Computer Vision based Vehicle Detection and Speed Estimation for Road Safety

Mohammed Ibrahim, B S Divya and Stephan Thompson

November 28, 2021

Computer Vision based Vehicle Detection and Speed Estimation for Road Safety

Mohammed Ibrahim
Department of computer science and Engineering
M S Ramaiah University of Applied Sciences
Bengaluru, India
miayaz26@gmail.com

Divya BS
Department of computer science and Engineering
M S Ramaiah University of Applied Sciences
Bengaluru, India
divyabies@gmail.com

Thompson S
Department of computer science and Engineering
M S Ramaiah University of Applied Sciences
Bengaluru, India
thompsoncse@gmail.com

***Abstract*— Road safety is of immense significance, as travel is a basic necessity. According to data published by World Health organization (WHO), nearly 1.3 million people loses their lives due to accidents and road crashes. This research work focuses on deploying object detection and tracking algorithms to differentiate over-speeding vehicles in the traffic using Machine Learning concepts such as Computer Vision and Deep Learning. A MobileNet SSD architecture trained on MS COCO dataset for vehicle detection and identification was adopted. Subsequently, detected vehicles were tracked with respect to their centroids and with the help of simple velocity equation, speed of the particular vehicle was registered. Moreover, the vehicles over-speeding would be captured along with their timestamp. The captured images could be shared over file hosting platforms such as dropbox, google drive, etc. authorities could make use of this system to bring orders on roadways through imposing fines on the offenders. Apparently, it can reduce accidents and road crashes.**

Keywords*— *Object Detection, Deep Learning, Computer Vision, Centroid Tracking

I. Introduction

Road safety refers to the procedures and techniques employed by road transport authorities to avoid road accidents and crashes to a maximum extent. According to the data obtained from World Health Organization (WHO), more than 1.3 million people die due to mishaps on road globally. Every 25 seconds a person loses a life on road and the condition is worse in developing countries such as India. According to a statement by Nitin Gadkari, the Honorable Union Minister of India, "the scenario of road accidents in India is far more serious than Covid- 19, registering around 415 deaths per day".

While there is a numerous reason for vehicle crashes, over-speeding constitutes to a major proportion. Speed of the vehicle is directly proportional to the likelihood of crash. With every 1% increase in the vehicle speed brings about 4% increase in fatal crash possibility and 3% increase in serious vehicle damage.

With the help of this research, one can impose fines on the traffic rules violators. Which can not only reduce the vehicle crashes, but also, we can impart a sense of responsibility on road users. This research is mainly developed with Deep Learning, Image Processing and neural network framework and techniques, that gives us an opportunity to explore, learn and make productive use of such areas. The deployment of which was explained elaborately in the upcoming sections.

A. Motivation

Each and every individual has the right to live and thrive their way into this world. These basic rights of human beings are being violated by the inadequate law enforcements of traffic rules and regulations. According to World Health Organization (WHO), on an average, road accidents cause harm to more than 3% of the Gross Domestic Product (GDP). Thus, the road crashes are directly related to the economic status of a country. We can strive to improve our economy considerably by reducing the road disasters.

We can easily grasp the increase in the trends of the deaths over the years with the help of various surveys. And if the situation persists, the death rate can jump up to 2,60,000 deaths per year. All these surveys and data indicates that some robust steps to be taken towards the road safety sector for the wellbeing of its citizens. This turns out to be the motivation for this research.

B. Related Work

There are breakthroughs that have been acquired in this similar field of work hitherto and primarily it is achieved through various uses of sensors, like VASCAR (Visual Average Speed Compute and Recorder), RADAR (Radio Detection and Ranging), LIDAR (Light Detection and Ranging), Speedometers, Tachometers and many more.

Although, in terms of result accuracy, they work adequate, but it is the cost of such assets that make them not only inaccessible, but impractical to deploy at most of the Roadways and Highways.

