Using Yolo V7 Development Of Complete Vids Solution Based On Latest Requirements To Provide Highway Traffic And Incident Real Time Info To The Atms Control Room Using Artificial Intelligence

Shubham Baliram Songire[1*], Dr.Uday Chandrakant Patkar[2], Parinita Jagannathrao Chate[3], Megha Adhikrao Patil[4], Lalita Kiran Wani[5], Aishwarya Sagar Pathak[6], Dr Shikha Bhardwaj Shrivas[7], Dr.Uday Patil[8]

[1*]Bharati Vidyapeeth's College of Engineering Lavale, Pune, Email: shubham.songire-bvcoel@bvp.edu.in,
uday.patkar@bharatividyapeeth.edu[2]
Parinita.Chate@bharatividyapeeth.edu[3]
Megha.Patil@bharatividyapeeth.edu[4]
lalita.wani@bharatividyapeeth.edu[5]
aishwarya.pathak@bharatividyapeeth.edu[6]
Shikha.Shrivas@bharatividyapeeth.edu[7]
uday.patil@bharatividyapeeth.edu[8]

***Coreesponding Author:** Shubham Baliram Songire

*Bharati Vidyapeeth's College of Engineering Lavale, Pune, Email: shubham.songire-bvcoel@bvp.edu.in,

Abstract: -
Roads are the most important mode of transport in India. It has a network of over 6,215,797 kilometres of roads as of 1 December 2021. India has the second-largest road network in the world, followed by the United States with 6,853,024 kilometres. Physical monitoring of such a large road network is not possible. Even the police cannot afford to stop the vehicle and fine them as it will lead to massive traffic jams. But this can be achieved through a smart computerized system which can detect and send e-challans to the violators.

This smart system will analyse the trends and help the government to focus where it needs to increase the size of roads and create alternative routes to manage heavy traffic. This allows the government to utilize its funds properly for road development which also benefits the people to get rid of traffic.

Video Incident Detection System (VIDS) is the easiest and smartest way to monitor traffic and check if the rules are being followed or not. It uses live camera feed to process video frames through the deep neural networks to identify image features and make decisions accordingly.

To make this system more efficient in these studies we have used the latest YOLOv7 model. Using artificial intelligence's deep neural network architecture YOLOv7 model is developed which is used for live object tracking this model has a higher frame rate and more accuracy than previous YOLO versions. This model has increased the frame rates from 5 FPS to 160 FPS which is best suitable for monitoring real-time traffic in a more efficient way. This model outperforms by using GPU-based systems which gives more computational speed and accuracy.

In these studies, we have proposed models for the vehicle classification system, speed and reverse driving detection system, helmet detection system, triple seat riding detection system, and number plate recognition system. This all models will help to make an autonomous traffic monitoring system and send e-challans to rule violators.

Keywords: ATMS, VIDS, YOLOv7, ANPR, GPU, Edge Device

Introduction: -

A country like India, with the second largest population in the world, is influenced by a huge dependence on its meager resources. Therefore, the demand for vehicles is continuously increasing. New vehicle manufacturing companies are coming to India to set up their manufacturing units which is affecting the fall in vehicle prices. These are the reasons for the increasing number of vehicles on the roads. This creates a traffic jam-like situation. Keeping an eye on such heavy traffic roads is a very challenging task for the traffic police. Stopping individual vehicles and keeping others waiting until the traffic police collects a challan from the violator is not a good way to monitor traffic.

Nowadays new superfast expressways like Samruddhi Mahamarg are getting developed in India with speed limits up to 150 km/h. To monitor such fast traffic moving highways smart solutions are necessary and using vision-based technology these things are possible. One of the best examples of a smart traffic management system is FAST Tag which is used throughout India at toll booths. FASTtag system detects the number plate coordinates using the object detection method and the OCR system reads the number from the segmented license plate image which is used for the automatic deduction of toll tax from the user's digital fast tag account. This method has been successfully implemented in India. Such a system makes a huge impact on crores of people's life. This saves fuel and time from long waiting queues at toll booths.

Development of the VIDS solution is based on the latest requirements to provide real-time incident detection and highway traffic detection to a centralized control room using the Neural Network-based system and API services.

This VIDS system consist of CCTV camera and video image processing units. Dedicated algorithms monitor the traffic data to take intelligent decisions. this algorithm performs remote incident detection at designated spots on the highway section and whenever any incident gets detected then it will send the output to the nearby police station from that location and centralized ATMS control room for storage, analysis, and reporting.

A video incident detection system (VIDS) will be installed in accident-prone places and other identified areas by enforcement agencies, also in vulnerable places where incidents happen usually. these images taken by VIDS camera shall be sent to the traffic management center or sub-centers in real-time using optical fiber cable, where specific images are shown on the large display monitors along with live traffic feed.

If any incident occurs, then the operator in that range can manually handle the VIDS camera and be able to take a feed of nearby VIDS cameras of that time. This will help the traffic police to track the incidents thoroughly to take appropriate actions on it. This system will help to mark dangerous areas where the incident happens more commonly, so the government can put warning boards on such places or Google Maps will auto those areas on their system. So, it will be life-saving for passengers who travel on such roads.

In these studies, we have tried to develop several end-to-end models which will help to monitor the traffic in efficient ways. these models include

1. Vehicle classification: we have proposed a model for vehicle classification which will be used to detect types of vehicles like Auto, Bike, Car, Light Commercial vehicle (LCV), Bus, Multi-axle vehicle (Truck), and Agriculture vehicles (Tractor). Using this we can send the proper category of e-challans to rule violators. This model is thoroughly trained on Indian vehicles to achieve the highest accuracy on Indian roads. This model will help to generate an automatic report on traffic flow, congestion, and density for each road and specific section of road so more arrangements can be made on such places.
2. Speed and Reverse Driving Detection: Every highway or expressway has its speed limits. To monitor whether speed limits are getting followed or not our model will

detect them autonomously. This will track the individual vehicle unique to them and find its speed, If it is found speeding then an e-challan will be sent to his number. Similar to lane detection here we have divided the road into two parts over the divider and given rules for the direction of vehicles for each lane if any vehicle goes the opposite way, then it gets identified using this proposed speed and lane detection model.

