Traffic Rule Violation Detection

Abha Wadjikar
Department of Computer science and engineering
Student
National Institute of Technology Karnataka
abhawadjikar.171co102@nitk.edu.in

Niwedita
Department of Computer science and engineering
Student
National Institute of Technology Karnataka
niwedita.171co227@nitk.edu.in

Sharmila Biswas
Department of Computer science and engineering
Student
National Institute of Technology Karnataka
sharmila.171co143@nitk.edu.in

ABSTRACT
Traffic violation detection systems are effective tools to help traffic administration to monitor the traffic condition. It can detect traffic violations, such as running red lights, over speeding, and vehicle retrogress in real-time. In the paper, we propose to automate the traffic rules violation detection system and make it easy to monitor the traffic and take action against the offender in a fast and efficient way. A software is proposed using TensorFlow with an SSD object detection model to detect cars that violate a set of predefined rules with the help of efficient algorithms. Each vehicle can be tracked across a surveillance video and useful vehicle information can be extracted. The vehicles that violate the speed limit are detected and notified. In this project, an intelligent control system is capable of tracking all vehicles, traffic guidance, and recording driving offenses along the roadway.

Keywords
Tensorflow, OpenALPR, SSD object detection, license plate, vehicle-monitoring.

1. INTRODUCTION
The increase in the number of vehicles in today's time is serving a reason for the increase in traffic rule violations and various crimes associated with it. The traffic rule violation leads to an increased risk of fatal road accidents, therefore, it is important to have a good infrastructure for this purpose. An automated traffic violation detection system can effectively monitor the traffic, and capture violations such that law enforcement can be applied quicker. This helps to reduce illegal driving so as to ensure smooth traffic flow.

Certain automated solutions were developed to eliminate the violations; however, each of them has certain limitations. For example, the video capturing cameras eliminated the need for an authority to be present to check rule violations. However, the video footage has to be checked manually to survey the road for any violation. In this proposed system, a solution for signal breaking violation is given. Tensorflow object detection API will capture the image from real-time videos and detect the traffic rule violation. Number plate recognition is done using OpenALPR APIs which will recognize the characters on the License plate.

2. RESEARCH
Road accidents in India claimed over 1.5 lakh lives in the country in 2019. Overspeeding has been identified as the foremost reason behind the maximum number of road-related deaths reported in India between January 1, 2018, and March 31, 2019. Keeping this in mind, we planned to provide a solution to identify the violators as quickly as possible by using accurate object detection models and recognition methods so that they can be notified of their offense.

There are several softwares that exist for traffic detection like the RFID Technology for smart Vehicle control using Traffic signal speed limit

tag communication [1]. However, this system has a lot of infrastructure dependencies to be able to fulfill its purpose.

Currently, the government is using the challan system to decrease the traffic violation. A CCTV camera continuously records footage of the ongoing traffic. If any vehicle breaks any traffic rule, the act is recorded in the footage and the police try to extract the number from the vehicle's screen-shot captured from the CCTV footage and the offense will be registered in the records. But several complaints pertaining to challan being sent to persons who do not own the vehicles hauled up for violating traffic rules. There is a possibility of fraud as it's a manual system. Therefore an automated system is required.

Most of the strategies and techniques used in automated systems have heavy limitations. One of the limitations is due to low computational resources at the user level. Other important limitations that need to be tackled are lack of proper data analysis of the measured trained data, dependency on the motion of the objects, inability to differentiate one object from another, and also concern over the speed of the object under detection and Illuminacy. Also, a consistent intelligent system requires consistent components such as intelligent software for crisis management, reporting, and training the system for different conditions. Therefore, the genetic algorithm plays a significant role in the path of our project.