Visual Average Speed Computer and Recorder (VASCAR) is a gadget used to estimate the speed of the vehicles on the road. The VASCAR covers only certain area of the road where it is installed. It does not depend on any RADAR or LIDAR for the estimation of the speeds. The working of VASCAR is quite accurate, whose accuracy ranges from plus to minus 1%. But it requires a lot of manual approaches to detect the speed. First of all, a distance between two points for example, poles, trees, or any physical things are calculated manually. Secondary a person needs to be operating the VASCAR. Whenever a vehicle enters the first point, a button needs to be pressed and when the vehicle exits through the last point, the timer has to be stopped manually. This method needs a lot of human intervention, which can be prone to errors. The working of the VASCAR is based on the same velocity equation i.e., Speed = Distance/ Time There are breakthroughs that have been acquired in this similar field of work hitherto and primarily it is achieved through various uses of sensors, like VASCAR (Visual Average Speed Compute and Recorder), RADAR (Radio Detection and Ranging), LIDAR (Light Detection and Ranging), Speedometers, Tachometers and many more. Although, in terms of result accuracy, they work adequate, but it is the cost of such assets that make them not only inaccessible, but impractical to deploy at most of the Roadways and Highways.

Visual Average Speed Computer and Recorder (VASCAR) is a gadget used to estimate the speed of the vehicles on the road. The VASCAR covers only certain area of the road where it is installed. It does not depend on any RADAR or LIDAR for the estimation of the speeds. The working of VASCAR is quite accurate, whose accuracy ranges from plus to minus 1%. But it requires a lot of manual approaches to detect the speed. First of all, a distance between two points for example, poles, trees, or any physical things are calculated manually. Secondary a person needs to be operating the VASCAR. Whenever a vehicle enters the first point, a button needs to be pressed and when the vehicle exits through the last point, the timer has to be stopped manually. This method needs a lot of human intervention, which can be prone to errors. The working of the VASCAR is based on the same velocity equation i.e., Speed = Distance/ Time.

C. Constraints

Following are the constraints of this research:

- No dataset available for two wheelers and three wheelers which makes their detection arduous.
- Vehicles over-taking other vehicles has a potential of swapping their ID's
- Two overlapping vehicles could be detected as a single vehicle.
- The camera needs to installed very accurately and a small tilt in the camera angle could produce inaccurate speeds.
- The vehicles on the other side of the road should not be detected, which can provide disrupted readings.
- If this model is to be installed on a remote camera, a Raspberry pi and intel movidius neural compute stick (NCS) is required for fast computation.

II. LITERATURE SURVEY

An approach to create vehicle speed detection system from a video scene was conducted in 2009 to estimate the speeds of the vehicles travelling on the road [1][2]. Their work was successful in estimating speeds but due to low effectiveness and low performance of the system it could not be employed in real time. And it requires a lot of time for computing. The system lacked in efficiency whereas the approach can be used to create a more robust system.

An Efficient Convolutional Neural Networks for Mobile Vision Applications i.e., the mobile Nets was utilized [3]. Mobile Nets are based on a streamlined architecture that uses depth wise separable convolutions to build light weight deep neural networks. Smaller and faster Mobile Nets using width multiplier and resolution multiplier by trading off a reasonable amount of accuracy to reduce size and latency. With the help of Light weight deep neural networks one can use it for wide range applications for object detection [4].

The combination of MobileNet and SSDs using Caffe Version of the original Tensorflow implementation by training on COCO dataset using OpenCV 3.3 version was explained [5]. This deep learning model uses (MobileNet + SSD), and can detect objects in approximately 6-8 FPS [6]. Further speed improvements can be obtained by applying skip frames, using different variations of MobileNet and using the quantized version of Squeeze Net. The fact that the author had provided stored as well as real time analysis of the images in the algorithm was mentionable.

The speed tracking of the vehicles was explained [7]. They tracked the vehicles using edge detection. But Speed calculated was not very accurate. Further work was required in image processing domain. And not applicable for real time purpose due to more image processing techniques. This method included Noise removal, background subtraction, morphological operations add up to a lot of computation. Thus, real time implementation isn't possible.

A Region Based Framework Regression/classification-based Framework was discussed [8]. We can employ this study to improve localization accuracy on small objects under partial occlusions, it is necessary to modify network architectures. This study has been meaningful for the developments in neural networks and related learning systems, which provides valuable insights and guidelines for future progress. This Provides a detailed review on deep learning-based object detection frameworks which handle different sub-problems, such as occlusion, clutter and low resolution, with different degrees of modifications on R-CNN.