3. Helmet Detection: Wearing a helmet is lifesaving while accidents, also this is compulsory as per Indian laws. So, this model will identify whether the driver is driving with or without a helmet. If any motorcycle driver and his co-passenger are found without helmets, then an e-challan will be sent to his number.
4. Triple Seat Detection: Driving a motorcycle with three or more passengers is a crime according to Indian laws which also threatens the safety of other passengers. So, using the customized program for a person and motorcycle detection we have made this model which will detect triple-seat riding vehicles and punish them with e-challans.
5. Number Plate detection: To perform all above models smoothly it is important to get number plate of vehicle to send challans, so this model is made to detect the license plate very accurately. This model will detect the number plate using object tracking mechanism and read number written over it using OCR method.

This is an introduction to all the key features of our models which makes one end-to-end traffic monitoring system. This system can be implemented in real-time traffic to detect incidents and take appropriate action.

After the deployment of autonomous incident detection systems travellers will always try to follow the traffic rules because of the fear of VIDS cameras. Previously generally people knew which spots usually traffic police stop stays and in which spots they don’t, so accordingly people follow traffic rules. but after this VIDS cameras implementation autonomous incident detection will take place, so people will follow all the traffic rules which will increase the safety of travellers, and rules get followed strictly without any complaint. While driving on road there is no necessity that always we have to follow the rules, because other drivers' mistake also causes harm to a number of peoples within the seconds. So, for safer and more secure travel rules must be followed by everyone and our model will help them to get follow.

Python:

This project is made using the python programming language because AI algorithms and machine learning models are complex predictive technologies that are simplified using the python programming language. It has vast applications in this data science domain. With its clear code and lots of dedicated machine learning libraries, shifts the focus of the model towards algorithms than the syntax of the program. Python language is easy to learn, consistent, and intuitive. In this project, many python libraries are used like PyTorch, numpy, scikit-learn, scipy. Software solutions developed over python are platform independent, they can be built and run on any operating system platform like Windows, Linux, Mac, etc. This makes python programming language more convenient for programmers.

YOLO V7:

You Only Look Once (YOLO) is the open-source state of the art, real time object detection algorithm. YOLOv7 is the latest research work published in July 2022. This model is very fast compared to the previous ones. it detects images with the speed of 5 FPS to 160 FPS at 56.8% of mean average precision (AP) of the highest accuracy. This YOLO model is built over python’s PyTorch package. This is an open-source machine learning framework that accelerates the path from research prototyping to production deployment with faster computational speed.

As every model follows some basic structure. Depending upon model scale some model structures may vary but this is the generalized structure of YOLO in which:

- Backbone part is a convolutional neural network (CNN) that pools image pixels to form features at different granularities.
- Feature pyramids are used to detect objects at different scales.
- Neck combines and mixes the convolutional network layer representations and passes them to the prediction head.
- Head is the final part of CNN where the YOLO model predicts the bounding box's locations and the classes of the bounding boxes.

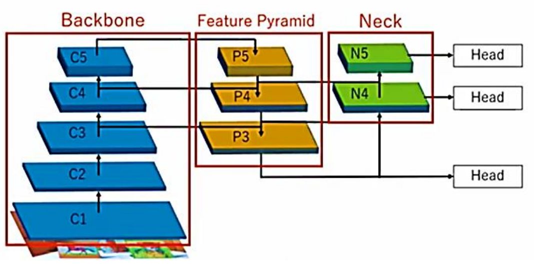


Fig. 1: YOLO model graphical presentation

Models trained in these studies are trained using a transfer learning approach over YOLOv7's pretrained weights file from YOLOv7 official GitHub repository.

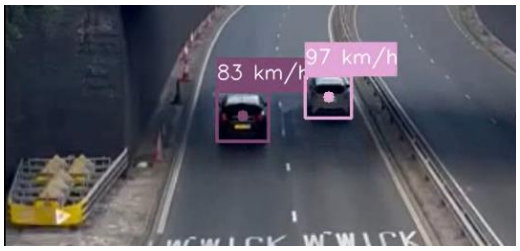


Fig. 2: Object Tracking using YOLOv7

The process of training a custom model includes several steps. The first step is to collect the dataset and labelled images dataset using tools like LabelImg. For data collection, images are downloaded using the batch download technique which is majorly used here for data collection and some Kaggle datasets are used.

DeepSort:

DeepSORT is a computer vision tracking algorithm used for tracking objects by assigning a unique ID to each passing object. Trackers are very faster than detectors. because of this, trackers can be used in real-time scenarios and has many applications in the real world. Multi-camera surveillance can also be possible using this algorithm. when an object gets tracked through one camera with a specific ID after passing it through another camera the same ID will remain to continue to that object. this will help to find the path of the moving vehicles. DeepSort uses objects spatial and temporal features to track the object throughout the video.

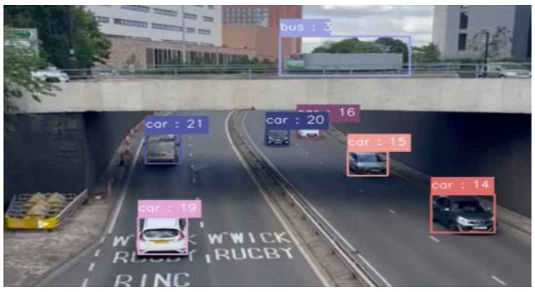


Fig. 3: Deesort algorithm working. Unique Id's are allocated to each vehicle.

As shown in figure (3) the VIDS system starts following vehicles whenever they come under the camera range, it tracks the vehicle throughout the camera feed, but this thing is also possible that it will record the same incident a number of times. To avoid this thing in our studies modified deepsort version is used. Deepsort will Generate one unique ID for each vehicle passing and record the incident once per ID which prevents the duplicate record generation of the same event.
The deepsort algorithm consists of 4 steps. In the first step, objects get detected using the YOLO model after this the second step is multi-object tracking, in this step Kalman filter is used to process the correlation of frame-by-frame data. this uses the position and velocity of an object to predict where it is likely to be. then Hungarian algorithm is used for correlation measurement and finally, CNN based neural network is used for training and feature extraction.

