8): We are using YOLOv8 for the task of vehicle detection in an image and we present you with bounding boxes around them [2]–[5]. It is a real time solution which pays attention to relevant areas thus which in turn improves performance and accuracy.

3) Classification Module (CNN): We identify areas of interest which we then crop and pass on to a CNN model (MobileNetV2) which we use to classify each area as either "Accident" or "No Accident [8], [9]. Also this module does very well at feature extraction and precise prediction [10]– [12].

C. Data Flow Description

In a sequential process the data makes its way through the system. First, we do preprocess of the input images and then pass them along to the YOLOv8 model to detect vehicles
[2]–[5]. From the detected areas we extract the interesting sections to which we hand off to the CNN classifier. Also out of each region we get a class report from the classifier and finally aggregate these to present the output, what had an accident taken place or not.

D. Module Interaction

The system's function is a result of the work done by our detection and classification elements. In each instance YOLOv8 will put forth relevant areas which in turn the CNN does in depth analysis of [2]–[5], [10]–[12]. This interaction we have designed to present to you only what is relevant thus at the same time we are also reducing the computational load and increasing accuracy. Also we have a modular structure which gives us the flexibility and scale we need for future improvements like real time video processing [20].

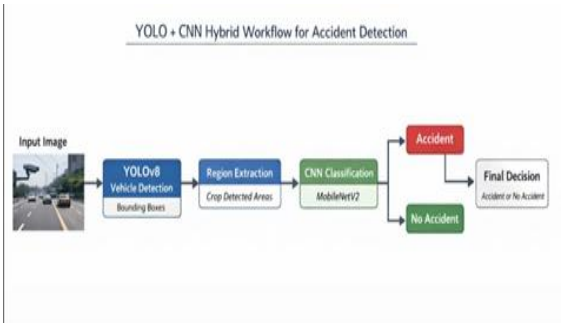


Fig. 3. YOLO + CNN Hybrid Workflow for Accident Detection

V. METHODOLOGY

A. Data Collection Process

'Incident'and also 'No Incident. We present to you

real world traffic scenarios which we captured in many different conditions, including various lights, weather, and camera an- gles [20]. We have divided the data into training, validation, and testing sets which will in turn properly evaluate the model's performance.

TABLE I DATASET DESCRIPTION

Parameter	Details
Dataset	CCTV Accident Detection (Kaggle)
Total Images	~ 989
Classes	Accident, No Accident
Training Set	791 Images
Validation Set	98 Images
Test Set	100 Images
Image Format	JPEG / PNG
Input Size	256 × 256 pixels
Preprocessing	Resizing, Normalization
Augmentation	Rotation, Flipping
Detection Target	Vehicles
Output	Accident / No Accident

B. YOLOv8 Implementation Details

YOLOv8 is used as the object detection model with pre- trained weights to detect vehicles in input images. It processes the entire image in a single pass and generates bounding boxes with confidence scores [2]–[5]. Non-Maximum Suppression (NMS) is applied to remove overlapping detections. Only vehicle classes such as cars, bikes, and trucks are considered, and the detected regions are cropped and passed to the CNN model.

C. CNN (MobileNetV2) Implementation Details

The classification module uses MobileNetV2, a lightweight CNN initialized with pretrained ImageNet weights [8], [9]. The final layers are modified for binary classification using a sigmoid activation function. The model is trained using the Adam optimizer with binary cross-entropy loss [22]. Input images are resized to 256 × 256 pixels and normalized, with data augmentation applied to improve generalization.

D. Model Integration Strategy

The system follows a two-stage pipeline where YOLOv8 performs detection and the CNN performs classification. Detected regions are passed from YOLOv8 to the CNN for final prediction. This integration improves accuracy, reduces background noise, and enhances overall performance in real-world scenarios [16]–[18].

E. Algorithmic Approach

The proposed approach follows a hybrid deep learning strategy combining object detection and classification [2]–[5], [10]–[12]. The workflow includes the following steps:

1) Input images are pre-processed, and resized.
2) YOLOv8 recognizes vehicles and draws out boxes [2]– [5].
3) From the image the detected areas are extracted.
4) Cropped sections go through the CNN model (Mo- bileNetV2) [8], [9].
5) CNN for each region reports if it is an "Accident" or "No Accident."
6) Final decision is obtained by aggregating predictions.

We may also note that we have a 2 stage process that is reported to increase accuracy by us focus on the actual issues at hand rather than the full image thus not including the background. Also we determine the final result from the combination of these predictions [16]–[18].