Multi-Scale feature maps, Convolution predictors, Default boxes and aspect ratios was used [9]. 300 × 300 input, SSD achieves 74.3% mAP on VOC2007 at 59 FPS on a Nvidia Titan X and for 512 × 512 input, SSD achieves 76.9% mAP, outperforming a comparable state-of-the-art Faster R-CNN model. Experiments and demonstrations prove that given appropriate training strategies, a larger number of carefully chosen default bounding boxes results in improved performance. Given the same VGG-16 base architecture, SSD compares favorably to its object detector counterparts in terms of both accuracy and speed. SSD model provides a useful building block for larger systems that employ an object detection component.

The main methodology used was bounding box and classes prediction [10]. At 320 × 320 YOLOv3 runs in 22 ms at 28.2 mAP, as accurate as SSD but three times faster. It also achieves 57.9 AP50 in 51 ms on a Titan X, compared to 57.5 AP50 in 198 ms by RetinaNet, similar performance but 3.8× faster. In conclusion the authors mentions that computer vision is already being put to questionable use and as researchers we have a responsibility to at least consider the harm our work might be doing. It calculates the Anchor box offset predictions Linear (x,y) predictions instead of logistic. It is possible that some of these techniques could eventually produce good results, perhaps they just need some tuning to stabilize the training.

Real time vehicle tracking with speed estimation was discussed [11]. The work included Detecting vehicles with MobileNet SSD and applying speed estimation concurrently. Although a Powerful system for detection further engineering is required because of the limitations. In this work Swapping of object id's when vehicles change direction was a big problem. And it Requires manual calibration. An overall efficient system for real time application. Requires comparatively less computation power.

III. Study of Datasets and Architectures

Datasets play an important role in deep learning algorithms. In fact, the efficiency of the proposed architecture completely relies on the dataset used for training. Architecture is the conceptual model of desired functionality.

This section focuses on the architecture design of the research, the sequential application of the features, flow of coding syntax and the Hardware and Software requirements along with their versions.

A. Proposed Architecture

This section can also be referred as Sequence Design. The major segments of the research can be apportioned into 3 phases, hence creating a sequential plan for befitting research configuration, analysis, working and performance. Fig 1 describes the overall concept.

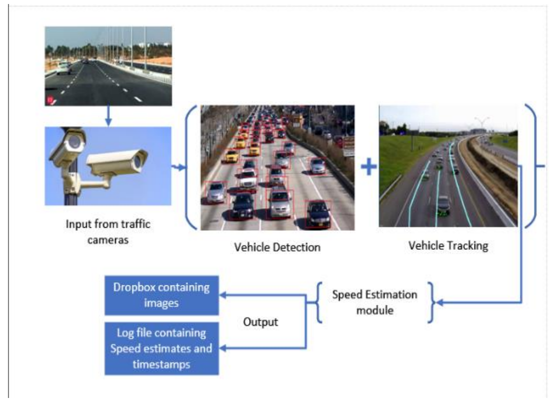

Fig. 1. Phase1: Vehicle Detection. Phase2: Vehicle Tracking. Phase3: Speed Estimation. The above Fig portrays the entire High Level Design of the project. The inputs were obtained from the traffic cameras. This input was then processed with image processing techniques such as image smoothing, scaling, noise reduction. Then the inputs were analyzed and detected for any vehicles. The detected vehicles were tracked and their speeds were calculated. If the vehicle violates the maximum speed limit their image was captured and saved into Dropbox. The log file contains the other details such as the actual speed of the vehicle, specific date and time, and the image ID and ID of the Vehicle.

B. Vehicle Detection Algorithm

To detect surrounding objects accurately, several existing detection systems which could classify objects and evaluate it at various locations in an image were investigated. RCNN uses region proposal methods to generate possible bounding boxes in an image. Then, various ConvNets are applied to classify each box. The post processed results give finer output boxes. The slow testing-time, complex training pipeline and the large storage does not fit into our application. So, the models which are compatible with resource constrained devices and have a tiny architecture are desired. From the literature it was found that MobileNet-SSD was the more suitable model for this research.

