Automatic Vehicle Recognition and Multi-object Feature Extraction Based on Machine Learning

E. Esakki Vigneswaran(✉) and M. Selvaganesh

Sri Ramakrishna Engineering College, Coimbatore, India
{esakkivignesh, selvaganesh.m}@srec.ac.in

Abstract. Over the past few years the influence of surveillance has left a massive impact on the social lives of people. The introduction of controlling the surroundings with the surveillance system would provide zero error and faster reaction time. In this proposed work, we have developed a system that will detect and analyze the traffic signal images. Thousands of traffic signal images are fed into the computer and trained based on the margins of particular classes. A weight file is generated from this training process. YOLO (you only look once) is an algorithm used for training the images and detecting the images. It is a network for object detection. In this project, the objects are vehicles and human beings. The identification of objects is done by searching the location on the image and arrange those objects with its prediction level. Existing methods like R-CNN and its variations, used a pipeline methodolgy to analyze and segment the images in multiple steps. Accuracy and speed of recognition is very slow in existing methodologies because of the individually done component training. Proposed methodology with YOLO is performed by a unique neural network. The output describes the computation and name of the classes from a fed image, which is used as vehicles and human beings.

Keywords: YOLO · Machine learning · OpenCV · R-CNN · CNN

1 Introduction

YOLO (You Only Look Once) is an algorithm which is used for detecting the objects in the input image or video. The main objective of this algorithm is to classify the type of object present in the image and to lay a boundary over it. This method proves to be more optimum than the previous methods like CNN, R-CNN and FR-CNN [1]. A python script is then developed to count the number of items in each class and display them in detail [4]. Using the concept of machine learning with YOLO algorithm we have developed a program to identify type and number of objects fed as input to the computer. Applications like traffic control and stand-alone cars works fine without any manual errors [2].

2 Proposed System

The objective of the proposed system is to detect the type of objects and display the count of each object in detail. An algorithm named YOLO is used for detecting and bounding the objects in the images. The thousands of traffic signal images are given into the algorithm for classification and analyzing the image components. Therefore, algorithms like R-CNN, YOLO have been developed to find these occurrences in a faster way.

2.1 Literature Survey

Shaif Choudhury describes a vehicle detection technique that can be used for traffic surveillance systems. Traffic monotiring system is proposed with Haar based cascade classifier [6]. Rashmika Nawaratne demonstrates the video surveillance system with monitoring, hazard detection and amenity management. Existing methods are lagging from learning from available videos [7]. Nirmal Purohit proposed a technique for identifying and classifying an enemy vehicle in military defense system using HOG and SVM are utilized [3]. Jia-Ping Lin show that the image recognition with YOLO can be applied in many applications of Intelligent Transportation System [5].

2.2 Training Process

Training a dataset is to make sure that the machine recognizes the input provided to the camera at the time of processing. LABELIMG is a Manual image process used for manually defining regions in an image and creating a textual description of those regions. This process is also called as Annotation. The Explicitation for the training process is illustrated in Fig. 1. First, we added a thousand images for the purpose of training. The images are split in the ratio 0:9 for validation and training respectively. The images that are trained has to be finally checked with the images saved for validation.

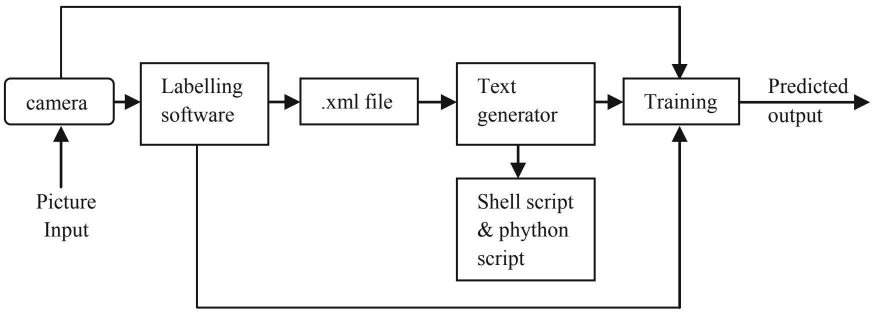


Fig. 1. Explicitation of the training process

In an LABELIMG software, the annotation for each thousand images are done i.e., all these images undergo a process called annotation where all the coordinates of the objects in the images are marked and named according to their class. After Annotation an .xml file is created. It contains the names of the class with their coordinates inside the image. The information about the annotated files are serially arranged according to the time of detection. The .xml files are then converted into text files. A text generator is used for converting the xml files into text files. A python script and a shell script are used for executing the Text generator process. The training process understands only text files so the text generator scripts are added before the process. The main principle of annotation process is to make the system learn to name and detect the objects in the image to the original one. The Schema chart for the training process is illustrated in Fig. 2.