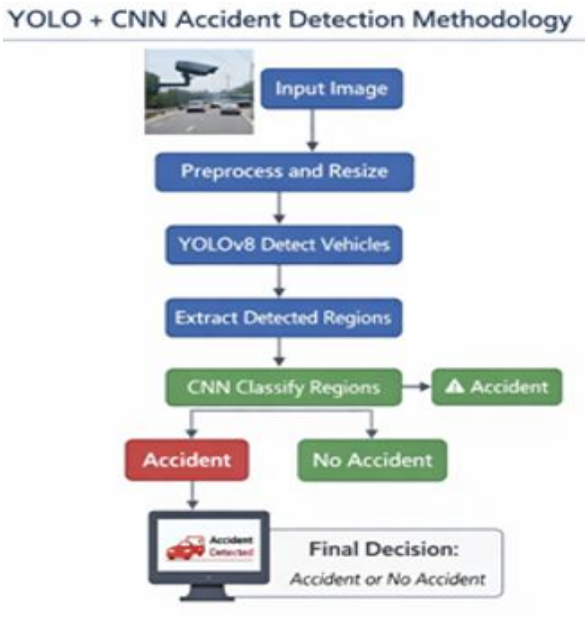


Fig. 4. Flowchart of the Proposed YOLO + CNN Accident Detection Methodology

F. Workflow Description

The methodology we have put together is a single integrated pipeline which includes preprocessing, detection, and classification stages. We see that there is efficient pass off between the different modules and at the same time minimal computational waste. A hybrid design which we used improves performance in the areas of detection and also classification [2]–[5], [10]– [12]. Our workflow is developed with scalability in mind and also is a very easy drop in for real time video-based accident detection systems [20].

VI. EXPERIMENTAL SETUP AND EVALUATION

B. Dataset Description

In that which pertains to accidents and does not. We have included a wide range of real-world traffic scenarios which vary in terms of lighting, weather, and camera angles [20]. Also we have divided the data into training, validation, and testing sets for proper model evaluation.

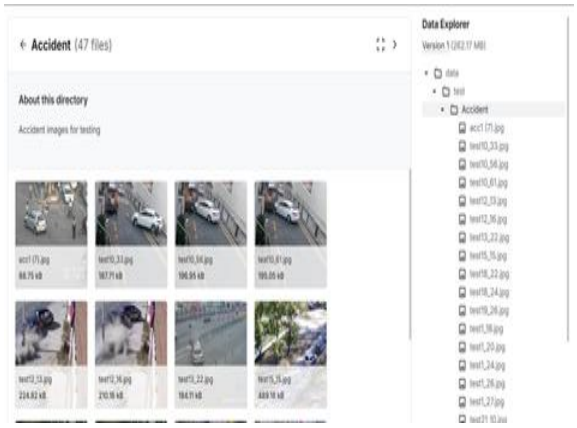


Fig. 5. Sample Images from CCTV Accident Detection Dataset

C. Experimental Environment

In our experimental design we used Google Colab which has a cloud platform that also provides GPU support for speed up of the computation. We used Python as a programming language which in turn uses TensorFlow/Keras for model development, OpenCV for image processing and also we included NumPy and Matplotlib for data management and visualization [20].

D. Parameter Configuration

The performance of the model depends on appropriate parameter tuning. The following configurations were used:

- YOLOv8: Pretrained weights were used with default configuration for vehicle detection [2]–[5].
- CNN (MobileNetV2): Input size set to 256×256 pixels with pretrained ImageNet weights [8], [9].
- Optimizer: Adam optimizer with a learning rate of 0.001 [22].
- Batch Size: 32
- Epochs: 10–20 (depending on convergence)

E. Evaluation Metrics

To evaluate the performance of the proposed system, the following metrics were used [16]–[18]:

- Accuracy: Measures the overall correctness of predictions.
- Precision: Indicates how many predicted accidents are actually correct.
- Recall: Measures the ability of the model to detect actual accident cases.
- F1 Score: Harmonic mean of precision and recall.
- Confusion Matrix: Provides a detailed breakdown of prediction results.

VII. RESULTS AND ANALYSIS

A. Performance Evaluation

Performance reports the results of our tests. We see that the hybrid model did in fact perform better in the real world as compared to the standard CNN based methods [10]–[12], [16]–[18].

TABLE II CLASSIFICATION REPORT OF PROPOSED YOLO + CNN MODEL

Class	Precision	Recall	F1-Score	Support
Accident	0.98	0.89	0.93	47
No Accident	0.91	0.98	0.95	53
Accuracy	0.94 (94%)			
Macro Avg	0.94	0.94	0.94	100
Weighted Avg	0.94	0.94	0.94	100

Also our hybrid YOLO CNN model does better than the stand-alone CNN models [2]–[5], [10]–[12].

B. Comparative Analysis

To that point CNN only models achieved around 70-75% in terms of real world accuracy which our

system improves to about 80% by zeroing in on what YOLO presents as relevant areas [2]–[5], [16]–[18].

