ACCIDENT DETECTION & RESPONSE SYSTEM

A PROJECT REPORT

Submitted by

17CP021: Shlok Bhatt

17CP032: Jainil Panchal

in partial fulfilment for the award of the degree of

B. TECH. (COMPUTER ENGINEERING)

Under the course of

CP442: PROJECT-II

BIRLA VISHVAKARMA MAHAVIDYALAYA
ENGINEERING COLLEGE

(An Autonomous Institution)

VALLABH VIDYANAGAR

Affiliated to

GUJARAT TECHNOLOGICAL UNIVERSITY, AHMEDABAD

Academic Year: 2020 – 2021

B. V. M. ENGINEERING COLLEGE, VALLABH VIDYANAGAR-388120

APPROVAL SHEET

The project work entitled "**ACCIDENT DETECTION & RESPONSE SYSTEM**" carried out by "**Shlok Bhatt (17CP021), Jainil Panchal (17CP032)"** is approved for the submission in the course **CP442, Project-II** for the partial fulfillment for the award of the degree of B. Tech. (Computer Engineering).

Examiner(s) Name & Designation:

Signature:

Date:

Place: Birla Vishvakarma Mahavidyalaya,
Vallabh Vidyanagar, Anand.

CERTIFICATE

This is to certify that Project Work embodied in this project report titled "**ACCIDENT DETECTION & RESPONSE SYSTEM**" was carried out by "**Shlok Bhatt (17CP021), Jainil Panchal (17CP032)**" under the course **CP442, Project-II** for the partial fulfillment for the award of the degree of B. Tech. (Computer Engineering). Followings are the supervisors at the institute.

Date:

Place: Birla Vishvakarma Mahavidyalaya,
Vallabh Vidyanagar, Anand.

(**Prof. Mahasweta J. Joshi**)

Assistant Professor of Computer Engineering, BVM

(**Ms. Dipa Soni**)

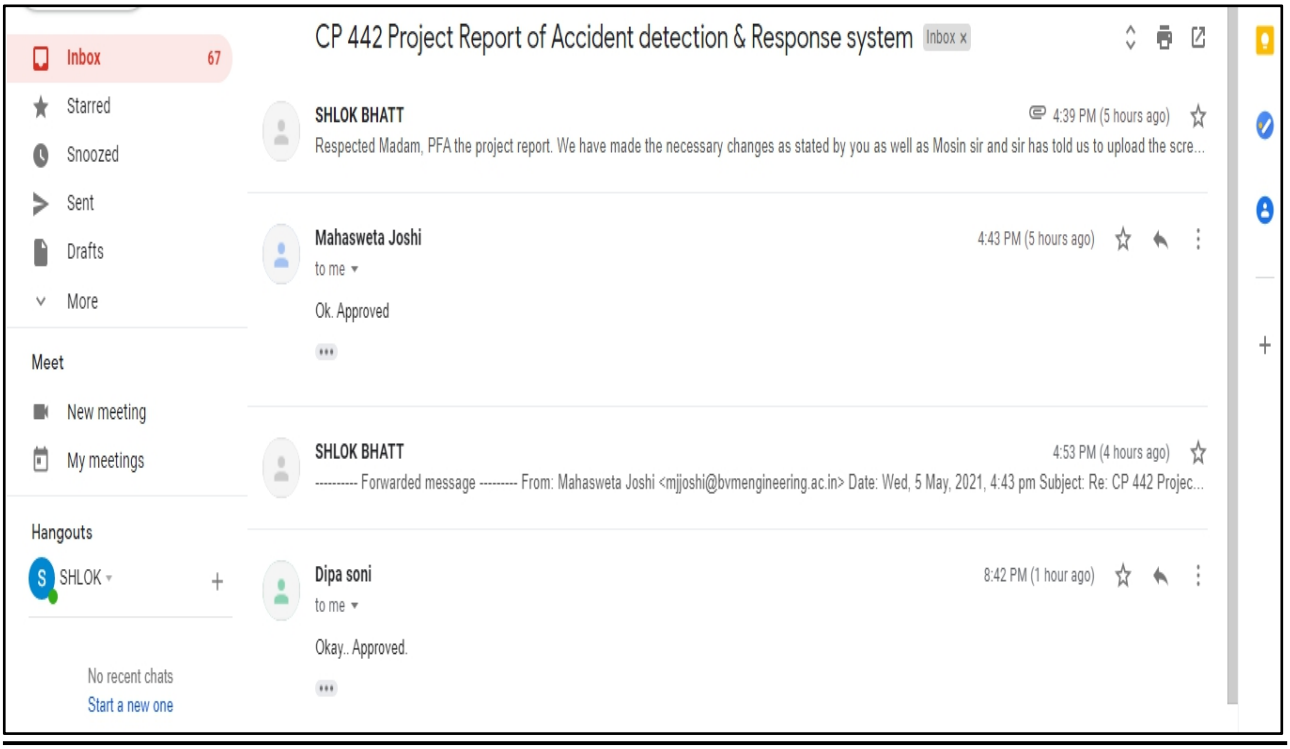
Trainee Assistant Professor of Computer Engineering, BVM

(Dr. Darshak G Thakore)

Prof. & Head

Computer Engineering Department, BVM

APPROVAL FROM GUIDES



DECLARATION OF ORIGINALITY

We hereby certify that we are the sole authors of this report under the course CP442: Project-II and that neither any part of this report nor the whole of the report has been submitted for a degree to any other University or Institution.

We certify that, to the best of our knowledge, the current report does not infringe upon anyone's copyright nor violate any proprietary rights and that any ideas, techniques, quotations or any other material from the work of other people included in our report, published or otherwise, are fully acknowledged in accordance with the standard referencing practices. Furthermore, to the extent that we have included copyrighted material that surpasses the boundary of fair dealing within the meaning of the Indian Copyright (Amendment) Act 2012, we certify that we have obtained a written permission from the copyright owner(s) to include such material(s) in the current report and have included copies of such copyright clearances to our appendix.

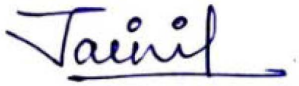
We declare that this is a true copy of the report, including any final revisions, as approved by the report review committee.

We have checked the write up of the present report using an anti-plagiarism database and it is in the allowable limit. Even though later on in case of any complaint pertaining to plagiarism, we are solely responsible for the same and we understand that as per UGC norms, University can even revoke the degree conferred to the student submitting this report.

Date:

Institute code: 007

Shlok Bhatt
(17CP021)



Jainil Panchal
(17CP032)

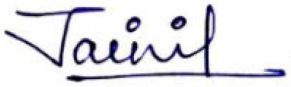
ACKNOWLEDGEMENT

We would like to express our sincere gratitude to **Prof. Mahasweta J. Joshi** and **Ms. Dipa Soni** for their guidance and constant supervision. We would also like to thank our institution for its support towards achieving our goal.

We are indebted to Birla Vishvakarma Mahavidyalaya Engineering College and our project guides for providing necessary information regarding the project and for their support in completing the project. With their feedback, we were able to improve our project in various aspects.

Date:

Shlok Bhatt
(17CP021)



Jainil Panchal
(17CP032)

ABSTRACT

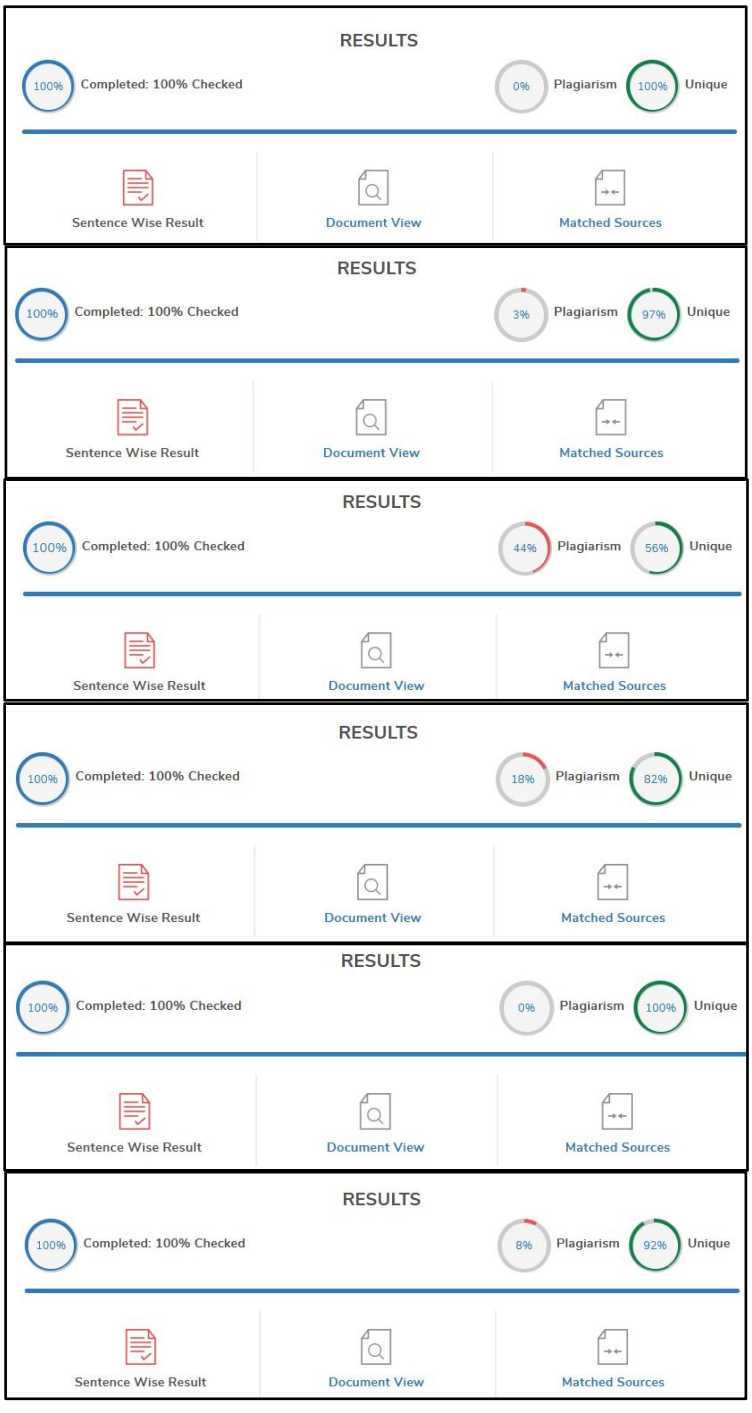
In recent times due to rapid development in the infrastructure of roads, there has been an increase in the number of vehicles on the road which in turn also increases the risk of road accidents. There have been many incidents in which the victims have lost their lives or suffered tragic injuries due to delays in response by the emergency services. With the breakthroughs in the area of Object detection made possible by the advancements in the field of Neural Networks, it is now possible to design a system with a decent amount of accuracy. Thus, our project seeks to address the aforementioned problem by using a reliable, efficient, and fast object detection algorithm to design a robust accident detection and response system.

Our project is composed of two modules namely, the accident detection module and the response system module. For the first module i.e., the accident detection module we have collected datasets from various sources which comprise both the still images as well as video clips of accidents from different camera angles. Also, it consists of accidents from various countries and types which will help the system to detect accidents in different kinds of situations and circumstances. For the task of accident detection, we first tested and evaluated three different object detection algorithms namely, YOLOv3, SSD MobileNet, and Faster RCNN. After our analysis, we finalized the YOLOv3 algorithm for our accident detection task. we have used the YOLOv3 algorithm.