There are several object detection methods, one of these is YOLOv3 (YOLO version 3). YOLO stands for You Only Look Once. It's an object detector that uses features learned by a deep convolutional neural network to detect an object. Being an FCN, YOLO is invariant to the size of the input image. One of the biggest problems is that if we want to process our images in batches (images in batches can be processed in parallel by the GPU, leading to speed boosts), we need to have all images of fixed height and width so that we can concatenate multiple images into a large batch.

Once the object is detected, we need a reliable method to detect the license plate from the vehicle so that we can extract useful information linked to the license plate about the user. Using precise software to implement detection and recognition of the characters of the license plates on vehicles is extremely important.

There are several software used to detect license plates from vehicles. The most popularly used one is ANPR. ANPR stands for Automatic Number Plate Recognition and is a term used to describe the software that takes input from an ALPR camera and recognizes the alphanumeric characters on a plate using the software, and then this analyzed data gained through the software can be stored in video storage or can be used to match it up against database records.

In reality, there is no big difference between a License Plate Recognition and Automatic Number Plate Recognition. In fact, ALPR and ANPR are two different terms used to describe the hardware and software used for license plate recognition and capturing those number plates using surveillance camera systems.

3. SSD OBJECT DETECTION

3.1. Model

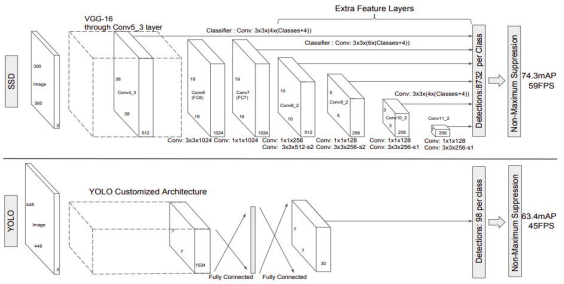

Fig 1[10]

SSD is based on a feed-forward convolutional network that produces a fixed-size collection of

bounding boxes and scores for the presence of object class instances in those boxes, followed by a non-maximum suppression step to produce the final detections. The early network layers are based on a standard architecture used for high-quality image classification (truncated before any classification layers), which we will call the base network2. We then add auxiliary structure to the network to produce detections with the following key features:

Multi-scale feature maps for detection We add convolutional feature layers to the end of the truncated base network. These layers decrease in size progressively and allow predictions of detections at multiple scales.

Default boxes and aspect ratios We associate a set of default bounding boxes with each feature map cell, for multiple feature maps at the top of the network. Specifically, for each box out of k at a given location, we compute c class scores and the 4 offsets relative to the original default box shape. This results in a total of (c + 4)k filters that are applied around each location in the feature map, yielding (c + 4) kmn outputs for an m × n feature map. Our default boxes are similar to the anchor boxes used in Faster R-CNN [9], however, we apply them to several feature maps of different resolutions.

3.2. Training

During training, we need to determine which default boxes correspond to ground truth detection and train the network accordingly. For each ground truth box, we are selecting from default boxes that vary over a location, aspect ratio, and scale.

Let x pij = {1, 0} be an indicator for matching the i-th default box to the

P j-th ground truth box of category p.

In the matching strategy above, we can have i x pij ≥ 1. The overall objective loss function is a weighted sum of the localization loss (loc) and the confidence loss (conf):

$$L(x,c,l,g) = 1/N \left(L_{conf}(x,c) + \alpha L_{loc}(x,l,g) \right) \qquad (1)$$

where N is the number of matched default boxes.

If N = 0, wet set the loss to 0. The localization loss is a Smooth L1 loss [6] between the predicted box (l) and the ground truth box (g) parameters. Similar to Faster R-CNN [2], we regress to offsets for the center (cx, cy) of the default bounding box (d) and for its width (w) and height (h).

$$L_{loc}(x,l,g) = \sum^{N}_{i \in Pos} \sum_{m \in \{cx,\, cy,\, w,h\}} x^{k}_{ij} smooth_{L1}(l^{m}_{i} = g^{m}_{j}) \qquad (2)$$

$$g_j^{cx} = (g_j^{cx} - g_i^{cx})/d_i^{w} \qquad g_j^{cy} = (g_j^{cy} - d_i^{cy})/d_i^{h}$$

$$g_j^{w} = log\,(g_j^{w}/d_i^{w}) \qquad g_j^{h} = log(g_j^{h}/d_i^{h})$$

3.3. Comparison results with YOLO

While comparing the above SSD model to the existing YOLO, we found that SSD (that uses multi-scale convolutional feature maps at the top of the network instead of fully connected layers as YOLO does) is faster and more accurate than YOLO.