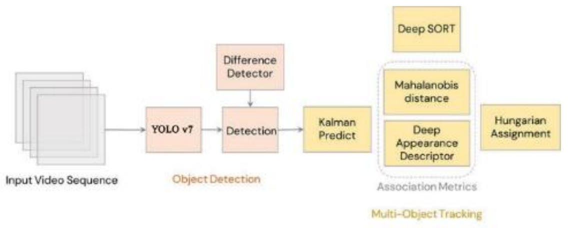


Fig. 4: Deepsort algorithm architecture diagram

This deepsort algorithm is used in our studies over each individual model to generate unique Id for objects. To apply this algorithm, a few steps are followed which are as follows:

1. All files from the Deepsort GitHub repository are cloned inside the YOLO model folder.
2. Then Inside cfg folder customized weight yolov7.yaml file is copied from the YOLO models directory. The same process is followed for coco.yaml file inside cfg folder.
3. Inside the demo.py file paths of the input video, the output video, and YOLO weights are set properly.

After these three steps unique id gets generated over each object as per our YOLO models customized trained classes.

Vehicle Classification:

Making a VIDS system for monitoring Indian roads, and vehicle classification is the first step in this process to make the system more accurate. we have collected vehicles images dataset of Indian roads, so classification can be performed at greater accuracy, and no one will suffer from anonymous challans after deployment of this model.

To classify the vehicles in our study we have made a classification based on the government of India's fast tag policy. In this model, we have taken 7 classes to generalize the model which are Auto, Bike, Car, Light Commercial vehicle (LCV), Bus, Multi-axle vehicle (Truck), and Agriculture vehicle (Tractor). These are the broader categories that make classification according to law. All these classes have different tax slabs in Indian law. So, we can send an accurate challan to them.

To collect the dataset for this model, we have used CCTV camera footage that is set over Indian highways. Using image processing tools, frames are taken from the videos with a gap of 15 seconds each. Using this method, we made the dataset of 4000 images, and another 2000 images were collected through Google using batch download methods.

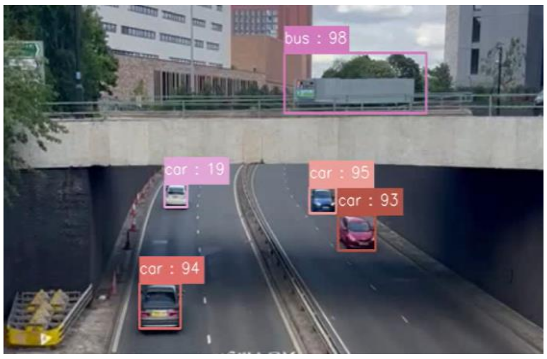


Fig. 5: Vehicle classification model detections

This model is made such that whenever any incident gets detected, the model will store details of vehicle class type from our 7 classes (Auto, Bike, Car, Light Commercial vehicle, Bus, Multi-axle vehicle, and Agriculture vehicle), the time stamp of that event and the camera ID from the highway where the incident get detected. Vehicle class will help to generate challan bills, and timestamp will help to record the incident time so in the case of big accidents police can check the feed from all nearby VIDS cameras and take relevant action on a specific incident, also CCTV footage can be used by police to present the incident evidence in courtroom trials.

Speed and Lane Detection:

Speed detection is one of the features of this model. every highway has its speed limits. Overcrossing this speed limit can create panic-like situations for travelers or major incidents can happen. Also, manually police can't be able to record the number plate of such over speeding vehicles, but our VIDS system is able to track the speed of every passing vehicle and using ANPR it will send an e-challan to rule violators.

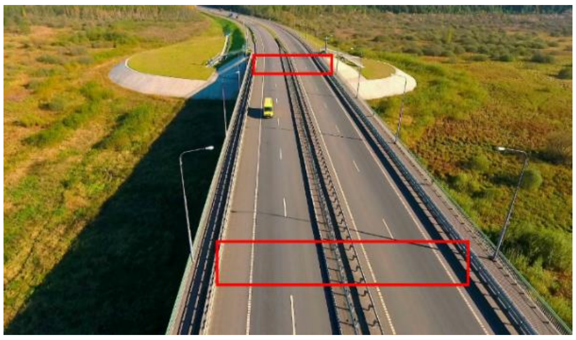


Fig. 6: Virtual polygons over highway to calculate vehicle speed.

In this study to make the speed detection algorithm, 2 polygons are virtually drawn over the highway as shown in figure(6) and whenever vehicles pass through any of the polygons it will start tracking them till another polygons. Speed calculation is performed using the time taken by the vehicle to cross the distance between one polygon to another. using the formula of speed,

$$\text{Speed} = \frac{\text{Distance}}{\text{Time}}$$

here we have taken the distance as the predefined distance between two polygons and time is calculated as the total time taken by a vehicle to pass one polygon to another. So, using this predefined distance and the recorded time, the speed of the vehicle gets calculated. Whenever any vehicle tries to cross the speed limit, the VIDS system mark that incident and will take further decisions according to pipelined process.

Code snippet for the speed detection algorithm:

```
vehicles_entering = {}
vehicles_elapsed_time = {}

area_1 = [(180,520),(475,499),(530,548),(163,583)]# two polygons drawn
area_2 = [(127,684),(670,625),(718,664),(109,736)]

cx = int((bbox[0] + bbox[2]) / 2)  #center x and y of each passing vehicle .
cy = int((bbox[1] + bbox[3]) / 2)

result = cv2.pointPolygonTest(np.array(area_1, np.int32), (int(cx), int(cy)), False) # validates car center passed polygon
or not

if result>=0:
   vehicles_entering[track.track_id]=time.time() # store starting time with its id in dict.

if track.track_id in vehicles_entering:    # if vehicle from vehicle entering dictonary is passed through another polygon
   result = cv2.pointPolygonTest(np.array(area_2, np.int32), (int(cx), int(cy)), False)
   if result>0:  # if passed calculate total time taken
      elapsed_time = time.time() - vehicles_entering[track.track_id]
      if track.track_id not in vehicles_elapsed_time:
         vehicles_elapsed_time[track.track_id] = elapsed_time
      if track.track_id in vehicles_elapsed_time:
         elapsed_time = vehicles_elapsed_time[track.track_id]
      # Calculate average speed
      distance=30 # meters
      a_speed_ms = distance/elapsed_time
      a_speed_kh = a_speed_ms*3.6
      color = colors[int(track.track_id) % len(colors)]
      color = [i * 255 for i in color]
      cv2.rectangle(frame, (int(bbox[0]), int(bbox[1])), (int(bbox[2]), int(bbox[3])), color, 2)
       cv2.rectangle(frame, (int(bbox[0]), int(bbox[1]-30)), (int(bbox[0])+(len(class_name)+len(str(track.track_id)))*17,
int(bbox[1])), color, -1)
            cv2.putText(frame,  str(int(a_speed_kh))+'  km/h',(int(bbox[0]),  int(bbox[1]-11)),0,  0.6,  (255,255,255),1,
lineType=cv2.LINE_AA)
      cv2.circle(frame, (cx, cy), 5, color,-1) # marking center of vehicle
if verbose == 2:
     print("Tracker ID: {}, Class: {},  BBox Coords (xmin, ymin, xmax, ymax): {}".format(str(track.track_id),
class_name, (int(bbox[0]), int(bbox[1]), int(bbox[2]), int(bbox[3]))))

for i,area in enumerate([area_1, area_2]):
      if i==1:
         continue
```