YOLOv3 is an object detection algorithm that uses DarkNet-53 CNN for the detection of objects. As compared to many other contemporary objection detection algorithms it is faster and has fairly decent accuracy. Hence, due to this reason, we have used it in our project as it is a good choice for accident detection in real-time without much loss of accuracy. Our second module namely, response system modules generate an alert when an accident is detected and sends it to the nearby authorities. We plan to design a simple Email-based alert system for this purpose but other complex and more efficient response systems can be designed by using IoT (Internet of Things) or other technologies. Thus, our project can help the victims of the accidents get proper treatment in time and will be beneficial to the authorities as well as the people.

PLAGIARISM REPORT

Average Plagiarism: 12.167%



The tool used: https://smallseotools.com/plagiarism-checker

TABLE OF CONTENTS

List of Figures

List of Tables

List of Symbols, Abbreviations & Nomenclature

Abbreviation	Full-form
WHO	World Health Organization
CCTV	Closed-circuit television
CPU	Central Processing Unit
GPU	Graphical Processing Unit
TPU	Tensor Processing Unit
YOLO	You Look Only Once
SSD	Convolution Neural Network
R-CNN	Region-based Convolutional Neural Networks
FPN	Feature Pyramid Network
SMS	Short Message Service
PIL	Python Imaging Library
OpenCV	Open Computer Vision Library
IOU	Intersection Over Union
FPS	Frames per Second
CNN	Convolutional Neural Network
IoT	Internet of Things
HTML	HyperText Markup Language
CSS	Cascading Style Sheets
JS	JavaScript
PHP	PHP Hypertext Preprocessor
XML	Extensible Markup Language
JSON	JavaScript Object Notation
XAMPP	Cross-platform Apache MariaDB (MySQL), PHP, and Perl
SMTP	Simple Mail Transfer Protocol

ESMTP	Extended SMTP
RPN	:Region Proposal Networks
mAP	Mean Average Precision
mAR	Mean Average Recall
TP	True Positive
TN	True Negative
FP	False Positive
FN	False Negative
UI	User Interface

CHAPTER I - INTRODUCTION

1.1 Introduction (Relevance to practical field and importance of study proposed):

- According to the W.H.O., there are about 1.35 million people and 20-50 million injuries as a result of car accidents worldwide each year.[1]

- Many steps have been taken by car manufacturers to reduce road crash damage, especially with a focus on effective and efficient safety systems. These programs have been able to reduce the number of causes of road accidents.

- But accidents can still occur, and a quick response from paramedics can greatly reduce the number of injured and dead passengers and the impact and magnitude of such accidents.

- Prompt response to road accidents is very important in vehicle management. On the one hand, we should provide health care to people who have been injured in an accident and prevent their health from getting worse, on the other hand, serious accidents often create long-term congestion, if an emergency response is delayed. To minimize those side effects, road accidents need to be identified quickly.

- Generally, the footage of a CCTV is monitored by humans at Traffic / Police facilities. The key point is that CCTV can directly capture crashes within a certain range. These days, more and more CCTV cameras are being installed in urban areas and on highways.

- Although monitoring by humans on CCTV may be reliable, it requires a lot of time and is a tedious task. Therefore, the need arises to create such a system to detect automatic collisions from CCTV footage.

- In recent years, computer technology has improved rapidly and is widely used in the development in the area of transportation (i.e., Autonomous Driving Cars), all thanks to the growing power of computers as well as Deep learning. Vision-based object detection, based on deep learning methods, has been greatly improved.

1.2 Aim:

- The aim of our project is to detect accidents with decent accuracy in real-time and generate an appropriate response to alert the nearby authorities like paramedics, traffic police, police, etc. so that the victims of the accident can get treatment in a timely manner and the risk of mortality/serious injuries which in turn will also help in resolving the congestion of traffic quickly. Thus, it will be beneficial to the authorities as well as the citizens.

1.3 Motivation:

- In our everyday life, we see a lot of accidents happening all around us and various persons falling victim to death or critical injuries due to the delayed response by the authorities like late arrival of ambulance, police, etc. This problem becomes graver especially in our country India due to inadequate infrastructure facilities, scarcity of resources, etc. Thus, we got the motivation to build an accident detection system and response system which effectively and optimally utilizes the available resources like the footage of CCTV cameras which are installed on the roads to detect the accidents in real-time and alert the nearby authorities quickly so that the victims of the accident can get the help and necessary medical attention at a proper time which can help to minimize the casualties in the accidents.

1.4 Scope:

- Our system can be used by the traffic police for accident detection in areas where the density of traffic and the frequency of accidents is high. Also, it can be used by government bodies like the ministry of transportation to find out the causes of accidents and design solutions to reduce the number of accidents. It can also be used by researchers to develop more efficient accident detection systems.

CHAPTER II- LITERATURE SURVEY & OBJECTIVES

2.1 Literature Survey:

- Till now, significant amounts of research and work is done in the area of accident detection and response generation. Various kinds of deep learning models like CNN, R-CNN, Faster R-CNN, SSD MobileNet, etc. as well as other types of algorithms that are used in the field of computer vision have been for the implementation of accident detection and response systems.

- As the field of computer vision, neural networks witnessed astonishing developments over the past few years along with the development of powerful GPUs, many of the researchers across the world have made great progress in accident detection using one-stage detectors - YOLO, Retina net, S.S.D. (Single Stage Detectors), etc. and two-stage detectors - R-CNN, Fast R-CNN, Faster R-CNN, FPN (Feature Pyramid Network), etc.

- Also, there have been some good improvements in the response generation when an accident occurs like user-focused like an SMS to the victim's kin, chip of accident detection in the car, etc.

- But various limitations have been encountered which hinder the goal of achieving maximum accuracy in real-time.
    - They are as follows:
        - Low availability of the relevant dataset
        - High computational costs
        - Low accuracy in low light (low visibility conditions)
        - Most literature focusing on motor vehicle crashes and not on non-motor vehicle crashes like bicycle-related and pedestrian-related
        - Low accuracy in congested traffic conditions
        - Low availability of Indian accident dataset
        - Limited response systems which directly alerts the relevant authorities

- Thus, overall, there have been fruitful and decent improvements in the area of accident detection and response systems but there is still a long road for achieving the main goal of a robust and accurate system.

2.2 Objectives:

- To develop an efficient system for accident detection in accident-prone areas
- To generate a simple and efficient response system that can be useful for alerting the relevant authorities
- To utilize the limited available resources (like CCTV cameras) and get the best use out of them for the benefit of authorities and people
- To help in reducing the mortality rate occurring due to delay in treatment of the victim
- To help in resolving the traffic jams occurring due to accidents in a short duration of time by alerting the nearby authorities quickly
- To help the authorities in analyzing the accidents and help them to address the root cause of them

CHAPTER III - METHODOLOGY & EXPERIMENT

3.1 Proposed System:

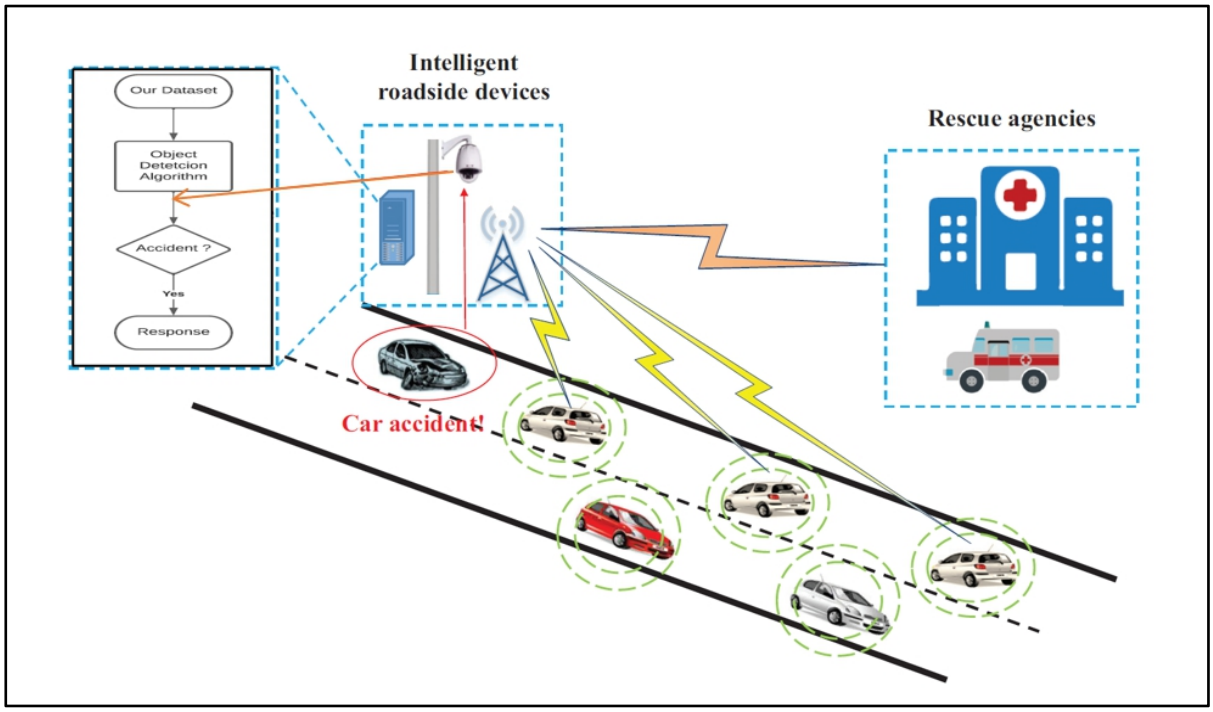


Fig. 3.1 Proposed System [2]

- Our proposed system (as shown in Fig. 3.1) will detect accidents in real-time using CCTV footage which in turn will generate an alert that will be sent to the nearby authorities so that they can take appropriate and required action quickly and efficiently. Thus, our system will help the victims of the accident to get the medical attention in a timely manner and will also help in resolving the traffic jam quickly.

3.1.1 Use Case:

- The below figure Fig. 3.2 shows the use case diagram of our system. Here there are two entities namely, Vehicles and Rescue agencies. It shows how the two entities function in the system and what they can do.

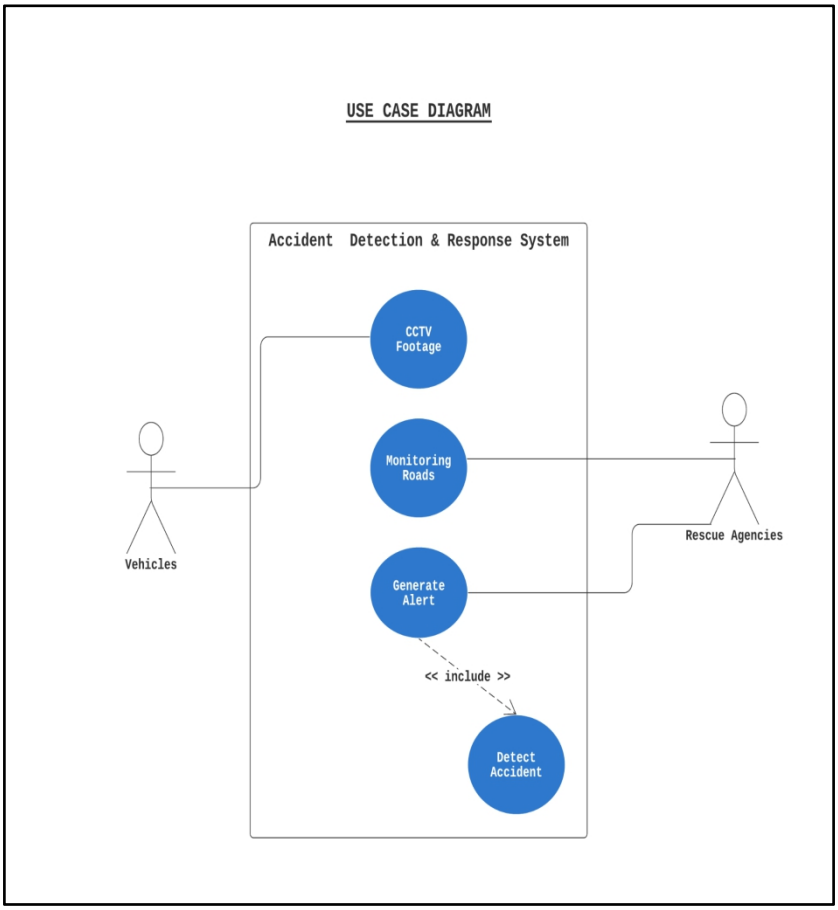


Fig. 3.2 Use Case Diagram

- The CCTV constantly monitors the vehicles and sends its footage back to the server where our system is running. The rescue agencies monitor the roads through CCTV and if an accident is detected then an alert is generated and it is sent to all the nearby rescue agencies. On receiving the alert, the rescue agencies arrive at the accident site in a timely manner to help the victims.