SSD is simply relative to methods that require object proposals because it completely eliminates proposal generation and subsequent pixel or feature resampling stages and encapsulates all computation in a single network. This makes SSD easy to train and straightforward to integrate into systems that require a detection component. Experimental results on the PASCAL VOC, COCO, and ILSVRC datasets confirm that SSD has competitive accuracy to methods that utilize an additional object proposal step and is much faster while providing a unified framework for both training and inference. For 300 × 300 input, SSD achieves 74.3% mAP1 on the VOC2007 test at 59 FPS on an Nvidia Titan X and for 512 × 512 input, SSD achieves 76.9% mAP, outperforming a comparable state-of-the-art Faster R-CNN model. Compared to other single stage methods, SSD has much better accuracy even with a smaller input image size.

4. OPENALPR LICENCE PLATE DETECTION

In India, basically, there are two kinds of license-plates, black characters in the white plates and black characters in yellow plates, white characters on red-plates. The former for private vehicles and latter for commercial, public service vehicles. The system tries to address these two categories of plates. The image processing module makes use of the API called OpenALPR.

The OpenALPR works in two major steps: License plate detection and character recognition. For license plate detection, it uses OpenCV's cascade classifier[6] which needs to be trained with images of license plates to be detected. For character recognition, it uses Tesseract's OCR [7] which needs to be trained with characters on the license plates. Along with these two trained files, a configuration file is needed which contains the data regarding the size of a license plate and characters on it as per government rules. If the license plate and characters on it are as per government standards then the license plate is recognized, otherwise, no license plate is detected. The OpenCV's cascade classifier is trained with Indian license plates for detection purposes. For recognizing characters on the license plates, trained data of European license plates were used as European license plates are very similar to Indian license plates.

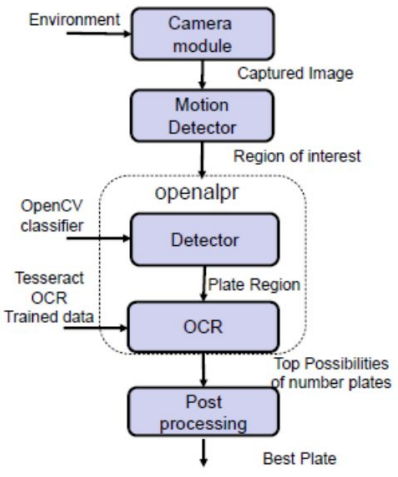

Fig 2[12]

4.1. Creating a license plate database

For the detection of license plates, more than 1000 images are required for achieving the desired accuracy required which is of the order of 98% or more. The database has been created with variations in capturing distance, angle, and light intensities. Total of 1300 images were collected with a variation. While obtaining images it was made sure that the license plate is readable by the human eye in that condition. The 4786 negative samples were obtained from an online database. The images contained roads, structures, living rooms, people, etc. anything other than a number plate. Each negative image should be equal to or greater than the desired training window [8].

4.2. Preparing training data

The number of negative images used is more than twice the positive images as good results were obtained by this ratio. We need to obtain the positive samples from the images captured by cropping only the license plates from the images. The opencv_createsamples[6] utility from the OpenCV library is used for preparing the positive samples. The controlled amount of random variation is introduced into the existing samples and more samples are created using this utility. The images are rotated around all three axes by 15 degrees and an intensity deviation of 50 was used. All the images are then resized to the width and height of 50 and 11 pixels respectively as the aspect ratio of the Indian license plate is 4.5. The spatial dimensions of samples are kept small (i.e. 50 x 11) so that training time is reduced, and the license plate of this small size could also be detected. This utility prepares all the positive samples with specified variations and packs them in a vector file which is then used for training of the classifier.