- MobileNet SSD trained on COCO dataset.

Basically, MobileNet works as a feature extracting base network for (Single-Shot multi box Detection) SSD, combinedly MobileNet-SSD gives the feature extraction and image classification results necessary for object detection as shown below.

The MS COCO (Microsoft Common Objects in Context) dataset is a large-scale object detection, segmentation, key-point detection, and captioning dataset. The dataset consists of 328K images. The COCO Dataset has 80 classes. In this research, the pretrained MobileNet SSD constitutes of 21 Classes, as this research requires the classes which have only Automobiles and Vehicles in them.

— Classes that contain Automobiles and Vehicles like

1. Bus
2. Car
3. Motorbike

The designs or diagrams in this section will focus on the coding syntax and primary python packages required for executing the program using Python and OpenCV. This section digs deeper into the phasing design of the research and emphasizes on the flow of code with the help of low-level diagrams as shown as in Fig 2.

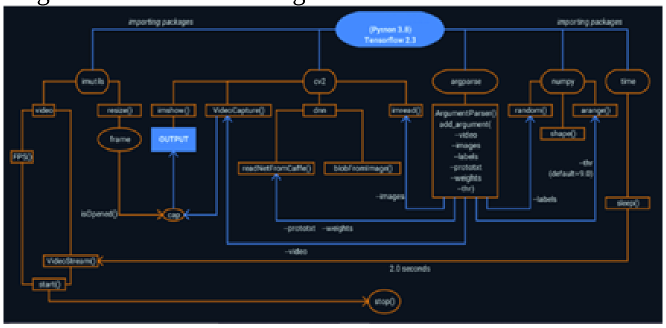

Fig. 2. The flowchart representing the flow of code implemented for the real-time Cars detection executed on PyCharm 2012.2 (IPython 3.8)

The above diagram depicts the pipeline employed to obtain vehicle detection with the help of MobileNet SSD Architecture trained on MS COCO dataset. The Threshold parameter needs to be adjusted manually for confidence. The MobileNet Architecture can detect multiple objects in a single shot. That's why the name SSD (Single Shot Object Detection). MobileNet SSD is based on a Feed-Forward convolutional neural network.

C. Vehicle tracking Algorithm

Vehicle tracking is a utilization of object tracking where moving objects are situated inside video data. Thus, vehicle tracking algorithms can handle live, real-time film and furthermore pre-recorded video documents.

The vehicle tracking algorithm is called centroid tracking as it depends on the Euclidean distance between (1) existing vehicle centroid (i.e., vehicle the centroid tracker has effectively seen previously) and (2) new centroids for the same vehicle between ensuing casings in a video.

When a vehicle is detected by above mentioned vehicle detection algorithm, the coordinates of the bounding boxes produced by detections were forwarded to the tracking algorithm. The tracking algorithm calculates the centroids of the bounding boxes by determining the diagonals.

When the object moves in any direction, coordinates of the bounding boxes were updated. The algorithm keeps constantly checking whether the centroids were at exact center. Else it w update the location of the centroid. This process keeps repeating for every detected vehicle.

Steps involved in a vehicle tracking algorithm*:*

Step 1: Accepting the input bounding boxes coordinates and computing their centroids with the help of bounding box diagonals.

Step 2: When the new bounding boxes are obtained for the same vehicle, we will compute the Euclidean distance between newly obtained bounding boxes and the existing ones for the same vehicle.

Step 3: Updating the bounding box coordinates (x, y) of the moving vehicle

Step 4: Registering new objects with the help of their Bounding boxes

Step 5: Deregistering of the older objects along with their IDs when they disappear from the frame.

D. Speed estimation

Each centroid represents a vehicle. One can easily track the movement of the vehicles by their centroids. The camera was installed on one side of a road or highway which will cover one complete side of the road, which means the camera was installed at such an angle that all incoming vehicles can be detected and tracked.