3.1.2 Flowchart of the system:

- The below figure Fig. 3.3 shows the flow chart of our system. When the system is turned ON the CCTV footage is constantly streamed online to the server where our system is running. It samples the video footage in frames and constantly checks in each frame whether an accident is there or not.

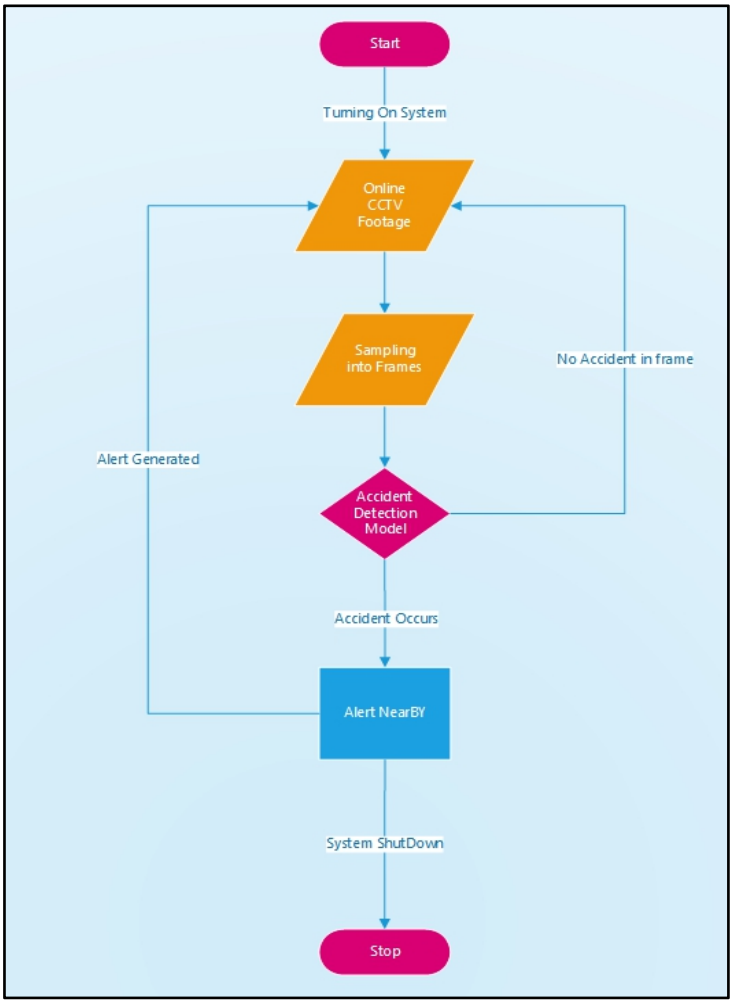


Fig. 3.3 Flowchart

- If an accident is detected it generates an alert to the nearby rescue agencies and then it once again goes back to monitoring the footage and sampling it to check whether an accident has occurred or not.

- If no accident is detected then it goes back to monitoring the footage and sampling it to check if an accident has occurred or not.

- When the system is shut down all the process of accident detection and alert generation is terminated.

3.2 Project Modules:

- **Accident Detection:** This module detects the accident with the help of footage from CCTVs that are installed on the roads. It basically shows the bounding box around the area where the accident is happening in real-time.

- **Response System:** This module will generate an appropriate alert/response and send it to the nearby authorities whenever an accident is detected. At a basic level, it will send an Email alert to the concerned authorities.

3.3 Tools & Technologies:

3.3.1 Languages:

- **Python 3.8:** This is an open-source programming language with certain powerful libraries for developing neural networks & performing image processing.
- **HTML:** The HTML (Hypertext Markup Language) is the standard markup language for documents designed to be displayed in a web browser. It can be assisted by technologies such as CSS and scripting languages such as JavaScript.
- **CSS:** CSS (Cascading Style Sheets) is a style sheet language used for describing the presentation of a document written in a markup language such as HTML. CSS is a cornerstone technology of the World Wide Web, alongside HTML and JavaScript
- **JavaScript:** JavaScript (JS) is the Programming Language for the Web.JS can update and change both HTML and CSS.JS can calculate, manipulate and validate data. It has curly-bracket syntax, dynamic typing, prototype-based object-orientation, and first-class functions
- **PHP 7:** PHP (PHP: Hypertext Preprocessor) is a server scripting language and a powerful tool for making dynamic and interactive Web pages. PHP is a widely-used, free, and efficient alternative to competitors such as Microsoft's ASP.

3.3.2 Tools / Platforms:

- **Google Colaboratory:** Google Colaboratory is a free online cloud-based Jupyter notebook environment that allows us to train our machine learning and deep learning models on CPUs, GPUs, and TPUs.
- **Jupyter Notebook:** The Jupyter Notebook is an open-source web application that allows you to create and share documents that contain live code, equations, visualizations, and narrative text.
- **Notepad++:** Notepad++ is a free-to-use source code editor compatible for various languages.
- **IBM Cloud Annotation:** IBM Cloud Annotations is a fast, easy, and collaborative open-source image annotation tool provided by IBM which provides utilities to export the dataset to various well-known formats like JSON, YOLO, etc.

- **LabellMg**: LabellMg is a graphical image annotation tool. It is written in Python and uses Qt for its graphical interface. Annotations are saved as XML files in PASCAL VOC format, the format used by ImageNet. Besides, it also supports YOLO and CreateML formats.
- **XAMPP:** XAMPP is the most popular PHP development environment. XAMPP is a completely free, easy to install Apache distribution containing MariaDB, PHP, and Perl. The XAMPP open-source package has been set up to be incredibly easy to install and to use.
- **RoboFlow:** RoboFlow aims to democratize computer vision by streamlining the computer vision process from end to end**.** RoboFlow allows you to upload, annotate, organize, train, and deploy with ease.

3.3.3 Libraries / Frameworks:

- **NumPy:** NumPy is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays.
- **TensorFlow 2.0:** TensorFlow is a free and open-source software library for machine learning. It can be used across a range of tasks but has a particular focus on training and inference of deep neural networks. TensorFlow is a symbolic math library based on dataflow and differentiable programming.
- **Keras:** Keras is an open-source library that provides a Python interface for artificial neural networks.
- **OpenCV:** OpenCV is a library of programming functions mainly aimed at real-time computer vision.
- **Seaborn:** Seaborn is a Python data visualization library based on matplotlib. It provides a high-level interface for drawing attractive and informative statistical graphics.
- **Pillow**: Python Imaging Library (PIL) is a free and open-source additional library for the Python programming language that adds support for opening, manipulating, and saving many different image file formats. It is available for Windows, Mac OS X, and Linux.
- **Darknet:** Darknet is an open-source neural network framework written in C and CUDA. It is fast, easy to install, and supports CPU and GPU computation.
- **Flask:** Flask is a popular Python web framework, meaning it is a third-party Python library used for developing & deploying a web application.
- **Smtplib:** The smtplib module defines an SMTP client session object that can be used to send mail to any Internet machine with an SMTP or ESMTP listener daemon.
- **Bootstrap:** Bootstrap is a free and open-source CSS framework directed at responsive, mobile-first front-end web development. It contains CSS and JavaScript-based design templates for typography, forms, buttons, navigation, and other interface components.

3.3.4 Detection Algorithms:

- **YOLO:** It is a real-time object detection algorithm, which is one of the most effective object detection algorithms.
- **SSD MobileNet V2:** The ssd_mobilenet_v2_coco model is a Single-Shot multi-box Detection (SSD) network intended to perform object detection.
- **Faster RCNN:** Faster R-CNN model object detection algorithm that is composed of a feature extraction network which is typically a pre-trained CNN, similar to what we had used for its predecessor.

3.4 Data Collection:

- Shah, Ankit and Lamare, Jean Baptiste and Anh, Tuan Nguyen and Hauptmann, Alexander CADP:" A Novel Dataset for CCTV traffic Camera-based Accident Analysis " https://arxiv.org/pdf/1809.05782v2.pdf

- IITH_Accident Dataset Request (300 MB),
  Hyderabad City Video Dataset for Accident Detection from Hyderabad City CCTV Network
  https://www.iith.ac.in/vigil/resources.html

- Anticipating Accidents in Dashcam
  https://visionlab.uncc.edu/download/summary/60-data/477-ucf-anomaly-detection-dataset

3.5 System Flow Architecture:

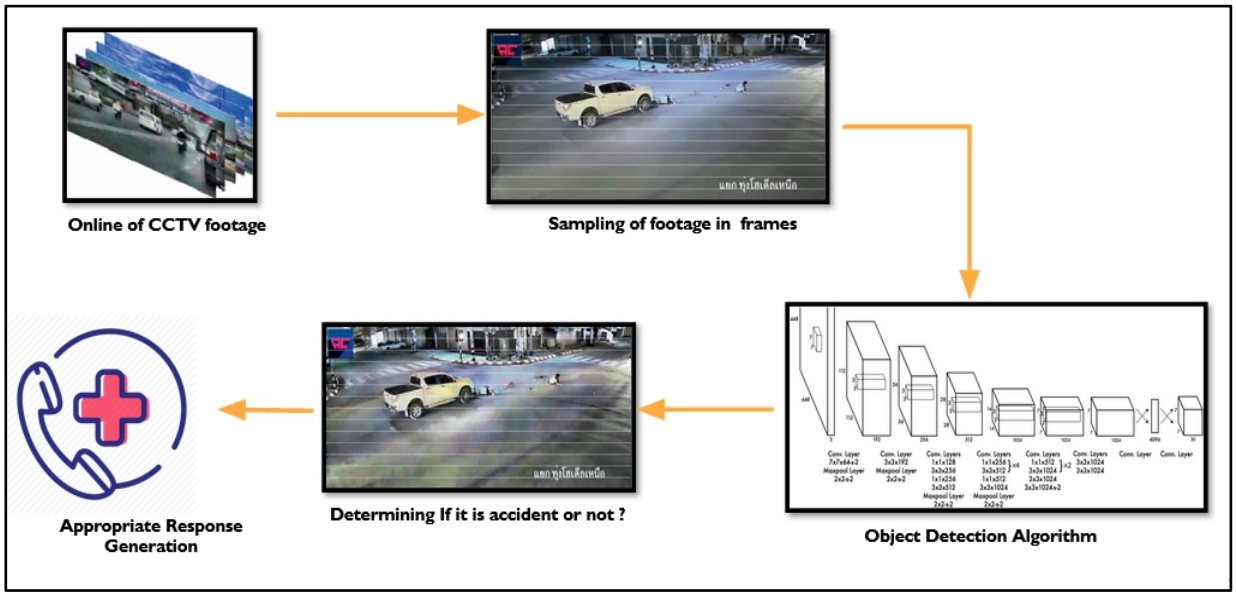


Fig. 3.4 System-Flow Architecture

Fig. 3.4 shows the system flow architecture in which CCTV footage will be given to the system that will get sampled into frames & our custom trained accident detection model will run on that. If the frame contains an accident then the system will generate an appropriate response to alert the nearby rescue agencies.