4.3. Training

Few parameters are to be optimized for obtaining the best results such as the type of features to be used, min hit rate and max false alarm rate at each stage, and a number of stages to be used.

i. Min Hit Rate:
If this parameter is set to 1, then each stage tends to train more rigorously for detecting all the positive samples. This increases the accuracy of training data but reduces accuracy over new samples. This parameter is chosen to be 0.95 which is a decent value.

ii. Max False Alarm Rate:
Overall false alarm rate is the measure of the accuracy of the classifier thus it should be close to zero, preferably in the order of 10-5[8]. Higher the value more lenient will be each stage and more negative samples would be misclassified as positive samples. Thus, to optimize between the time taken and accuracy, the value chosen is 0.5 per stage which is a moderate choice.

iii. Feature Type:
OpenCV provides two options for feature selection: HAAR-like features, LBP features[8]. The algorithm using HAAR-like features is very accurate but computationally expensive while the LBP algorithm is less accurate but very simple[6]. The features to be used must be selected wisely to optimize between speed and accuracy

iv. Number of stages:
The OpenCV's cascade classifier consists of a number of stages that are to be configured by the user depending on the complexity of the application. By using more stages, classifiers at initial stages are made less complex which helps in increasing detection speed by eliminating negative windows at the initial stage itself and spending more time on only promising windows.

After choosing the optimum values for all parameters, the classifier was trained with all the positive and negative images available. After training is completed, the cascade.xml file is generated which is used by OpenALPR for the detection of Indian license plates.

4.4. Comparison Results

In reality, there is no big difference between a License Plate Recognition and Automatic Number Plate Recognition. In fact, ALPR and ANPR are two different terms used to describe the hardware and software used for license plate recognition and capturing those number plates using surveillance camera systems.

5. WORKING

TensorFlow with an SSD object detection model is used to detect cars. From the detections in each frame, each vehicle can be tracked across a video and can be checked if it crossed the red light. The speed of the vehicle is calculated. The vehicles that violate the speed limit and cross the red light are detected and the license number is extracted using the OpenALPR for the identification purpose.

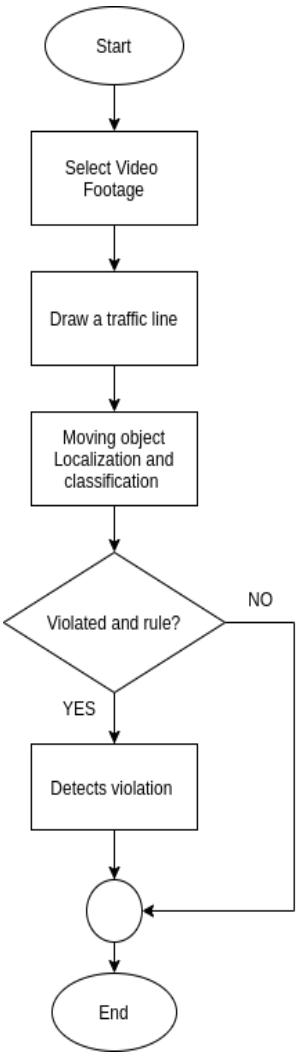

Fig. 3

5.1. Identify vehicles:

To identify the type of vehicle and create a frame around it.

```
def
matchVehicles(currentFrameVehicles,width,height
:
  define a max_distance to detect the frame;
  define coordinates of the baseframe;
   detect the vehicle and create a frame across
respect to
  baseframe;
  use a recursive method to find the minimum fr

for all the vehicle v:
       if(distance<max_distance):
          update the position of the v;
     else
         frame of the vehicle doesn't exist;
```