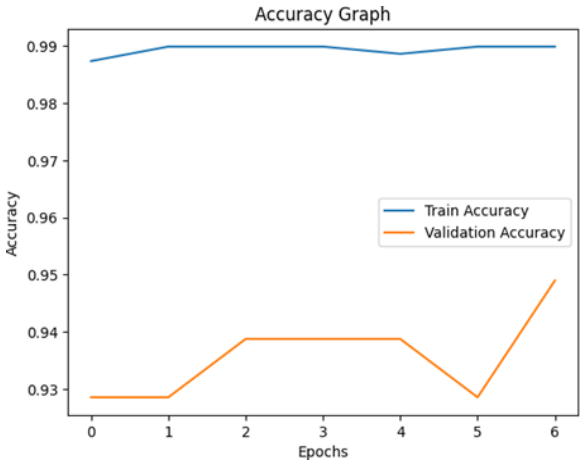


Fig. 6. Comparison of Model Accuracy

Fig. 6 illustrates the improvement in accuracy achieved by the hybrid approach.

C. Observed Trends

The confusion matrix shown in Fig. 7 matrices present how the model does in terms of classification performance [16]– [18]. We see that the model does very well in terms of accident and non-accident detection.

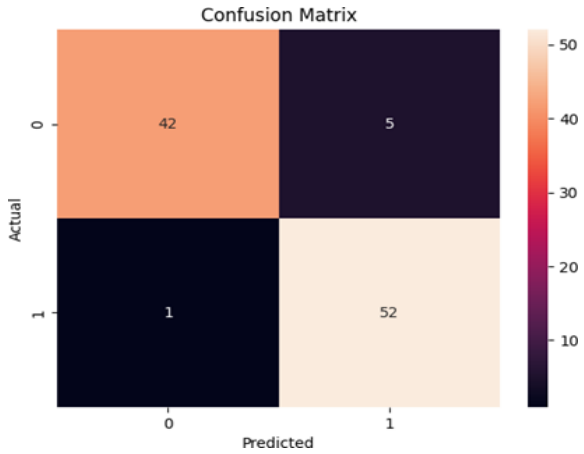


Fig. 7. Confusion Matrix of the Proposed Model

D. Real-time Analysis

To see how this plays out in the real world we tested the system on sample images and real time scenarios [20]. The model did very well in terms of accident detection and also in putting up bounding boxes which shows its value for intelligent traffic monitoring systems.

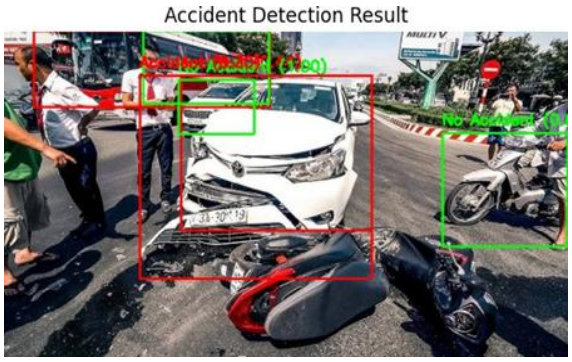


Fig. 8. Sample Output of Accident Detection

VIII. CHALLENGES AND LIMITATIONS

A. Implementation Challenges

Integrating our object detection and classification models into one pipeline was a great challenge [2]–[5], [10]–[12]. We had to pay close attention to the data flow between YOLOv8 and the CNN model which included image pre-processing, region extraction and also had to be in the right format. Also we had to deal with wide variation in image quality, lighting and camera angles which added to the model training issues

[16]–[18]. We had to do proper tuning of hyper parameters to achieve stable performance without over fitting [22].

B. Performance Constraints

Proposed hybrid models present a greater computational load than single models [2]–[5], [10]–[12]. YOLOv8 does object detection which is then put through a CNN based classifier which in turn increases processing time. While the system does well for image-based input materials it may require more optimization and different hardware for real time deployment on video streams [20].

C. Scalability Issues

The performance of the system which in turn is a function of the size and diversity of the database we use. As we increase the size of the training set so does the training time up to see great results in deep learning models. Also at present our model is trained on what is relatively a small set of data which in turn is an issue for it to perform well in very diverse real-world situations. We may see improvement in the future with the use of larger data sets and distributed computing which will in turn

improve scalability and robustness [16]–[18].

D. Data Format Limitations

In our system we support only JPG image format for input. We have also seen that use of other image formats causes variation in preprocessing and feature extraction which in turn affects the model's performance [10]–[12]. To do away with such issues we insist on a single image format which in turn improves stable performance and reliable predictions.