In this code Initially, two dictionaries are made to store the information of car ids that are passing through area 1 with their time stamp. If that same car Id passes through the second area, then it will count vehicle speed using the speed formula and store its information in another dictionary. Here fixed distance is used as 30 meters, which can be changed according to the distance between two marked areas.

For lane detection, one virtual line is drawn on the divider of the road. To identify the vehicles running in opposite direction program is made such that it marks the vehicles which are traveling from area 2 to area 1, as we have given area 1 name to the first passing polygon of each lane and area 2 to the second polygon which comes in a way.

ANPR:
Automatic number plate recognition (ANPR) is the system to detects the number plate using a CCTV camera feed and reads numbers written over the segmented image of the license plate, optical character recognition (OCR) system is used. This process has become a matter of milliseconds because of deep learning and OCR. Within seconds driver gets an e-challan for his rule violation.

This system can be used in traffic management, smart parking, toll booths, and validating the PUC deadline of vehicles to make humans less and instant autonomous systems. This system can also help to find the stolen and non-registered vehicles.

This model works in 2 steps:
1. The first step is number plate detection. VIDS camera is used to take a live feed of the road, and it gets passed to the YOLO model. The YOLOv7 model is trained to detect number plates with higher accuracy. This makes a bounding box over the number plate of the vehicle. Which is then used for number reading.
2. Second part is Recognition. A number plate inside the bounding box is considered for this step. A segmented image of a number plate is given as input to the OCR system which detects the characters inside the image and detects the text of the number plate and passes it to the server for pipelined models working.

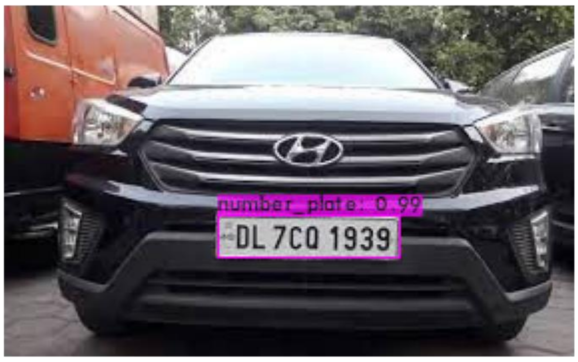


Fig. 7: ANPR model detections

Factors affecting ANPR success:
1. The input cameras for this ANPR system requires to be fitted at a little height from the ground so they can capture the number plates easily and clearly. The distance between the camera and the car should be as minimum as possible to get better results. If we set the camera on the top of the light pole, then the number plate can't be visible clearly which results in a decrease in the accuracy of working of the model.

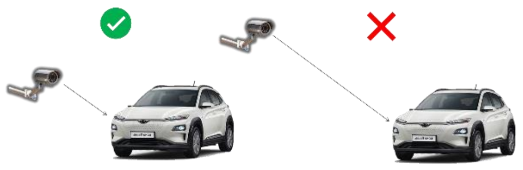


Fig. 8: Ideal position for camera distance

2. The angle made by the camera must be clear such that it captures the number plate from the front side only. if the camera is getting input from the top of a car, then for sure it can't be able to capture the number plate. It is commonly advised to maintain camera angle less than 45° from vertically and horizontally as well.

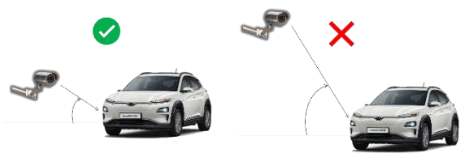


Fig. 9: Ideal position for camera angle

3. Setting the proper zoom-in level, the camera will capture only road view instead of other roadside noise. So, by giving proper zoom-in level, YOLO algorithm's accuracy level gets increased significantly.

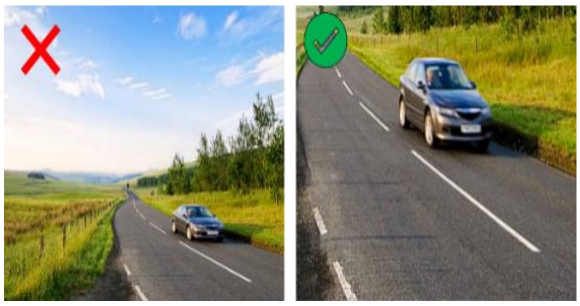
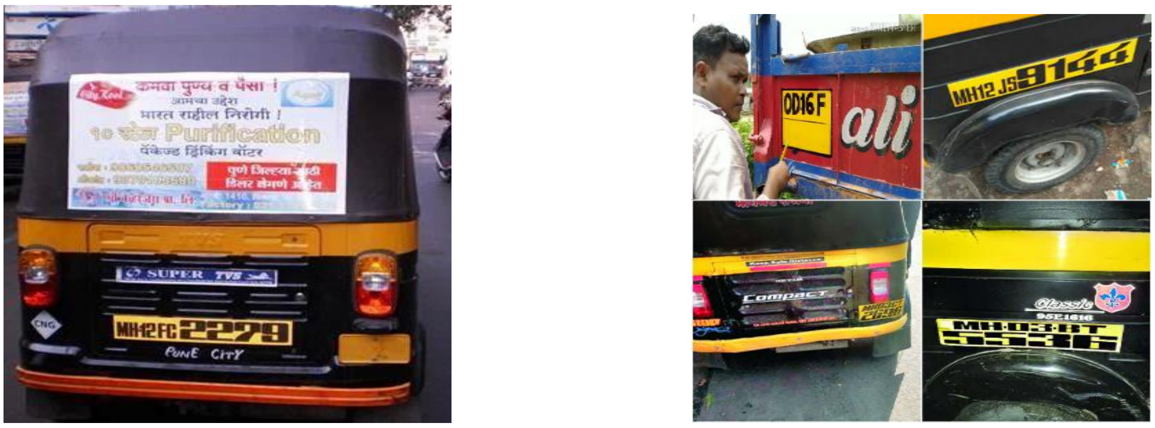