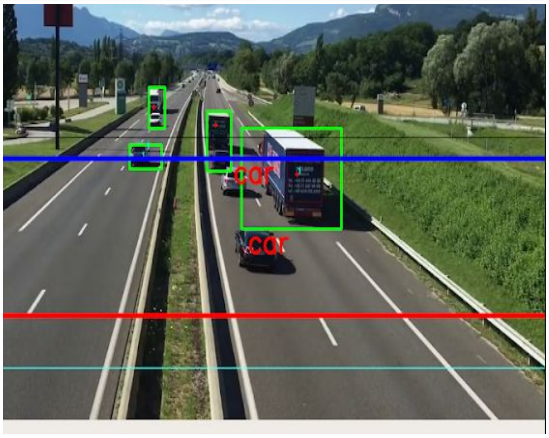

Fig 4

5.2. Check for red light:

This is a function made to check if the person has crossed the red light or not.

```
def checkRedLightCrossed(image):

define a crossing line with coordinates
x13,y13;
Initialize a variable count;
     for all the vehicles v:
if v has crossed:
   increment count;
   add the vehicle image in culprit list
   with timestamp;
```

5.3 Speed detection

This function helps to check the speed of the vehicle.

```
def checkSpeed(time,img):
        store exitTime and enterTime;
        speed=60/(v.exitTime-v.enterTime)
       print(speed)

Where v=vehicle
          if speed>60:
                    add the vehicle in the
violators list with the
           timestamp;
```

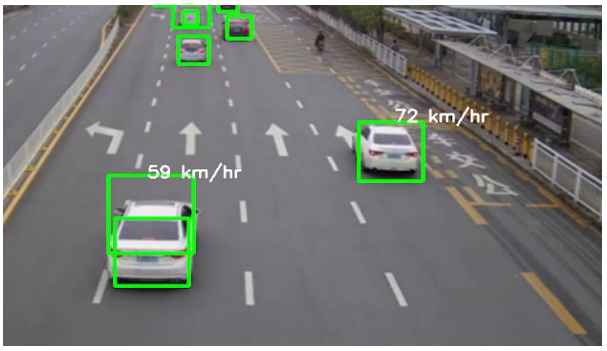

Fig 5

5.4. License Plate Detection:

The following is a basic flow of how the license plate is detected. We write a function called getLicensePlateNumber that gives the text detected from the license plate.

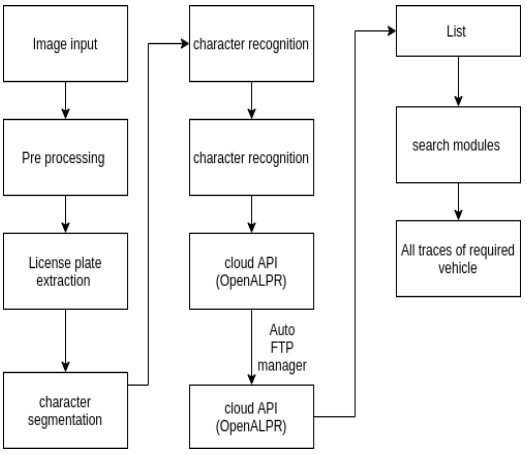

Fig 6

```
def getLicensePlateNumber(filer):

for all the recognized vehicle:
      crop out the detected license plate region from the image;
identify the edges/shape of the license plate;
transforms the perspective to straight-on view based on ideal license plate size;
Isolates and clean up the image so that the characters can be processed individually
Analyze each character and store it.
```