3.6 Walk-through of Portal:

1. Fig. 3.5 shows the homepage of our web portal through which we can login(which is running on *"localhost/crash").*

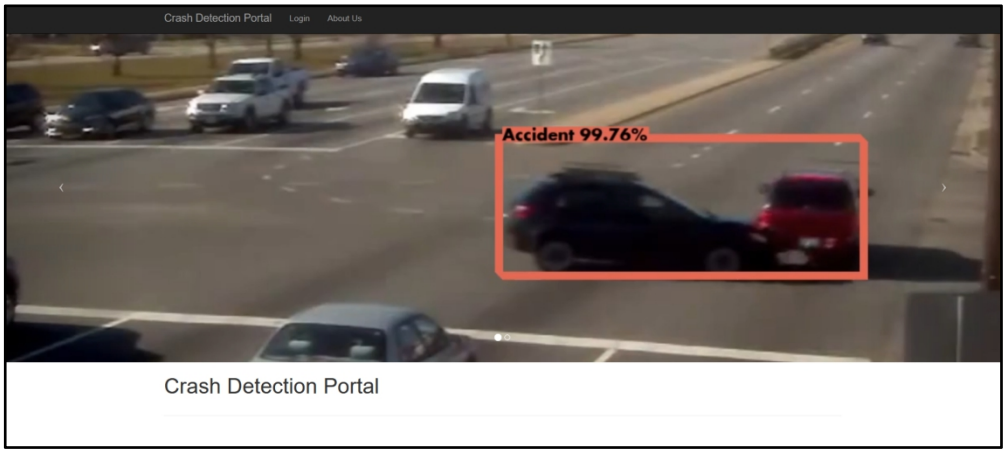


Fig. 3.5 Portal Homepage

2. Fig. 3.6 shows the "Login Page" through which the administrator can login into the portal.

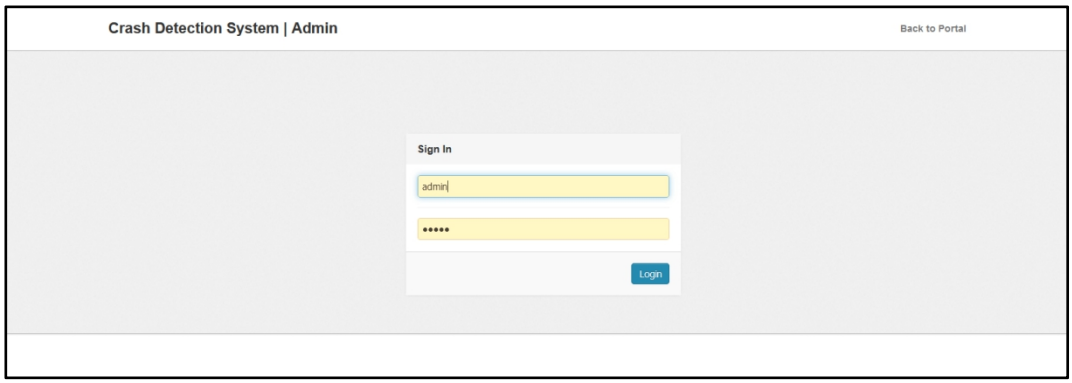


Fig. 3.6 Login Page

3. Fig. 3.7 shows the "Admin - Dashboard". Here, various tabs are there which provides various functionalities regarding our system.

- With change password & forgot password functionality
- On the sidebar, we have multiple options for navigation
- Check It out!!

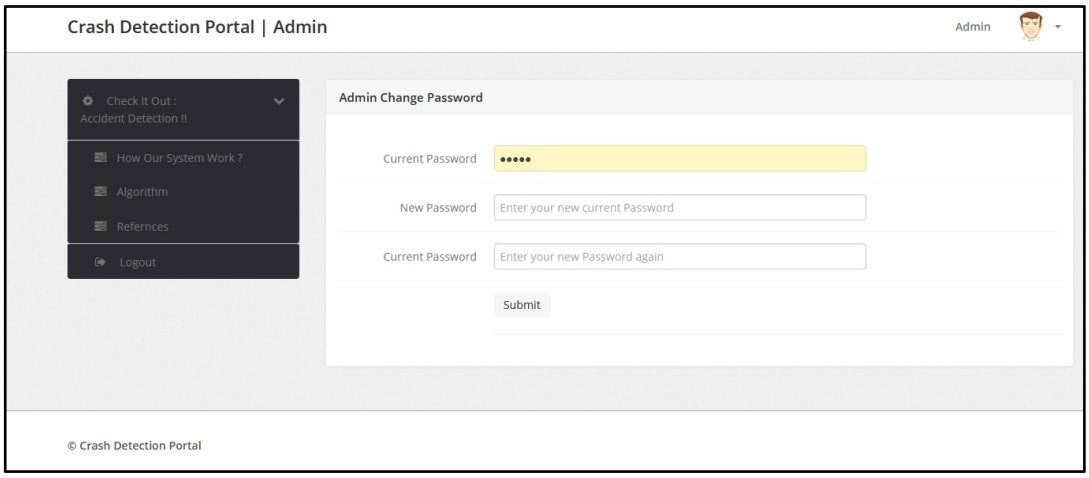


Fig. 3.7 Admin Panel Dashboard

- Fig. 3.8 shows the page of Accident Detection & Response on images. It contains the "Browse.." button to upload the image file. Also, it contains "Send" button to run the accident detection model on it as well as "Back To Portal" button for going back to the portal.

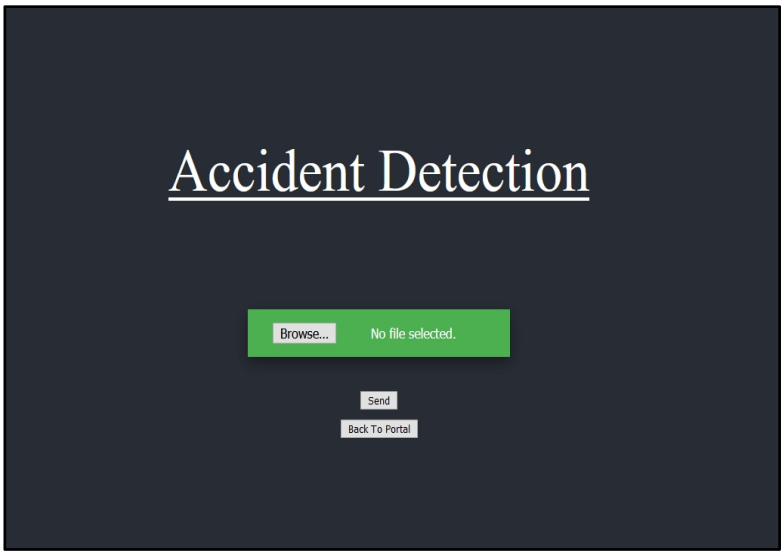


Fig. 3.8 Flask Server page for Accident Detection Image

- Fig. 3.9 shows the page of Accident Detection & Response on video / live feed. It contains numerous buttons like "On/Off" to run the accident detection model on the video as well as "Back To Portal" button for going back to the portal.

Fig. 3.9 Flask Server page for Accident Detection on Video / Live stream

- Fig. 3.10 shows "How does our System work?" page which is one of many tabs available in the Admin Dashboard like Algorithm, References, Logout, etc.

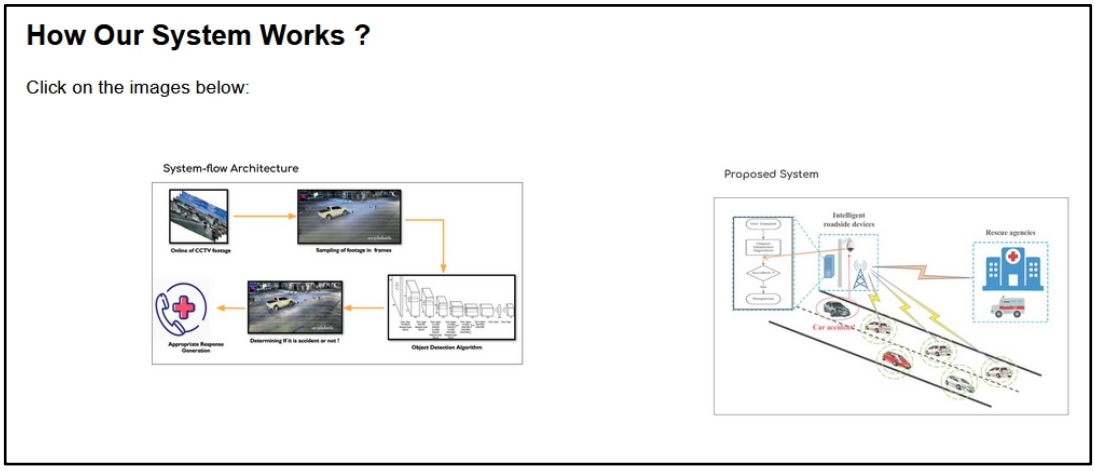


Fig. 3.10 Project Details Web Page

3.7 Algorithms deep dive

3.7.1 Faster RCNN:

- A Faster RCNN detector was implemented shortly after the introduction of Fast RCNN by S.

Ren et al. To overcome the drawbacks of Fast RCNN, a network referred to as Region Proposal Network (RPN) was introduced in Faster RCNN. Fast RCNN performs both region proposal generation and detection tasks. [3]

- It comprises of two modules:
    1. The first module is a deep fully convolutional network that proposes regions, and
    2. The second module is the Fast R-CNN detector that uses the proposed regions.[4]

- The entire system is a single, unified network for object detection.[4]

- A Region Proposal Network (RPN) takes an image (of any size) as input and outputs a set of rectangular object proposals, each with an objectness score.[4]

- Faster RCNN is a true end-to-end deep learning object detector.