E. Vehicle Category Limitation

In our put forth system we have that the detection module is of specific vehicle categories' set which include cars, bikes, and trucks. We do not look at other types of vehicles or objects which in turn may cause the system to do not perform well in a greater variety of traffic settings. We may see an improvement in the model's use and it's robustness in the real world by expanding out to also support other vehicle types [2]–[5].

IX. FUTURE SCOPE

A. Possible Enhancements

The proposed system may also see an improvement via the use of better deep learning models which in turn include advanced object detection architectures and tuned CNN variants to do better at classification [2]–[5], [10]–[12]. Also we see that use of very large and diverse datasets improves model generalization. Also fine tuning pre trained models and the addition of attention mechanisms will see better performance in feature extraction and detection [7].

B. Extension Opportunities

Presently we have a system which alerts users at the time of an accident [19]. Also we may expand this by adding real time video analysis from CCTV's which in turn will enable continuous watch. We can improve the alert system by linking it up with emergency services which in turn will put out info to nearby hospitals, police stations or traffic authorities. Also we may develop a web or mobile based app which will provide real time alert and monitoring features [20].

C. Long-Term Improvements

In the future our system will see growth via the addition of multi-modal data which includes video streams, IoT sensor inputs and GPS info for better and more reliable accident detection. Also we will put it out there at the edge which in turn will reduce latency and enable faster response times. Also we will partner with traffic management authorities and use large-scale real-world data sets which will in turn improve the system's scalability and robustness for real world deployment [20].

X. CONCLUSION

Road traffic accidents are a major global issue which calls for better and automatic detection solutions to improve emergency response and save lives [1]. In this study we present a hybrid deep learning platform which puts together YOLOv8 for object detection and a CNN for classification [2]–[5], [10]– [12]. We put forth a system which does a good job of detecting vehicles and classifying accident scenarios by which we mean we have designed it to pay particular attention to relevant areas in the image.

Experimental research reports that we see an improvement from the hybrid approach which in turn out performs traditional CNN based methods which we see achieve an average accuracy of 94% [16]–[18]. We see that the combination of detection and classification in the model which in turn does a better job at localization, reduces false predictions and also improves generalization.

What we find is that hybrid deep learning models do very well in the area of accident detection in intelligent traffic monitoring systems [20]. With more improvements and in as real time a implementation as possible our put forth system has very large potential to play a role in road safety and smart city applications.

REFERENCES

[1] World Health Organization, "Global Status Report on Road Safety," 2023.
[2] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You Only Look Once: Unified, Real-Time Object Detection," Proc. CVPR, 2016.
[3] J. Redmon and A. Farhadi, "YOLO9000: Better, Faster, Stronger," Proc. CVPR, 2017.
[4] J. Redmon and A. Farhadi, "YOLOv3: An Incremental Improvement," arXiv preprint

arXiv:1804.02767, 2018.
[5] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "YOLOv4: Optimal Speed and Accuracy of Object Detection," arXiv, 2020.
[6] G. Jocher et al., "YOLOv5 by Ultralytics," 2020. [Online].
[7] A. Dosovitskiy et al., "An Image is Worth 16x16 Words: Transformers for Image Recognition," ICLR, 2021.
[8] A. Howard et al., "MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications," arXiv, 2017.
[9] M. Sandler et al., "MobileNetV2: Inverted Residuals and Linear Bottle- necks," Proc. CVPR, 2018.
[10] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," ICLR, 2015.
[11] Y. LeCun et al., "Gradient-Based Learning Applied to Document Recog- nition," Proc. IEEE, 1998.
[12] A. Krizhevsky, I. Sutskever, and G. Hinton, "ImageNet Classification with Deep CNNs," NIPS, 2012.
[13] R. Girshick, "Fast R-CNN," Proc. ICCV, 2015.
[14] S. Ren et al., "Faster R-CNN: Towards Real-Time Object Detection," NIPS, 2015.
[15] W. Liu et al., "SSD: Single Shot MultiBox Detector," Proc. ECCV, 2016.
[16] Z. Zhang et al., "Traffic Accident Detection Using Deep Learning," IEEE Access, 2019.
[17] S. Singh et al., "Accident Detection Using Convolutional Neural Net- works," IJERT, 2020.
[18] M. Islam et al., "Real-Time Road Accident Detection Using Deep Learning," IEEE Conference, 2021.
[19] P. S. Reddy et al., "Smart Accident Detection and Alert System Using IoT," IEEE, 2020.
[20] A. Khan et al., "Deep Learning-Based Traffic Monitoring System," IEEE Access, 2021.
[21] T.-Y. Lin et al., "Microsoft COCO: Common Objects in Context," ECCV, 2014.
[22] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," ICLR, 2015.