Fig 7

6. OUTPUT/RESULT

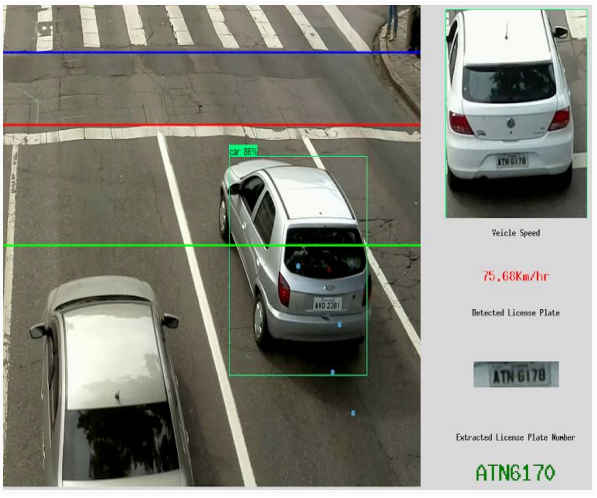

Fig 8

Violations like lane, red light as well as over speeding were tested separately using the same model. The system precision is calculated as the ratio of true detection and the summation of true as well as false detection. This system can be placed at a distance of 10-15m from the traffic signal to ensure an efficient result. Violation caused by each of the mentioned vehicles is analyzed separately. This system will eliminate the requirement of manpower and resources. The processing speed and accuracy of the system depends on various factors like the amount of traffic, the resolution of the camera, and the flexibility of the software.

7. CONCLUSION

The goal of the project is to automate the traffic signal violation detection system and make it easy for the traffic police department to monitor the traffic and take action against the violated vehicle owner in a fast and efficient way. Detecting and tracking the vehicle and their activities accurately is the main priority of the system.

8. ACKNOWLEDGMENT

We would like to express our sincere gratitude to our guide Prof. Shashidhar G Koolagudi for providing us the platform and suggestions throughout the project. We would also like to thank our project coordinator Dr. Bharthi for providing us the necessary guidance throughout the project.

9. FUTURE WORK

A message via SMS will be sent to the offender in case of a rule violation scenario, and the Regional Transport Office (RTO) will also be informed to track the report status.

The different features in this proposed system are:-

1)Reducing the workload of Traffic Police by automating the system.
2) Corruption Avoidance.
3) Automatic fine collector.

10. REFERENCES

[1] Imperial Journal of Interdisciplinary Research (IJIR) Vol-2, Issue-5, 2016 ISSN: 2454-1362, http://www.onlinejournal.in Violation Detection at Traffic Signals Using RFID System.

[2] Journal of Advanced Computing and Communication
Technologies (ISSN: 2347 - 2804) Volume No.

[3] Issue No. 3, June 2015 56 Traffic Detection System
Using Android.

[4]
https://github.com/openalpr/openalpr/wiki/OpenALPR-Design

[5]
http://www.slideshare.net/louiseantonio58/image-Processing-based-intelligent-traffic-Control-system Matlab-GUI

[6] Shengcai Liao, Xiangxin Zhu, Zhen Lei, Lun Zhang, and Stan Z. Li., "Learning Multi-scale Block Local Binary Patterns for Face Recognition", International Conference on Biometrics (ICB), 2007, pp. 828-837.

[7] Ray Smith, "An Overview of the Tesseract OCR Engine", In Google Inc.

[8] Paul Viola, Michael J. Jones, "Rapid Object Detection using a Boosted Cascade of Simple Features", In Conference on Computer Vision and Pattern Recognition (CVPR), 2001, pp. 511-51

[9] Ren, S., He, K., Girshick, R., Sun, J.: Faster R-CNN: Towards real-time object detection with region proposal networks. In: NIPS. (2015)

[10] https://arxiv.org/pdf/1512.02325.pdf

[11] International Journal of Computer Applications (0975
- 8887) Volume 43- No.14, April 2012 38 RFID Technology for Smart Vehicle Control using Traffic Signal Speed Limit Tag Communication.

[12] Desai, G.G. and Bartakke, P.P., Real-Time Implementation Of Indian License Plate Recognition System. In *2018 IEEE Punecon* (pp. 1-5). IEEE.