For speed estimation was carried out using the centroids. Four imaginary lines are created along the width of the input video frame as shown in the Fig 3 when the centroids meet the first line a timestamp of the centroid is registered. The same process is carried out for all the four imaginary lines. Each timestamp is registered which will help in speed estimation. And we have to manually calculate the distance covered by those imaginary lines. This process needs to be done on-site.

Now we have two main variables for the estimation of speed of vehicles i.e., Time and Distance. The basic formula for calculating speed is **Speed = Distance / Time**. We can easily calculate the speed with the two variables obtained. Here distance was measured in terms of pixels as shown in Fig 5 (from A to B was 360 pixels, B to C was 420, C to D was 480 and D to end was 540, which was corresponding to equidistance 80m) and time was measured in terms of number of frames in the video. Here it was 8-10 Frames Per Second. The online video mentioned in [12] was used for carrying out trials and evaluating the model performance.

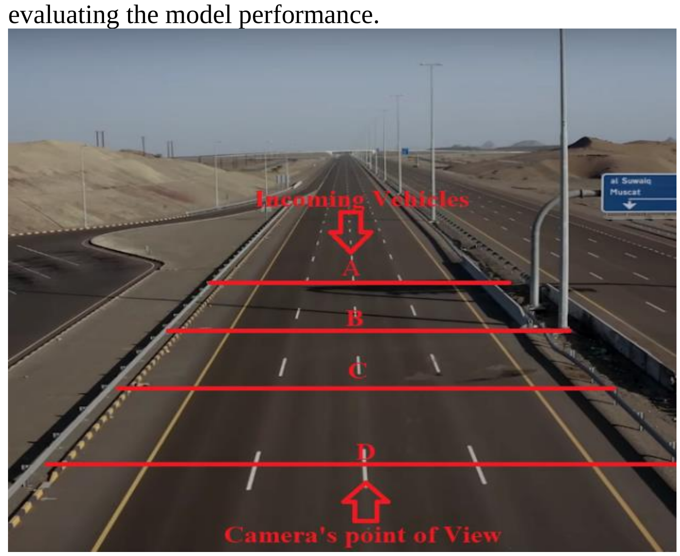

Fig. 3. Fig showing the imaginary lines for the calculation of speed. user viewing point is the camera's point of view and the vehicles approach from opposite side.

IV. IMPLEMENTATION

Implementation of all the above-mentioned modules integrated together, is carried out by running the code in the PyCharm

(Version 2012.2). The complete implementation of the code is explained in this section.

TABLE 1. Specifications of hardware requirements

Hardware Requirements	
Operating system	Windows, Ubuntu/Raspbian
CPU	Intel Core i3/i5, (or AMD Ryzen 3/5)
System type	64-Bit
RAM	8 GB
SD Card	4 GB
Camera Resolution	>10Mega pixels and minimum 30FPS
Processing unit for Raspberry Pi	Intel Movidius Neural Compute Stick (NCS)

To run the code successfully on a local hardware one need to be sure of the hardware requirements which were mentioned above. We can execute the code on any python terminals or any IDEs (interactive development environments). We have to be sure that the python version is higher than 3.0 and all the modules or libraries mentioned before were successfully installed.

V. RESULTS AND DISCUSSIONS

The protype software can detect, Track and estimate the speed of the vehicles travelling on the road. Through which law and order could be imposed on road and minimize road accidents due to over-speeding. With the help of this model, one can penalize the offenders as their images were captured. This project had the potential to be far more economical than the existing ways of penalizing the speed mongers. Without any doubt this can reduce the accidents. Fig 4 shows the results of vehicle detected and ID assigned.

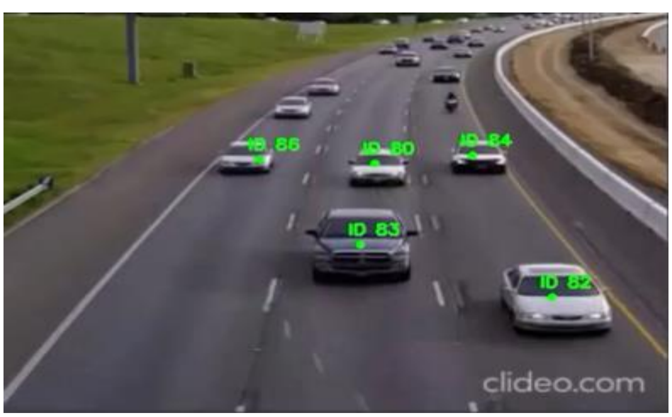

Fig. 4. Fig depicts the results obtained from applying Centroid tracking algorithm on the vehicles detected from MobileNet SSD trained on MS COCO Dataset. vehicles were tracked successfully and each vehicle is assigned a unique identification number ID's