- It relies on a Region Proposal Network (RPN) that is
    1. fully convolutional and
    2. can predict the object bounding boxes and "objectness" scores (i.e., a score quantifying how likely it is that a region of an image may contain an image).[5]

- The outputs of the RPNs are then passed into the R-CNN component for final classification and labeling.[5]

- Initially, first ROI pooling is performed, and then the pooled area is fed to CNN and two FC layers for SoftMax classification and the bounding box regressor. It is the first near-real-time object detector tested on the MS-COCO dataset; it achieved mAP = 42.7%, VOC-2012, mAP = 70.4%, and 17 fps with ZFNet. Despite Faster RCNN being much faster than Fast RCNN, there is computational redundancy at the final stage. [3]

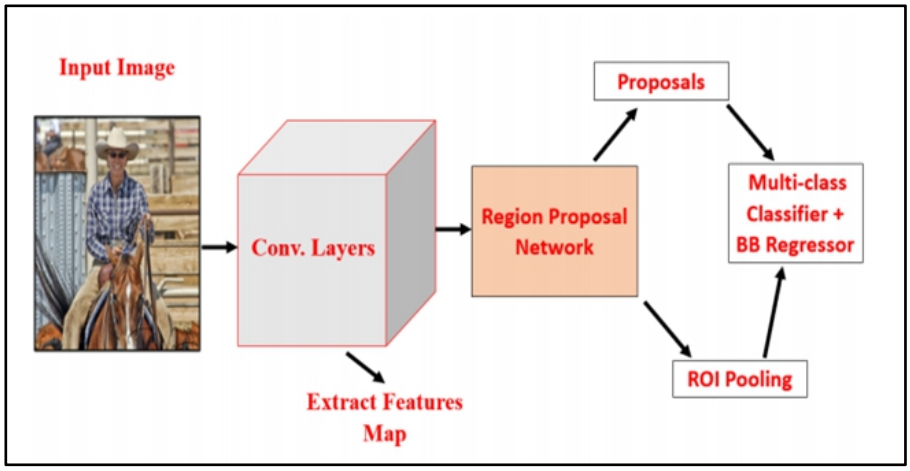


Fig. 3.11 Faster RCNN Block Diagram [3]

- Fig. 3.11 shows the block diagram showing the various parts of the Faster RCNN architecture.

- Though Faster RCNN are very accurate, they are slow as compared to other object detection algorithms, which is evident by the low FPS obtained on a GPU.

3.7.2 SSD MobileNet:

- SSD Mobilenet V2 Object detection model with FPN-lite feature extractor shared box predictor and focal loss, trained on COCO 2017 dataset with training images scaled to 640x640.

- SSD is designed purely for real-time object detection in a deep learning era. Instead of taking two shots as in the RCNN series, one for generating region proposals and another for detecting the object of each proposal, it uses only a single shot to detect multiple objects within an image.[3]

- To improve the detection accuracy of SSD, particularly in detecting small objects, it introduced "multi-reference and multi-resolution detection" techniques. To improve Fast RCNN's real-time speed detection accuracy, SSD eliminated the region proposal network (RPN).[3]

- SSD300 achieves, on the VOC-2012 dataset, mAP = 74.3% at 59 FPS, while SSD500 achieves, on the VOC-2007 dataset, mAP = 76.9% at 22 FPS, which outperforms Faster RCNN (mAP = 73.2% at 7 FPS) and YOLOv1 (mAP = 63.4% at 45 FPS).[3]

- The drawbacks of SSD: at the cost of speed, accuracy increases with the number of default boundary boxes. SSD detectors have more classification errors when compared to RCNN but low localization errors while dealing with similar categories. [3]

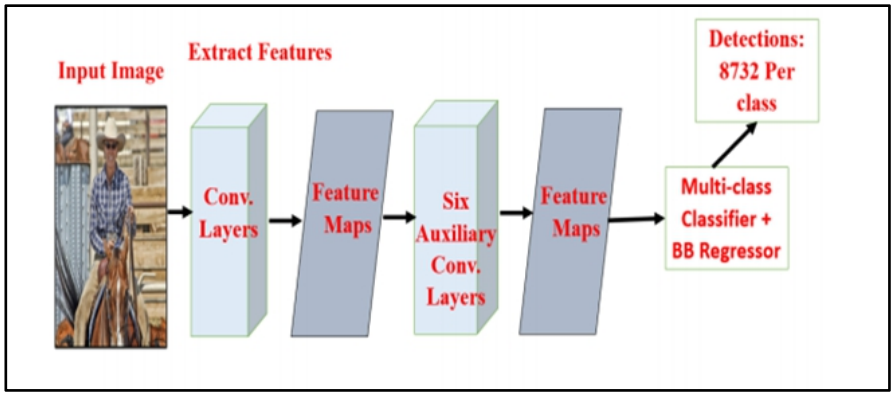


Fig. 3.12 SSD Block Diagram [3]

- Fig. 3.12 shows the block diagram of SSD. It shows various parts of SSD architecture.

- In general, single-stage detectors tend to be less accurate than two-stage detectors but are significantly faster. [5]

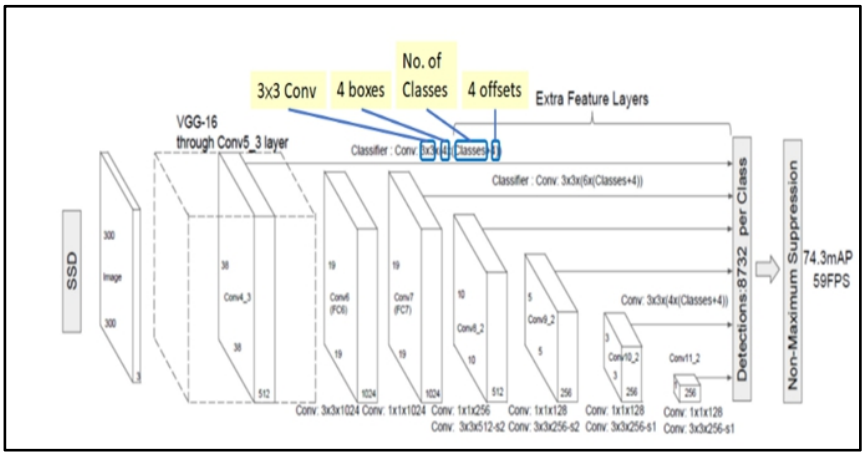


Fig. 3.13 SSD Network Architecture [6]

- Fig. 3.13 shows the SSD Network architecture. It consists of many layers.

3.7.3 YOLO V3:

- YOLO (You Only Look Once) is an object detection algorithm that is capable of detecting and classifying multiple objects from a single frame in real-time.

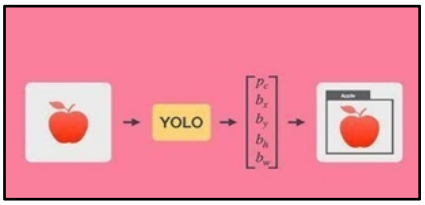


Fig. 3.14 YOLO Regression [7]

- Fig. 3.14 shows YOLO is a single regression problem. Unlike other detectors which use classification, YOLO predicts a vector that contains the bounding box coordinates as well as class probabilities boxes for the whole image in one run of the Algorithm.

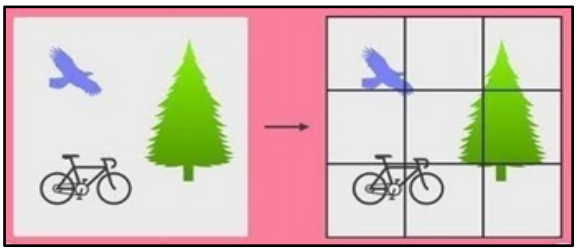


Fig. 3.15 YOLO Anchor boxes [7]

- As shown in Fig. 3.15 YOLO divides the input image into an SXS grid of anchor boxes. Each anchor box will perform a prediction of the class and location of the objects simultaneously with all the other anchor boxes.

- Each grid cell/anchor box predicts B bounding boxes as shown in figure Fig. 3.16 and confidence score for those boxes. If there are no objects the network returns a 0 for the confidence value.

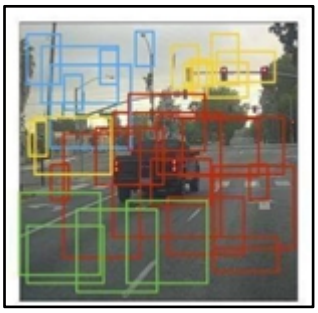


Fig. 3.16 YOLO Prediction [8]

- Each bounding box can be described using four descriptors:
    1. Center of the box (b_x, b_y)
    2. Width (b_w)
    3. Height (b_h)
    4. Value c corresponding to the class of an object

- During the one pass of forwarding propagation, YOLO determines the probability that the cell contains a certain class. The equation for the same is:
  The Probability that there is an object of certain class 'c', Score $_{c,i} = p_c \times c_I$ …. (1)

- A problem arises as there are too many bounding boxes in each grid cell, most are redundant/inaccurate hence we get rid of bounding boxes with low probability predictions.

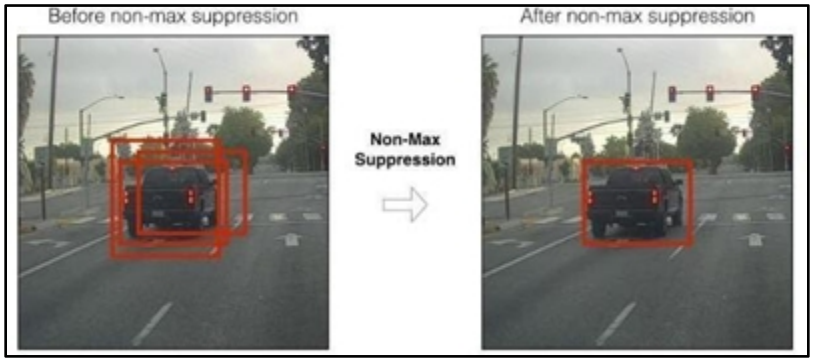


Fig. 3.17 YOLO Non-Max Suppression [8]

- To resolve this problem Non-max suppression which is shown in Fig. 3.17 eliminates the bounding boxes that are very close by performing the IoU (as shown in Fig. 3.17) with the one having the highest-class probability among them.

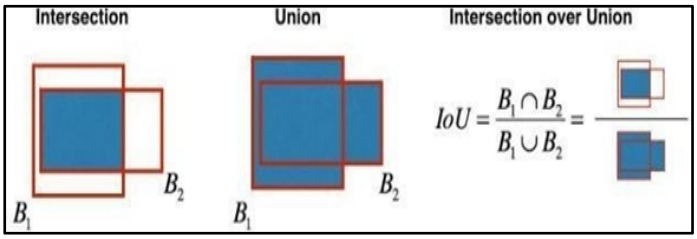


Fig. 3.18 YOLO IoU [8]

- It calculates the value of IoU as shown in Fig. 3.18 for all the bounding boxes respective to the one having the highest-class probability, it then rejects the bounding boxes whose value of IoU is greater than a threshold. It signifies that those two bounding boxes are covering the same object but the other one has a low probability for the same, thus it is eliminated.

- Once done, the algorithm finds the bounding box with the next highest-class probabilities and does the same process, it is done until we are left with all the different bounding boxes

3.8 Experiments, Analytical Computations, and tools used:

- For YOLO, we have used the Darknet framework to train our model. Darknet-53 is a convolutional neural network as shown in Fig. 3.19 that acts as a backbone for the YOLOv3 object detection approach. The improvements upon its predecessor Darknet-19 include the use of residual connections, as well as more layers.

	Type	Filters	Size	Output
	Convolutional	32	3 × 3	256 × 256
	Convolutional	64	3 × 3 / 2	128 × 128
1×	Convolutional	32	1 × 1	
	Convolutional	64	3 × 3	
	Residual			128 × 128
	Convolutional	128	3 × 3 / 2	64 × 64
2×	Convolutional	64	1 × 1	
	Convolutional	128	3 × 3	
	Residual			64 × 64
	Convolutional	256	3 × 3 / 2	32 × 32
8×	Convolutional	128	1 × 1	
	Convolutional	256	3 × 3	
	Residual			32 × 32
	Convolutional	512	3 × 3 / 2	16 × 16
8×	Convolutional	256	1 × 1	
	Convolutional	512	3 × 3	
	Residual			16 × 16
	Convolutional	1024	3 × 3 / 2	8 × 8
4×	Convolutional	512	1 × 1	
	Convolutional	1024	3 × 3	
	Residual			8 × 8
	Avgpool		Global	
	Connected		1000	
	Softmax			

Table 1. **Darknet-53.**

Fig. 3.19 DarkNet-53 CNN [9]

- We used IBM Cloud annotation Tool & LabellMg to draw bounding boxes around the accident region, and that tool will give us the compatible format of the dataset. After collecting the dataset, we manually selected the frame and used the IBM Cloud Annotation & LabellMg which will provide us model compatible dataset format.