Fig. 10: Ideal zoom range for VIDS camera

To make this model, the dataset from Kaggle and google images are collected and then labelled using the LabelImg tool. labelimg is a tool which is simple and fast to make bounding boxes for YOLO, CreateML and PascalVOC model formats. For training the ANPR model 1733 annoted images are used with 70% and 30% split for training and testing.



(a) (b)

Fig. 11: Special cases of number plate

Just as shown in figure (11) these are some special cases that are considered while collecting datasets for this model. vehicles like this that don't have number plates, have one small, colored portion of their vehicle as a number plate. More of such images are used to get the model trained well in these cases.

While labeling the model single class is used which is "number_plate". All images are labelled and then they are divided into 70% and 30% for train and test data as 2 new folders. In each folder images and labels are named 2 new folders are created to store images and their corresponding labels. YOLOv7 pretrained weights are used for transfer learning purposes. yolov7-custom.yaml named file is created in "cfg" folder in which the number of classes is taken as 1 and all yolov7 layers are kept as it is. Then another file is created with the name of custom_data.yaml which consists of information about the train and validation dataset path, number of classes, and names of each class are saved inside the "data" folder, downloaded from GitHub repository.

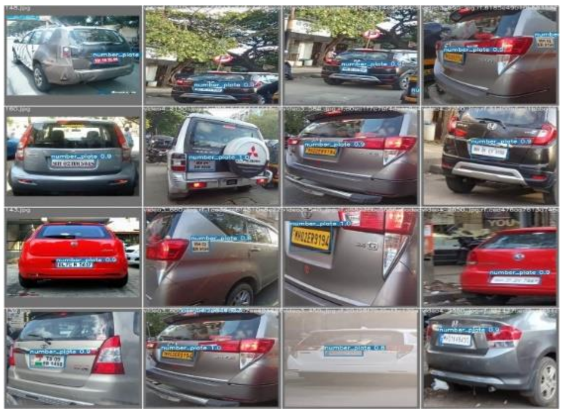


Fig. 12: ANPR model detections

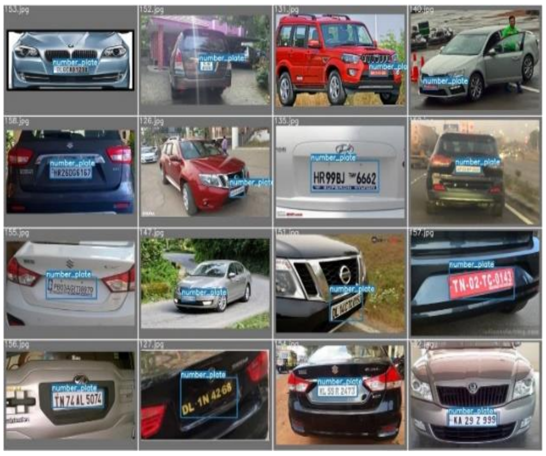


Fig. 13: ANPR model detections

This model is trained for 100 epochs and it has taken around 1.5 hours to completely train on number plate detection data. Finally, one API is made for this model which gives outputs as detected number plate characters and details about the timestamp of the incident, type of vehicle detected, license plate image, and camera number through which this number plate gets detected.

Helmet detection:

Road accidents are a major cause of concern in India. Each year three to five percent of India's GDP is used for road accidents. India has about 1% of world's vehicle population and contributes 6% to global road accidents.

Trauma to the brain can occur due to sudden stop, quick turn, or accident which can cause a skull fracture, jarring motion, head injuries, memory loss, sleep disorder sometimes permanent disability or death. To be safe from such road accidents government has already made strict rules to use helmets while driving a vehicle.

The helmet is a life savior in accidents. Studies have shown that sudden fall or collision helmet absorbs maximum impact energy which prevents the human brain from serious injuries or death. A helmet must be worn by everyone who is sited on the bike. Not just for adults even for small kids, helmets are available in the market. As we use proper clothes to cover our body helmets also be fit and steady on our head.

According to the Indian motor vehicle act, the helmet is compulsory for every individual over 4 years of age. In 62% of cases involving the death of two-wheeler, death occurs due to serious injuries to the head. Over 17% of road accidents are happens on straight roads and the maximum time two-wheelers are part of such accidents. So, helmets are a must to keep safe us from such accidents.

To make this helmet detection model, in these studies we have used an image dataset from Kaggle which consists of 1000 images for training and 300 images for testing. We have added some special cases here because our Indian government rules have given exemptions to the Sikh community for helmet compulsion. So, have added more Sikh people's vehicle driving images so they don't get challan.

Inside the train and test directory two new folders are created one contains images and another contains their labels. To detect if the person has worn the helmet or not, here 2 classes are made for this model labeling which are "with helmet" and "without helmet". YOLOv7 pretrained weights are used for transfer learning to train the model. The new file is created with the name yolov7-custom.yaml in "cfg" folder in which the number of classes are taken as 2 and all yolov7 layers are kept as it is. Another file is created with the name of custom_data.yaml which consists of information about train and validation dataset path, number of classes and names of each class and saved inside "data" folder of downloaded Github repository.