The Centroid tracking algorithm would track the vehicles detected with the help of MobileNet SSD trained on MS COCO Dataset. When the vehicles were detected, they were assigned distinct IDs for their identification. Vehicles were tracked with respect to their IDs. When a bounding box was created around the detected vehicle, its centroid was calculated by using their diagonals. The vehicles would have the same ID until they were in the frame. When the vehicle leaves the frame or the vehicle disappears their distinct ID was deregistered. When a new vehicle was detected, the algorithm would assign a new ID to that vehicle in the increasing number order.

The Frames per second (FPS) was around 8-10 FPS. If the vehicles stay in frames for more frames, this indicates the speed of the vehicle was less. Similarly, if vehicles stay in frames for less frames, this indicates the speed of the vehicle was more. So a constant frame per second (FPS) was very necessary to acquire more precise speeds. Fig 5 showed the output of the developed vehicle speed detecting software.

The speed and the ID of the vehicles were displayed in the terminal. Speed of each vehicle would be calculated and printed in the python terminal. The maximum speed of the vehicles was set to 65KMPH, from now on the vehicles exceeding the speed limit would be noted, captured and saved in local drive or dropbox (If Dropbox account was linked).

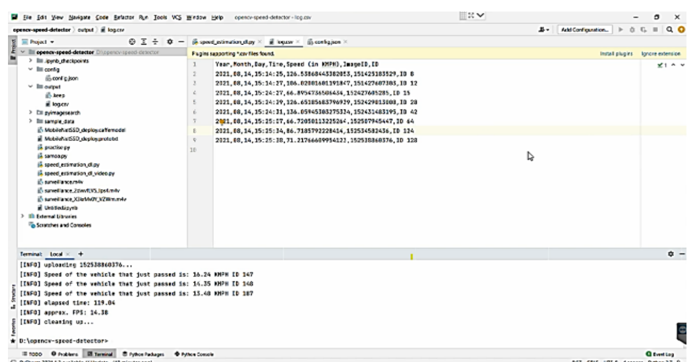

Fig. 5. The output showing the speed and ID of the vehicle

Moreover, the details of the vehicles will be stored in a log file. This log file was created automatically if it doesn't exist. The log file will be stored locally on the disk. This log file would have more details like Date, Time, Speed of the Vehicles in KMPH, Image ID, vehicle ID.

VI. ALPHA-BETA TESTING

The final integrated code was deployed on a local machine with i5 11th generation processor and Nvidia GeForce GTX 30 graphical processing unit. When the script was executed, the speed estimation model was loaded. A new window pops up displaying the vehicles being detected and tracked. The testing has been carried on the video mentioned in the sample data. The IDE (Interactive Development Environment) used was PyCharm. As one could see in the video the cars were being detected and tracked and their speeds would be displayed in the python terminal. Vehicles are given distinct IDs and they were displayed on the screen. The speed limit while testing was set to 35 KMPH. Any vehicle going beyond the speed limit would be tracked and captured. This data was accumulated in the local directories and on dropbox.

To test the model for various weather conditions such as sunny, cloudy and rainy the video clip has been edited in Cyber Link Power Director 365 as shown in Fig 6, Fig 7 and Fig 8. Three different videos would be tested and the results obtained were as mentioned below.