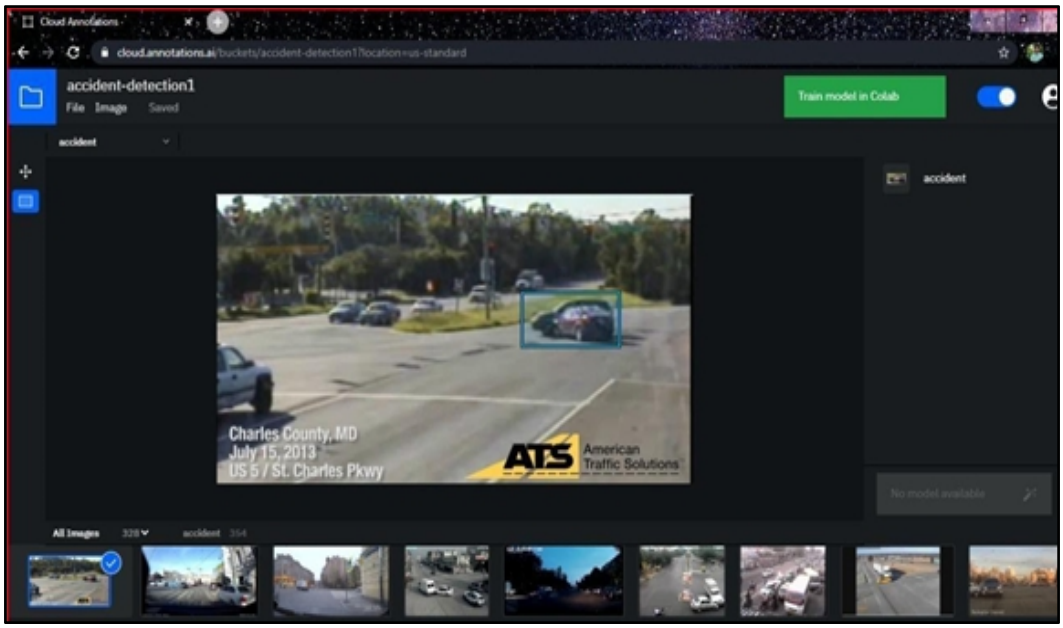


Fig. 3.20 IBM Cloud Annotation Tool

- Fig. 3.20 shows the dashboard of the IBM Cloud annotation tool's bucket.

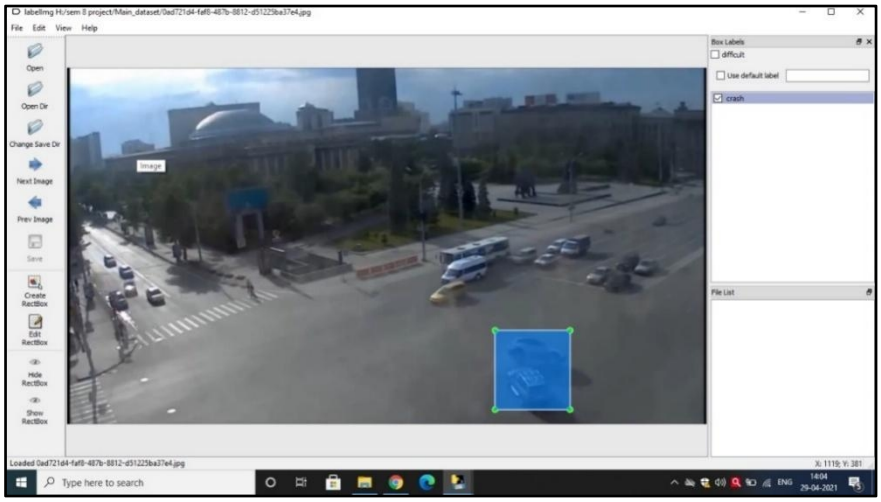


Fig. 3.21 LabellMg Annotation Tool

- Fig. 3.21 shows the Dashboard of LabellMg annotation tool.

- After hand labeling 500+ chosen accident frames & with the help of the IBM cloud annotation tool & LabellMg, we converted the dataset into the compatible format for each algorithm.

- After that, we used the darknet framework for the YOLO model, TensorFlow for SSD MobileNet & Faster RCNN for training on Google Collab.

- RoboFlow provides all of the tools you need to convert raw images into a custom trained computer vision model and deploy it for use in your applications. Today, RoboFlow supports object detection and classification models.

- We have used RoboFlow platform for splitting our dataset. It is a decent platform to upload your dataset, apply various pre-processing and augmentation techniques to your dataset. Fig. 3.22 shows the dashboard of RoboFlow.

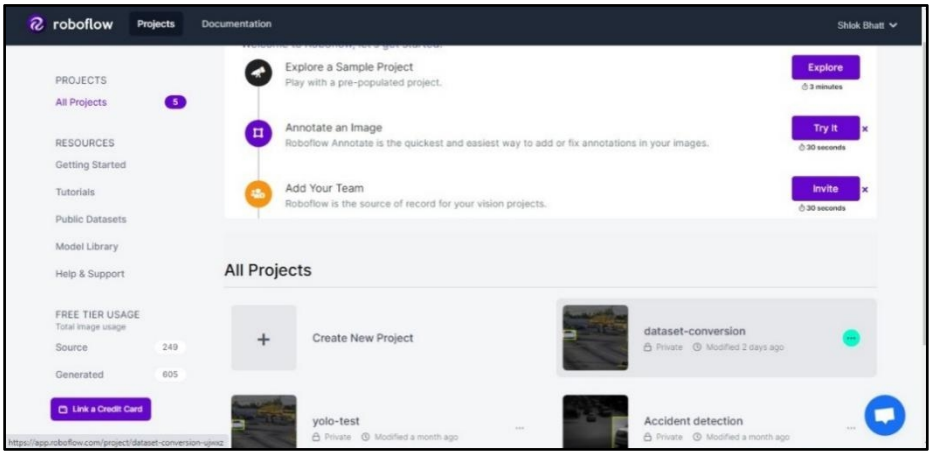


Fig. 3.22 RoboFlow Dashboard

- It also provides a facility to export your dataset in different formats like XML, TensorFlow record, PASCAL VOC, YOLOv3, etc. Also, there are various other useful functionalities and features available in this platform.

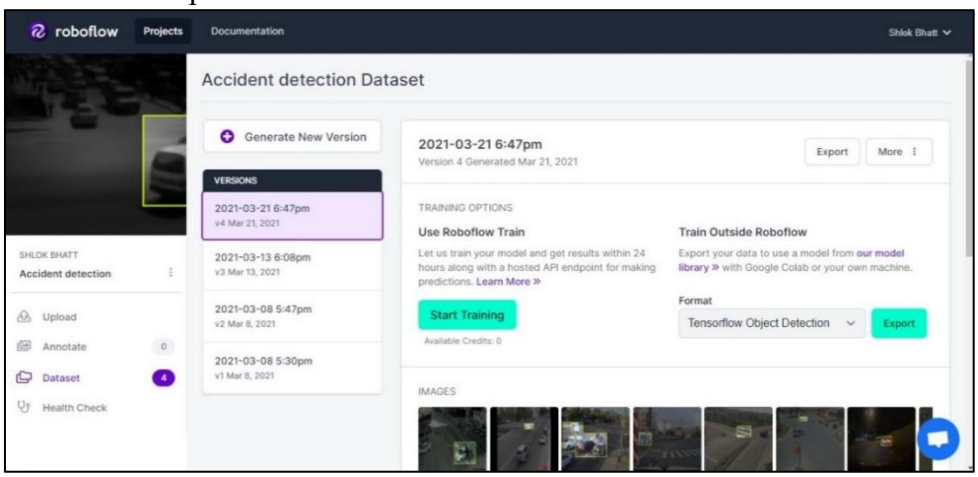


Fig. 3.23 RoboFlow Features

- Fig. 3.23 shows that it also provides a facility to export your dataset in different formats like XML, TensorFlow record, PASCAL VOC, YOLOv3, etc. Also, there are various other useful functionalities and features available on this platform.

CHAPTER IV: RESULT & ANALYSIS

4.1 Terminologies:

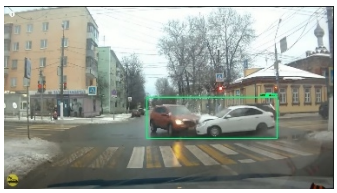
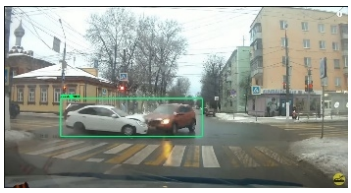
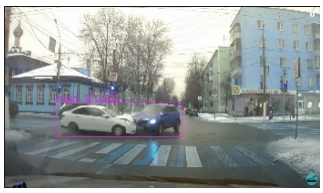
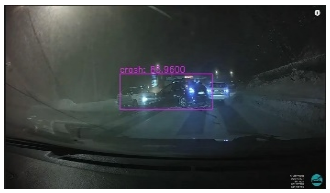
- **Threshold**: we predefine a threshold of IoU (for instance, 0.5) in classifying whether the prediction is a true positive or a false positive.
- **True positive (TP):** A true positive test result detects the condition when the condition is present.
- **True Negative (TN):** A true negative test result does not detect the condition when the condition is absent.
- **False positive (FP):** A false positive test result is one that detects the condition when the condition is absent.
- **False Negative (FN):** A false negative test result is one that does not detect the condition when the condition is present.
- The following Table 4.1 illustrates these four situations of two conditions combination.

Test **Condition**	**Present**	**Absent**
Positive	TP	FP
Negative	FN	TN

Table 4.1 Four situations of two conditions combination

- **Precision:** the number of true positives divided by the sum of true positive and false positive.
  $\text{Precision} = \frac{\text{TP}}{\text{TP+FP}}$ …. (2)
- **Recall:** the number of true positives divided by the sum of true positives and false negatives. However, the sum is just the number of ground truths, so it is not necessary to count the number of false negatives.
  $\text{Recall} = \frac{\text{TP}}{\text{TP+FN}}$ … (3)

4.2 Evaluation of YOLO v3, SSD MobileNet, Faster RCNN

Ground Truth Description	SSD MobileNet Output	Faster RCNN Output	YOLO v3 Output
Accident in frame (day time)			
Accident in frame (night time)			
Accident in frame (cloudy sky)			
Accident in frame (day time)			
Accident in frame (night time)			

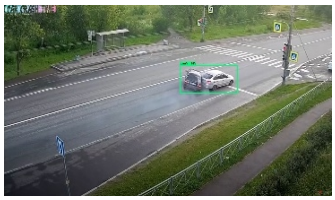
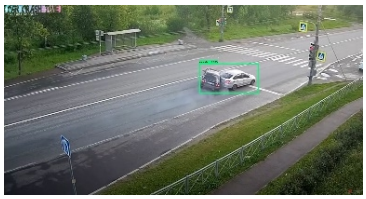
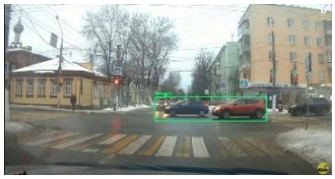
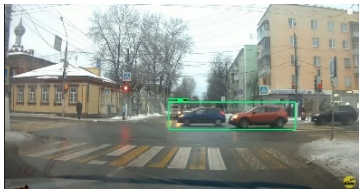
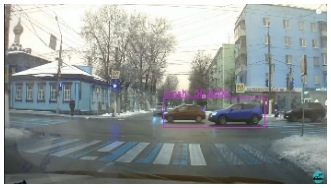
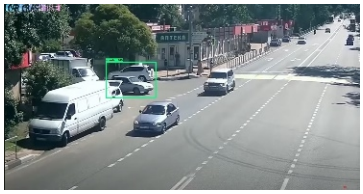
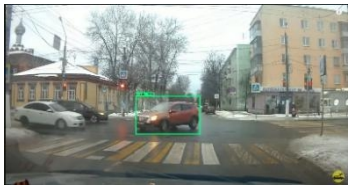
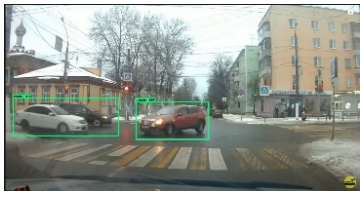
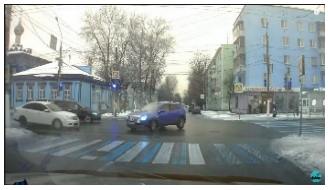
Accident in frame (day time)			
No Accident in frame (day time)			
No Accident in frame (traffic)			
No Accident in frame (traffic)			
No Accident in frame (day time)			

Table 4.2 Visual output of all the models in various scenarios

- We have set the threshold value for the confidence score as 0.5 for testing these ten images.