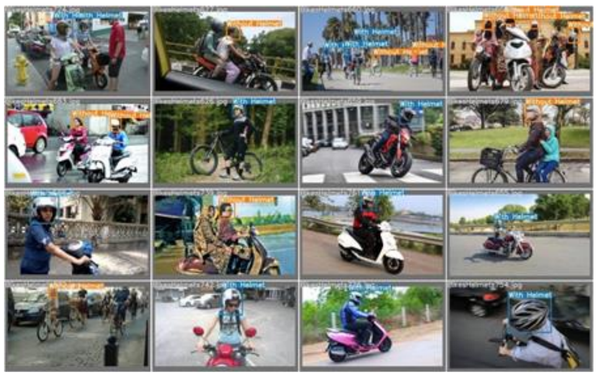


Fig. 14: Helmet detection model detections

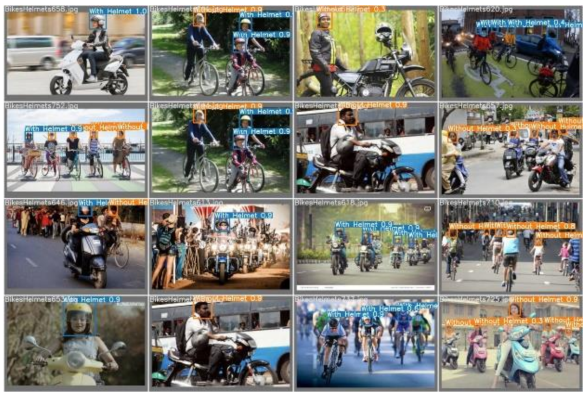


Fig. 15: Helmet detection model detections

These are some outputs that are predicted by the model after the successful completion of the training process. For the training, we have used 100 epochs. This model has taken around 1.2 hours to complete the training process.

Finally, API has made to take the outputs from the model as images of without helmet rule violators images, timestamp of the event, and camera id through which the incident gets detected. All these details are saved on the server for further processing.

Triple seat detection:
Triple seat motorcycle driving is a crime according to Indian laws, but many people do not follow it. To catch such people, we have made this model. This completely works over YOLOv7's pretrained weights file. From the weights file, we have considered 2 classes, motorcycle, and person. The class number for the person is 0 and the motorcycle has the number 3. So, we will get outputs in this format.

To make this triple seat detection model rectangle overlapping method is used here. Once the model starts getting VIDS camera footage this algorithm will find people and motorcycle from frames and start making bounding boxes over them. if the bounding box of motorcycle class overlaps with the more than 3 person classes then that event is considered as a triple seat motorcycle driving.

In this rectangle overlap method, two rectangles are said to be overlapped if area between that two rectangles will become greater than zero. Here coordinates of bounding boxes made by yolo model are used as top left and bottom right coordinate of each bounding box is considered and named as (R[0], R[1]) and (R[2], R[3]) respectively. For every two bounding boxes it will perform overlap detection. For any 2 bounding boxes named R1 and R2, they will collide each other if they do not satisfy this condition.

```
(R1[0]>=R2[2]) or (R1[2]<=R2[0]) or (R1[3]<=R2[1]) or (R1[1]>=R2[3])
```

The code snippet for this rectangle overlap program is as follows

```
if len(det):
    for R1 in boxes_3:
        c1 = 0
        for R2 in boxes_0:
            if (R1[0]>=R2[2]) or (R1[2]<=R2[0]) or (R1[3]<=R2[1]) or (R1[1]>=R2[3]):
                pass
            else:
                c1+=1
                if c1==3:
                    print("Tripcy found")
```

firstly, I have stored all the person and motorcycle box coordinates in boxes_0 and boxes_3 lists respectively, and here using them in the loop. In this program, if the motorcycle class is detected then it will take the bounding box coordinates of the person class one by one. If that person's box overlaps, then the count of c1 gets increased by one. likewise, if we get 3 and more person classes with one motorcycle class then it will print that event as Tripcy Found and on the GUI screen, it will give the label as triple_seat. API of this model stores information about the timestamp of the event, detected triple seat driving image frame and camera id which has detected this event.