Fig. 6. Sunny/clear weather

Fig. 7. Rainy Weather

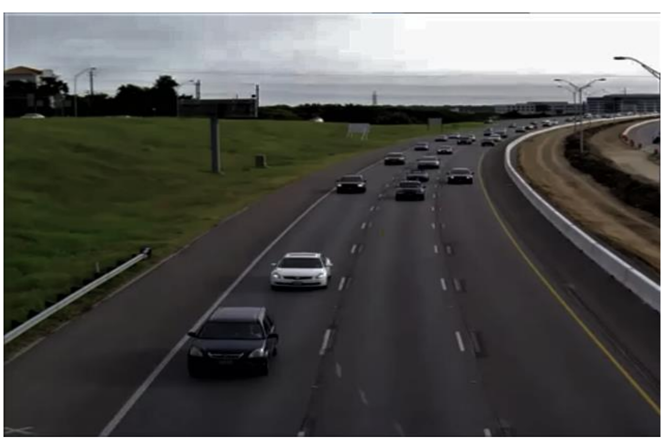

Fig. 8. Cloudy/Overcast weather

TABLE 2. Detection and Tracking acccuracy of the concept in different weather condition

Weather	Total Number of Vehicles	Total vehicles Detected	Detection Accuracy in %	Tracking Accuracy in %
Clear/Sunny	104	102	98.07	98
Rainy	104	89	85.57	85
Cloudy/overcast	104	95	91.34	92

VII. CONCLUSION

This research has been carried out in order to cope up with the need of hour. This model can differentiate Vehicles which are over-speeding from the normal traffic. These vehicles are captured and stored as evidence which be used by authorities governing roads. By this plan, we can reduce the over-speeding vehicles by scaring them off with hefty fines. By deploying this work, we can reduce the Vehicle crashes on road. Therefore, saving a chunk of the national GDP (Gross Domestic Product) which dips up to 3% due to vehicles crashes every year. Through this work, I've tried to demonstrate the possibility of using computer vision techniques in public transport systems.

REFERENCES

[1] Dhulavvagol, P. M., Desai, A. & Ganiger, R., 2017. Vehical Tracking and Speed Estimation of Moving Vehicles for Traffic Surveillance Applications. International Conference on Current Trends in Computer, Electrical, Electronics and Communication (CTCEEC).

[2] V. K. Madasu and M. Hanmandlu, 2010, Estimation of vehicle speed by motion tracking on image sequences, IEEE Intelligent Vehicles Symposium, 2010, pp. 185-190, doi: 10.1109/IVS.2010.5548051.

[3] Gupte, S., Masoud, O., Martin, R. F. K. & Papanikolopoulos, N. P., 2002. Detection and Classification of Vehicles. *IEEE Transactions on IntelligentTransportation Systems,* Volume 3, pp. 30-43.

[4] S. Hua, M. Kapoor and D. C. Anastasiu, "Vehicle Tracking and Speed Estimation from Traffic Videos," *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2018, pp. 153-1537, doi: 10.1109/CVPRW.2018.00028.

[5] Howard, A. G. et al., 2017. MobileNets: Efficient Convolutional Neural Networks for Mobile VisionApplications. *arXiv:1704.04861v1 [cs.CV].*

[6] S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In Advances in neural information processing systems, pages 91–99, 2015.

[7] Kumar, T. & Kushwaha, D. S., 2016. An Efficient Approach for Detection and SpeedEstimation of Moving Vehicles. *Twelfth International Multi-Conference on Information Processing-2016 (IMCIP-2016).*

[8] Kumar, T., Sachan, R. & D. S. Kushwaha, 2015. Smart City Traffic Management and Surveillance System for Indian Scenario. *Proceedings ofInternational Conference on Recent Advances in Mathematics, Statistics and Computer Science (ICRAMSCS).*

[9] Pornpanomchai, C. & Kongkittisan, K., 2009. Vehicle Speed Detection System. *2009 IEEE International Conference on Signal and Image Processing Applications,* pp. 4-5.

[10] Redmon, J. & Farhadi, A., 2018. YOLOv3: An Incremental Improvement. *arXiv:1804.02767 [cs.CV].*

[11] Wei Liu, D. A. D. E. et al., 2016. SSD: Single Shot MultiBox Detector. *arXiv:1512.02325 [cs.CV],* pp. 3-4.

[12] Supercircuits (2014) *Alibi ALI-IPU3030RV IP Camera Highway Surveillance* [Online Video] Available at: https://www.youtube.com/watch?v=_channel=Supercircuits