Ground Truth Description	SSD MobileNet Output description	Faster RCNN Output description	YOLO v3 Output description
Accident in frame (day time)	Detected accident with a very high confidence score(TP)	Detected accident with a very high confidence score(TP)	Detected accident with a very high confidence score(TP)
Accident in frame (night time)	Detected accident with a very high confidence score (TP)	Detected accident with a very high confidence score (TP)	Detected accident with a high confidence score (TP)
Accident in frame (cloudy sky)	Did not detect any accident (FN)	Detected two accidents in the image where one was TP and another was FP	Detected accident with a very high confidence score (TP)
Accident in frame (day time)	Detected accident with a very high confidence score (TP)	Detected accident with an average confidence score (TP)	Detected accident with a very high confidence score (TP)
Accident in frame (night time)	Detected accident with a below average confidence score (TP)	Detected accident with a below average confidence score (TP)	Detected accident with a below average confidence score (TP)
Accident in frame (day time)	Detected accident with a very high confidence score (TP)	Detected accident with a very high confidence score (TP)	Detected accident with a very high confidence score (TP)
No Accident in frame (day time)	Detected accident with high confidence score (FP)	Detected accident with high confidence score (FP)	Detected accident with an average confidence score (FP)
No Accident in frame (traffic)	Did not detect an accident (TN)	Did not detect an accident (TN)	Did not an detect accident (TN)

No Accident in frame (traffic)	Did not detect an accident (TN)	Detected accident with a very high confidence score (FP)	Did not an detect accident (TN)
No Accident in frame (day time)	Detected accident with a very high confidence score (FP)	Detected multiple accidents with a very high confidence score (FP)	Did not an detect accident (TN)

Table 4.3 Description of each output of the models in various scenarios

- **YOLO v3 Output:** YOLO v3 model performed quite decently on all the images. It took less loading time among all the three models as well as the confidence score was also high. In the above tables it can seen that barring a case of false positive in one of the frame, YOLO v3 performed very well on all the given images. It also had an high confidence score along with less loading time which indicates that this model could work well in real-time conditions.

- **Faster RCNN Output:** This model did not perform well on the given images. Also in the training, there was an issue in giving the required batch size due to the technical constraint of RAM. Hence, the model was not trained properly (If the aforementioned technical issue is resolved and the model is trained properly then it will have good accuracy with high confidence score). The model loading time was highest among all the three models and it also detected false positives which is evident in the above tables. The accuracy was also low compared to other models. All these indicate that Faster RCNN could not work well in real-time although it is one of the highest accurate object detection algorithms amongst its contemporaries due to high loading time.

- **SSD MobileNet Output:** This model also performed quite decently on all the images. The loading time was more than the Yolo V3 model but less than the Faster R-CNN model. Also, it detected a false negative as well as false positives (less than Faster RCNN) which is evident in the above tables. All this indicates that it could work well in real-time conditions but the accuracy could be compromised a little.

4.3 Comparison among the three algorithms

- For our accident detection task, we first trained our model using three different algorithms as described earlier. They are YOLO V3, SSD MobileNet, Faster RCNN.

- As per our analysis, we found the below results for Loading time of model, FPS & False-positive rate by running inference on the three models:
    - **Loading time:** YOLO V3 << SSD MobileNet < Faster RCNN
    - **FPS count:** YOLO V3 > SSD MobileNet > Faster RCNN
    - **False-positive rate:** YOLO V3 < SSD MobileNet < Faster RCNN
- Furthermore, we calculated Average Precision as well as Average Recall of the three models. In the bar graph(number) shown below, the comparison of YOLOv3, SSD MobileNet, and Faster RCNN for the aforementioned evaluation metrics is done. Fig. 4.1 shows the comparison of Average Precision, Average Recall and Detection time(in seconds) of all the three models.

Fig. 4.1 Bar Graph (Comparison)

4.4 Our custom trained YOLO v3 model

- After the evaluation of all the three models as well as comparing them, we finalized on YOLO v3 for the task of accident detection. Furthermore, our inference was in line with the general consensus of the YOLO v3 being the appropriate model for the real-time object detection as indicated by research in the area of object detection using deep learning models.
- We made some necessary changes as per our requirement in the config file of YOLO v3 model. Below, you can see the changes that we made:

config_file changes *:*

changes mades form default hyperparameters
max_batches = 2000 # no_of_classes = 2000*
steps = 80% of max_batches , 90% of max_batches = 1600 ,1800

for all yolo layers
[yolo] layer , no_classes = 1
*[convolution] layer above all [yolo] layer , filers = (no_classes + 5)*3 = 18*

- Fig. 4.2 shows the custom "config file" of our custom trained YOLO v3 model for the accident detection after the required changes.

```
[convolutional]
size=1
stride=1
pad=1
filters=18
activation=linear


[yolo]
mask = 0,1,2
anchors = 10,13,  16,30,  33,23,  30,61,  62,45,  59,119,  116,90,  156,198,  373,326
classes=1
num=9
jitter=.3
ignore_thresh = .8
truth_thresh = 1
random=0
```

Fig. 4.2 YOLO v3 custom config file

- In the above image you can observe how we have fine-tuned various hyperparameters like stride, padding, filter, activation function, etc. of the convolutional layers.

- Also, we have changed the batch size in the config file as per our requirement.

- Furthermore, in the yolo section in the config file, we have made the necessary changes in number of classes, anchors, etc. following the need of our accident detection.

4.5 Response system

```
import smtplib, ssl
import pymysql
from email.mime.text import MIMEText
from email.mime.multipart import MIMEMultipart
from email.mime.base import MIMEBase
from email import encoders
from db_support import *


def send_mail(confidence):

        sender = "flashback.0107@gmail.com"

        # receivers are fetched from Database
        receivers = get_rec()

        body_of_email = "Accident having confidence of " +str(confidence)+"%  is detected !!"
        msg = MIMEText(body_of_email, "html")
        msg['Subject'] = "Alert !! nearby accident detected !"
        msg['From'] = sender
        msg['To'] = ','.join(receivers)

        # we can send other deatils like accident_image in attachment
        s = smtplib.SMTP_SSL(host = 'smtp.gmail.com', port = 465)
        s.login(user = "flashback.0107@gmail.com", password = "dummy_f@123")


        s.sendmail(sender, receivers, msg.as_string())
        s.quit()
```

Fig. 4.3 Code snippet for sending an e-mail based alert

- Fig. 4.3 shows the code snippet of our e-mail-based response system. Here, we have used the necessary libraries and created a function called **send_mail** which sends an e-mail to a randomly fetched e-mail from the database.

- Furthermore, the below Fig 4.4 shows the code snippet of fetching the receiver's e-mail address from the database.

- For the aforementioned task, we have created a user-defined named **get_rec** which performs the fetching operation.

```
import pymysql
import random
emails = list()


def get_rec():
        db = pymysql.connect(user ="root" , password = "", host="localhost" , database = "crash")
        random_id = random.randint(1,4)

        print(random_id)
        cursor = db.cursor()

        sql = """ SELECT rec_email FROM `location_wise_contact` WHERE loc_id =""" +
                                str(random_id)+ """ """

        cursor.execute(sql)
        results = cursor.fetchall()
        db.close()


        emails.clear()
        for row in results :
                emails.append(row[0])

        return emails
```

Fig. 4.4 Code snippet for fetching receiver's e-mail address from the database

4.6 Running Object Detection:

4.6.1 Accident detection on the uploaded image

- On this web page, we can upload any image containing the accident, and then by clicking on the send button it performs the detection, and if an accident is detected with a confidence value higher than the threshold an email-based alert is sent to an email id associated with a particular location id selected randomly from the database. The database as shown in Fig. 4.5 contains the receiver's mail address with location_id. Below is a sample of that.

```
SELECT * FROM `location_wise_contact` ORDER BY `loc_id` ASC
```

loc_id ▲ 1	rec_email
1	jainilpanchal2000@gmail.com
2	bhattshlok12@gmail.com
3	17cp032@bvmengineering.ac.in
4	17cp021@bvmengineering.ac.in

Fig. 4.5 Database Snapshot

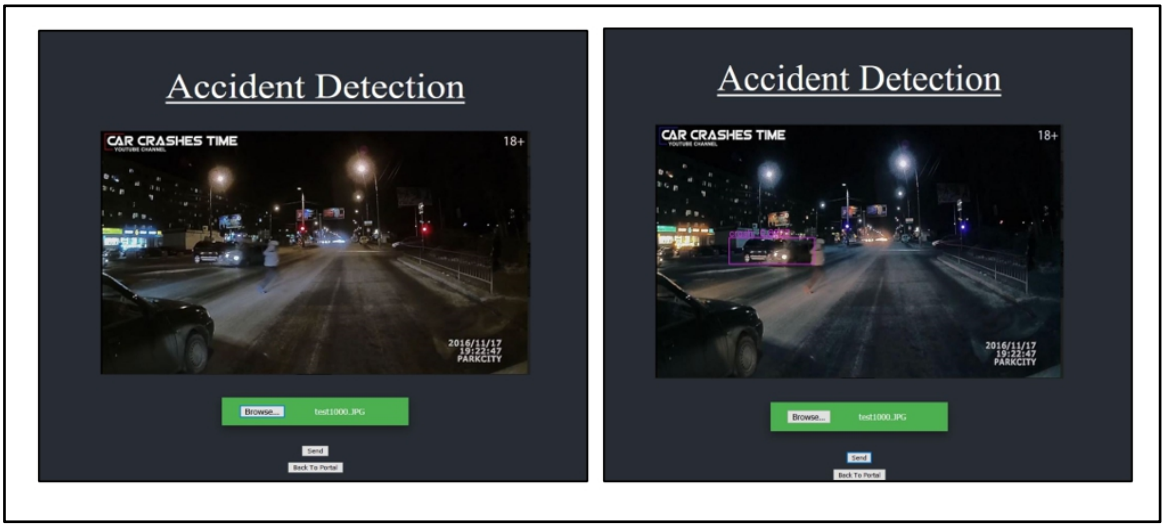


Fig. 4.6 Accident Detection on Image (on the portal)

- Fig. 4.6 shows the before and after of the accident detection running on the uploaded image.
- Fig. 4.7 shows the email-based alert : It contains the confidence score of the accident detected in the frame.