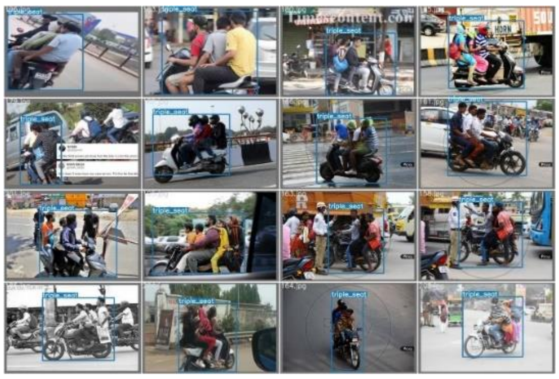


Fig. 16: Triple seat driving model detections

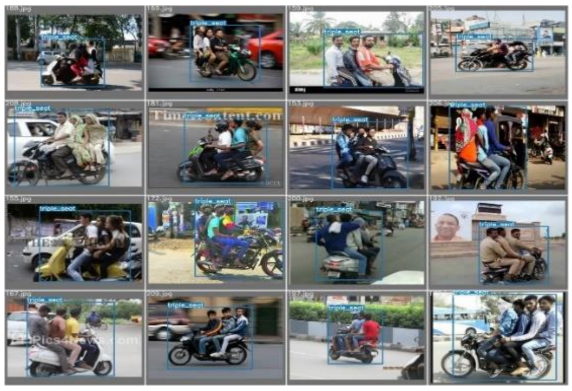
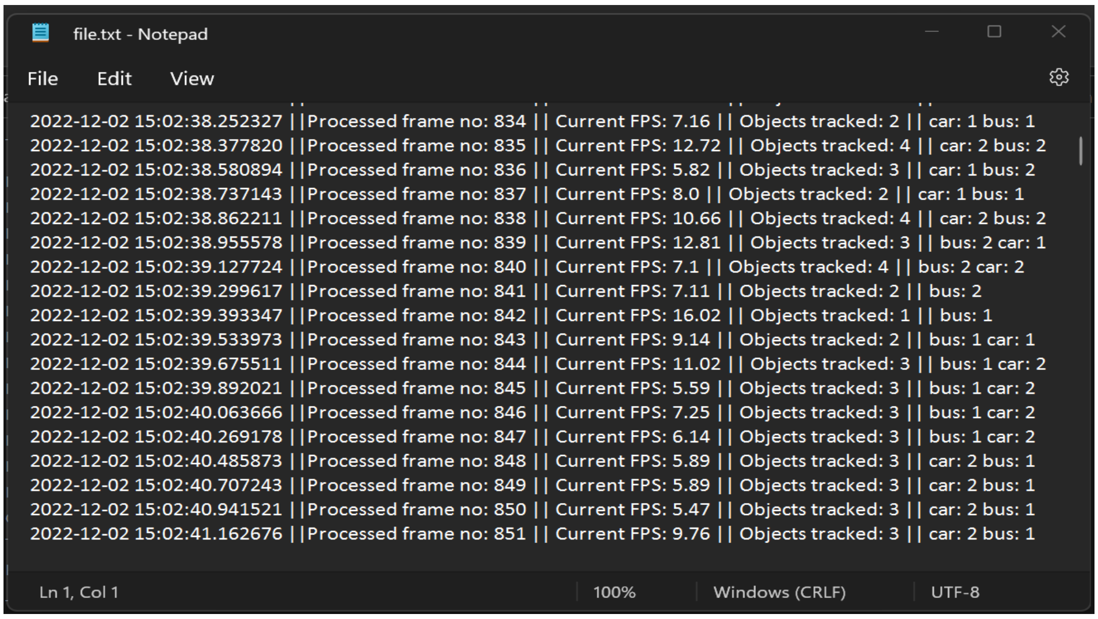


Fig. 17: Triple seat driving model detections

Implementation

Deployment of this model can be possible on both CPU and GPU, but it will affect the performance of the model accordingly. generally, GPUs are preferred for the deployment of yolo models.

To train all these models NVIDIA RTX 3070 GPU is used with 8 GB VRAM. These models generate log files on real-time traffic data. This stores the data of the timestamp of each frame with a number of detections performed over that frame. Type of each class with its occurrences in the frame. Everything gets recorded in the background with the log file. This file gets continued if the camera feed stops in any circumstances so it will never miss a single frames record generation.



```
2022-12-02 15:02:38.252327 ||Processed frame no: 834 || Current FPS: 7.16 || Objects tracked: 2 || car: 1 bus: 1
2022-12-02 15:02:38.377820 ||Processed frame no: 835 || Current FPS: 12.72 || Objects tracked: 4 || car: 2 bus: 2
2022-12-02 15:02:38.580894 ||Processed frame no: 836 || Current FPS: 5.82 || Objects tracked: 3 || car: 1 bus: 2
2022-12-02 15:02:38.737143 ||Processed frame no: 837 || Current FPS: 8.0 || Objects tracked: 2 || car: 1 bus: 1
2022-12-02 15:02:38.862211 ||Processed frame no: 838 || Current FPS: 10.66 || Objects tracked: 4 || car: 2 bus: 2
2022-12-02 15:02:38.955578 ||Processed frame no: 839 || Current FPS: 12.81 || Objects tracked: 3 || bus: 2 car: 1
2022-12-02 15:02:39.127724 ||Processed frame no: 840 || Current FPS: 7.1 || Objects tracked: 4 || bus: 2 car: 2
2022-12-02 15:02:39.299617 ||Processed frame no: 841 || Current FPS: 7.11 || Objects tracked: 2 || bus: 2
2022-12-02 15:02:39.393347 ||Processed frame no: 842 || Current FPS: 16.02 || Objects tracked: 1 || bus: 1
2022-12-02 15:02:39.533973 ||Processed frame no: 843 || Current FPS: 9.14 || Objects tracked: 2 || bus: 1 car: 1
2022-12-02 15:02:39.675511 ||Processed frame no: 844 || Current FPS: 11.02 || Objects tracked: 3 || bus: 1 car: 2
2022-12-02 15:02:39.892021 ||Processed frame no: 845 || Current FPS: 5.59 || Objects tracked: 3 || bus: 1 car: 2
2022-12-02 15:02:40.063666 ||Processed frame no: 846 || Current FPS: 7.25 || Objects tracked: 3 || bus: 1 car: 2
2022-12-02 15:02:40.269178 ||Processed frame no: 847 || Current FPS: 6.14 || Objects tracked: 3 || bus: 1 car: 2
2022-12-02 15:02:40.485873 ||Processed frame no: 848 || Current FPS: 5.89 || Objects tracked: 3 || car: 2 bus: 1
2022-12-02 15:02:40.707243 ||Processed frame no: 849 || Current FPS: 5.89 || Objects tracked: 3 || car: 2 bus: 1
2022-12-02 15:02:40.941521 ||Processed frame no: 850 || Current FPS: 5.47 || Objects tracked: 3 || car: 2 bus: 1
2022-12-02 15:02:41.162676 ||Processed frame no: 851 || Current FPS: 9.76 || Objects tracked: 3 || car: 2 bus: 1
```

Fig. 18: Logs file generation in background

System setup for model deployment:
With the latest NVIDIA RTX 4090, Cuda enabled GPU which gives the speed of 230 FPS on the YOLOv5 Nano model. This GPU is capable to run all models of the YOLO family with more than 30 FPS speed for 640*640 resolution images. AI GPUs like Tesla v100 and Tesla P100 can generate the best and constant results. According to the size of the model, this gives more FPS on smaller as compared to larger ones. This will not be always the case with the GTX 1080 Ti GPU. Even then, in some cases with older AI GPUs like P100, it is observed that smaller models run slower compared to larger models. For instance, YOLOv6-Tiny is running at 77 FPS, while YOLOv6-Nano, which is smaller than Tiny, is running at 71 FPS.

If the model performance is more than 30 FPS then it will be best suitable for real-time object detection and tracking. So, any GPU can be used to deploy YOLO models which is capable to gives more than 30 FPS speed.

If the model performance is more than 30 FPS then it will be best suitable for real time object detection and tracking. So, any GPU can be used to deploy YOLO models which is capable to gives more than 30 FPS speed.
Running YOLO model on CPU:

YOLO models run on a CPU, but speed and accuracy may vary according to the architecture of the CPU. Smaller models perform well on the CPU as compared to larger models.

GPU VRAM needed to train YOLO models:
There is no such limit for VRAM. More the size of VRAM we can take greater batch training size. VRAM affects the batch training size of models. But it is not compulsory to have greater size VRAM, with limited size also we can train YOLO models with less batch size. Small batch sizes will take more training time. One should need to be careful while giving this parameter otherwise error ‘CUDA out of memory’ will encounter.

After the successful development of all individual models, the final step is to make them run on real-time traffic feed. So, to deploy this, cameras are settled on real highways, giving a unique id to each camera. To directly connect these cameras to the cloud is not a good idea because the internet may be an issue in some areas, also continuous use of cloud services costs a lot of charges. So, an edgebox is the best way to implement these models. In the market, there are many companies that provide these edgebox devices. Programs are made such that they will store nearly 10 hours of daily data on these edge devices and then it will be sent to our cloud account. This will save a lot of deployment costs. For every 4 cameras, there is one edgebox device to perform operations. All the data of sequential 4 VIDS CCTV cameras get locally stored on edgebox for the cycle of 10 hours.

For further training and increasing accuracy, stored highway data from the cloud get used to the training model again and again after a few days of time. This process will increase the accuracy of the model with new coming data, also the right people will get challans. instead of guilty ones if normal people start receiving such challans, then it will make a negative impact on people’s mindset toward VIDS systems, so continuous model improvement is a must for these systems.

Conclusion:

To do our day to day’s activities everyone travels on the roads but there are many rules which drivers and travellers need to follow while traveling on road. to monitor whether all those rules are getting followed or not is a challenging job for traffic police. So, a few of these challenges can be solved using our proposed models. This will best fit for real-time monitoring of incidents.

In this study, we have used the YOLOv7 model which is faster than previous models used for real-time traffic monitoring. This will reduce human efforts and increase the efficiency of traffic monitoring. These models can be used in both daytime and nighttime using appropriate cameras.

YOLOv7 surpasses all the other well-known models in terms of speed and accuracy. This model has speed in range of 5 FPS to 160 FPS and has highest accuracy of 56.8% which is way ahead than transformer based detector SWIN-L Cascade-Mask R-CNN model which has 9.2 FPS and 53.9% AP. YOLOv7 model outperforms over its previous versions. In comparison to YOLOv4 model, YOLOv7 reduces the number of parameters by 75%, requires 36% less computation, and achieves 1.5% higher AP. So, after deployment of this model on real time traffic it will give the best results than conventional used traffic monitoring methods. These models will reduce the government cost to give salaries to people for monitoring also efficiency of work will increase as it works for 24*7 on road.

Future Scope:

- Nowadays all government highway and expressway contacts are given through bidding mode. if that private contractor takes the charge of complete monitoring of that highway using this VIDS system, then it will benefit the contractor to win the bid. Also, it will help common people from their safety point of view.
- Governments can use this model on existing highways or expressways to smartly monitor the huge and continuously increasing traffic.

- This will be a huge step toward the mission of digital India. This system will make an impact on the lives of crores of Indians through digital transformation.
- This model will increase the safety and security measures of travel in India. Which will increase India's ranking in road safety index.
- VIDS system will reduce the manpower needed to monitor the highways. 120 to 150 persons are required to collect tax at toll booths only. Same case for the traffic police, this system may not eliminate the role of the traffic police but may reduce the number of policemen required to patrol the highways.
- This entire system will reduce the time of common people to stand in long queues at toll booths.
- Suppose any vehicle is found missing on the highway then using camera feeds police can find the last VIDS camera on which that vehicle was detected for the last time, so police can find the missing vehicle near to last VIDS camera location.
- This system will auto-monitor the VIP car by tracking the time. If the car does not reach from one VIDS system to the other within a certain time, then the security personnel can take appropriate action.
- India has successfully implemented FastTag, UPI, and many more schemes which help to digitize the country. So, if this system gets implemented successfully on highways and expressways then India will have a huge influence in the world.

References:

1. Wang, C.Y., Bochkovskiy, A. and Liao, H.Y.M., 2022. YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors. *arXiv preprint arXiv:2207.02696*.
2. Wojke, N., Bewley, A. and Paulus, D., 2017, September. Simple online and realtime tracking with a deep association metric. In *2017 IEEE international conference on image processing (ICIP)* (pp. 3645-3649). IEEE.
3. Bochkovskiy, A., Wang, C.Y. and Liao, H.Y.M., 2020. Yolov4: Optimal speed and accuracy of object detection. *arXiv preprint arXiv:2004.10934*.
4. Lin, T.Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P. and Zitnick, C.L., 2014, September. Microsoft coco: Common objects in context. In *European conference on computer vision* (pp. 740-755). Springer, Cham.
5. Paul, Shantanu & Aradhya, Guru & Anand, Shreya & Lakshmi, K.s. (2021). Effectiveness of Mandating the FASTag in National Highways & Cross Border Toll Collection Centres.
6. C. Bari, A. Kumawat and A. Dhamaniya, "Effectiveness of FASTag System for Toll Payment in India," 2021 7th International Conference on Models and Technologies for Intelligent Transportation Systems (MT-ITS), 2021, pp. 1-6, doi: 10.1109/MT-ITS49943.2021.9529291.
7. Prevedouros, P. D., Ji, X. (Jerry), Papandreou, K., Kopelias, P., & Vegiri, V. (2006). Video Incident Detection Tests in Freeway Tunnels. Transportation Research Record, 1959(1), 130–139.
8. Kastrinaki, Vasiliki & Zervakis, Michalis & Kalaitzakis, Kostas. (2003). A survey of video processing techniques for traffic applications. Image and Vision Computing. 21. 359-381. 10.1016/S0262-8856(03)00004-0.
9. Yonten Jamtsho, Panomkhawn Riyamongkol, Rattapoom Waranusast. Real-time license plate detection for non-helmeted motorcyclist using YOLO.
10. P. Prabhakar, P. Anupama and S. R. Resmi, "Automatic vehicle number plate detection and recognition," 2014 International Conference on Control, Instrumentation, Communication and Computational Technologies (ICCICCT), 2014, pp. 185-190, doi:10.1109/ICCICCT.2014.699 2954.