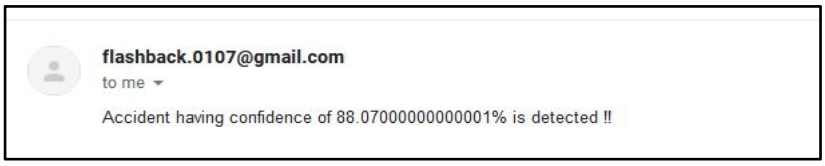


Fig. 4.7 Email Response

4.6.2 Accident detection on the video

- On this web page, a video clip containing accident/accidents is played, and then by clicking on the ON/OFF button it performs the detection, and if an accident is detected in a particular frame an email-based alert is sent to an email id associated with a particular location id selected randomly from the database. The email-based alert is the same as mentioned earlier. The FPS obtained here is quite low as the server runs on the local machine with Nvidia GTX 1050 GPU.

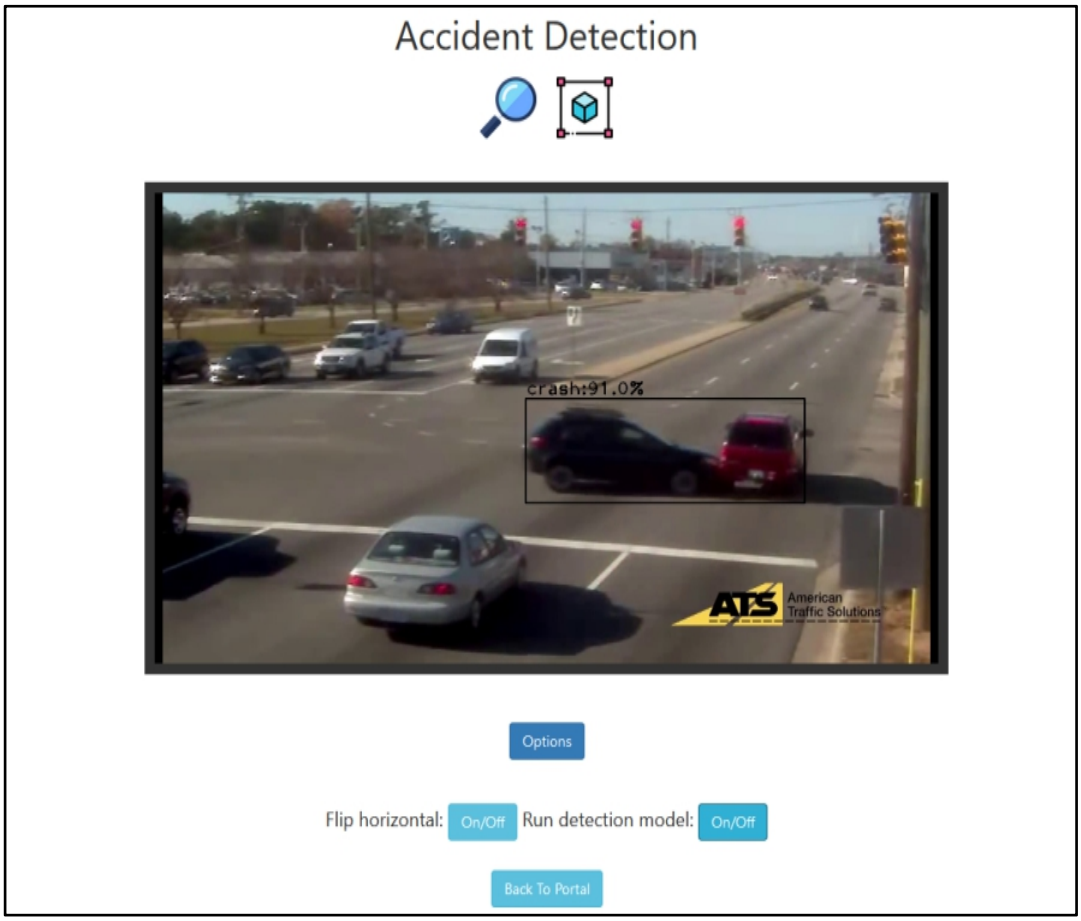


Fig. 4.8 Accident Detection on Video (on the portal)

- Fig. 4.8 shows the accident detection running on the video. Here, we can see the bounding box named “crash” with a confidence score showing that an accident has occurred.

CHAPTER V: SUMMARY, CONCLUSION & FUTURE WORK

5.1 Summary:

- To summarize our work, after identifying the problem we first did the literature survey to understand the work done in accident detection and tried to understand the major advancements and limitations in this area. After that, we started researching the datasets relevant to our project, and retrieved them. We decided to train and evaluate three different object detection algorithms for our accident detection task. The three algorithms were - YOLO v3, SSD MobileNet, and Faster RCNN.

- Furthermore, we started the process of hand-labeling the dataset and converted them into the compatible format of all three algorithms, and then trained our model on all the three algorithms. We tested our model on still images and on videos which forms the basis of our system. After necessary comparison and analysis, we finalized to use YOLO v3 for our system.

- We also created a web portal in which we can upload image or stream a video and check whether it contains an accident or not through the help of UI. Furthermore, we integrated it with a simple email-based alert system that sends an alert when an accident is detected on an email id fetched randomly from the database. The e-mail-based alert contains the confidence score of the detected accident. We also included some other tabs in the portal through which the admin/user can get additional information regarding our system.

- We also encountered some challenges which are as follows:

    - Footage of Accident has limited Access. It is governed by Traffic authorities or the Government and therefore not publicly available due to sensitive nature of recording such type of data
    - Scarcity of dataset containing accident footage on Indian roads
    - Training of model with dataset of limited variety & quantity
    - Low resolution of CCTV footage
    - Getting low FPS while running object detection on local machine (GPU : NVIDIA GTX 1050)
    - Integrating & coordinating all systems of rescue agencies

5.2 Conclusion:

- After working on this project, we can conclude that accident detection using CCTV footage can be a feasible and useful option compared to some other techniques if trained adequately as it can detect accidents within a short span of time with fairly decent accuracy.

- Also, as per the literature survey done by us as well as our evaluation and analysis of three object detection algorithms, it is evident that the YOLO algorithm outperforms various object detection methods and hence it can be used for such types of systems. Also, this type of system

can also help in analysis and research of the roads of our country which can help the Government to take relevant steps to reduce the accident rate across the country.

- Furthermore, we can also conclude that such types of more efficient and better-performing systems can be developed by the collaboration of the Government bodies with the engineers of various disciplines which will be beneficial to the society as well as the nation.

5.3 Future work:

- Including other important details in e-mail generated response along with confidence score
- Minimizing false predictions while detecting accidents
- Improving accuracy of the model during evenings & nights & detecting accidents in traffic
- Developing more efficient and effective response system

BIBLIOGRAPHY

[1] WHO, “Global status report on road safety 2018,”
Available at: https://www.who.int/violence_injury_prevention/road_safety_status/2018/enc

[2] Tian, C. Zhang, X. Duan and X. Wang, "An Automatic Car Accident Detection Method Based on Cooperative Vehicle Infrastructure Systems," in IEEE Access, vol. 7, pp. 127453-127463, 2019, doi: 10.1109/ACCESS.2019.2939532,
Available at: https://ieeexplore.ieee.org/abstract/document/8832160

[3] C. B. Murthy, M. F. Hashmi, N. D. Bokde, and Z. W. Geem, “Investigations of Object Detection in Images/Videos Using Various Deep Learning Techniques and Embedded Platforms—A Comprehensive Review,” Applied Sciences, vol. 10, no. 9, p. 3280, 2020,
Available at: https://www.mdpi.com/2076-3417/10/9/3280

[4] S. Ren, K. He, R. Girshick, and J. Sun, “Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks,” IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 39, no. 6, pp. 1137–1149, 2017,
Available at: https://arxiv.org/abs/1506.01497

[5] Adrian Rosebrock, “YOLO object detection with OpenCV”,
Available at: https://www.pyimagesearch.com/2018/11/12/yolo-object-detection-with-opencv/

[6] Sik-Ho Tsang, “Review: SSD — Single Shot Detector (Object Detection)”,
Available at: https://towardsdatascience.com/review-ssd-single-shot-detector-object-detection-851a94607d11

[7] NeuralNet.ai (Instagram) You Look Only Once,
Available at: https://www.instagram.com/p/CEMTuWMgt1s/

[8] Manish Gupta, YOLO — You Only Look Once “A State-of-the-Art Algorithm for Real-Time Object Detection System”,
Available at: https://towardsdatascience.com/yolo-you-only-look-once-3dbdbb608ec4

[9] Ammar, Adel and Koubaa, Anis and Ahmed, Mohanned and Saad, Abdulrahman,"Aerial Images Processing for Car Detection using Convolutional Neural Networks: Comparison between Faster R-CNN and YoloV3"
Available at:
https://www.researchgate.net/publication/336602731_Aerial_Images_Processing_for_Car_Detection_using_Convolutional_Neural_Networks_Comparison_between_Faster_R-CNN_and_YoloV3

[10] Chen Wang, Yulu Dai, Wei Zhou and Yifei Geng" A Vision-Based Video Crash Detection Framework for Mixed Traffic Flow Environment Considering Low-Visibility Condition", Journal of Advanced Transportation, vol. 2020, Article ID 9194028, Jan., 2020,
Available at: https://www.hindawi.com/journals/jat/2020/9194028/

[11] Huang, Y.-Q.; Zheng, J.-C.; Sun, S.-D.; Yang, C.-F.; Liu, J. Optimized YOLOv3 Algorithm and Its Application in Traffic Flow Detections. Appl. Sci. 2020, 10, 3079,
Available at: https://www.mdpi.com/2076-3417/10/9/3079/

[12] Joseph Redmon and Santosh Kumar Divvala and Ross B. Girshick and Ali Farhadi (2015). You Only Look Once: Unified, Real-Time Object DetectionCoRR, abs/1506.02640,
Available at: https://arxiv.org/abs/1506.02640/

[13] Joseph Redmon, Darknet: Open-Source Neural Networks in C. Darknet: Open-Source Neural Networks in C,
Available at: https://pjreddie.com/darknet/

[14] Pranali Satalgaonkar "Car-Accident-Detection-Using-Faster-RCNN",
Available at: https://github.com/PranaliSatalgaonkar/Car-Accident-Detection-Using-Faster-RCNN

[15] TensorFlow 2 Detection Model Zoo,
Available at:
https://github.com/tensorflow/models/blob/master/research/object_detection/g3doc/tf2_detection_zoo.md

[16] The AI Guy. (YouTube) YOLOv3-Cloud-Tutorial,

Available at (Video): https://www.youtube.com/watch?v=10joRJt39Ns,
Available at: https://github.com/theAIGuysCode/YOLOv3-Cloud-Tutorial

[17] ViAsmit, Object-Detection-YOLO (Flask + YOLO),
Available at (Video): https://www.youtube.com/watch?v=hOamcgGP73k,
Available at: https://github.com/ViAsmit/Object-Detection-YOLO

[18] Armaanpriyadarshan, Training-a-Custom-TensorFlow-2.X-Object-Detector,
Available at: https://github.com/ViAsmit/Object-Detection-YOLO

[19] Diegoinacio, object-detection-flask-OpenCV,
Available at: https://github.com/diegoinacio/object-detection-flask-opencv