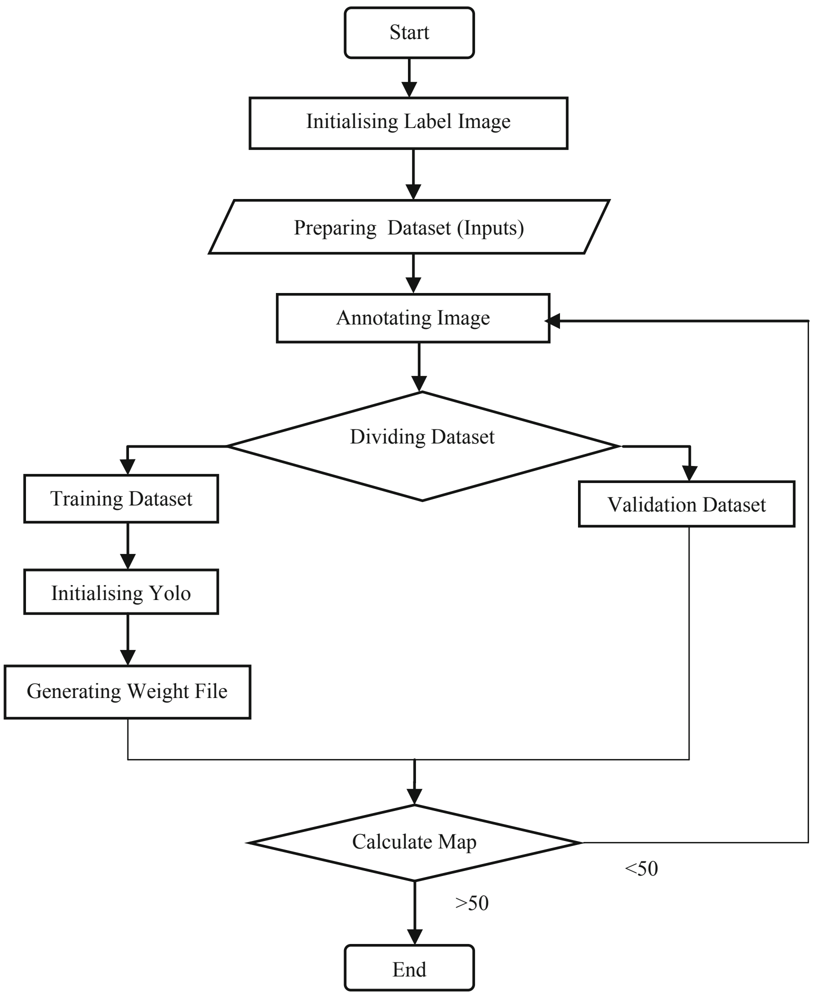


Fig. 2. Schema chart of training process

2.3 Prediction Process

At the time of process, a raw image is captured by the camera and sent to the server as shown in the Fig. 3. The server is setup for the making the process faster. The server contains the program for YOLO algorithm along with our pre-trained models. The pretrained models are nothing but the images that we trained after the validation process at the time of splitting up in the ratio 0:9. The raw image obtained from the camera now compares itself with the pre-trained images using the concept of YOLO. Now the algorithm divides the pictures into N × N grids. The system checks each grid for the number of objects inside it. If there are more than one object in a grid, the grid is then further divided into multiple grids. Once the unique objects are detected the boundaries for them declared.

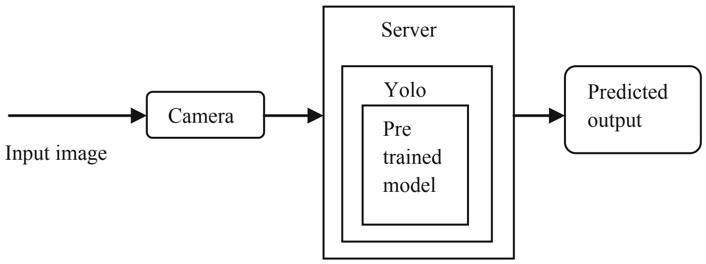


Fig. 3. Explicitation of the prediction process

Now an image with bounded objects and corresponding names in it is produced as the output. According to the code, the number of times each object detected is counted and the count is displayed. The schema chart of prediction process is shown in Fig. 4.

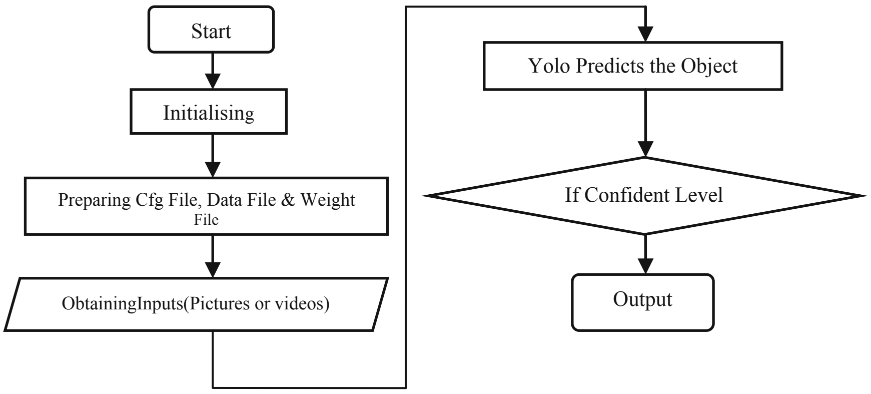

Start
Initialising
Preparing Cfg File, Data File & Weight File
ObtainingInputs(Pictures or videos)
Yolo Predicts the Object
If Confident Level
Output

Fig. 4. Schema chart of prediction process

3 Experimental Results

Once the prediction process is completed the predicted output image with objects being bounded and names being named in the image are obtained. The confident level of the object detection is increased. The output in terminal contains the accuracy level and the total count of each objects in number.

Annotation
Annotation is the initial stage of training process. The Fig. 5 shows the objects being marked as named according to their respective classes. A thousand images are trained in such a manner for prediction. For more accurate prediction a further more images are trained in the similar way.

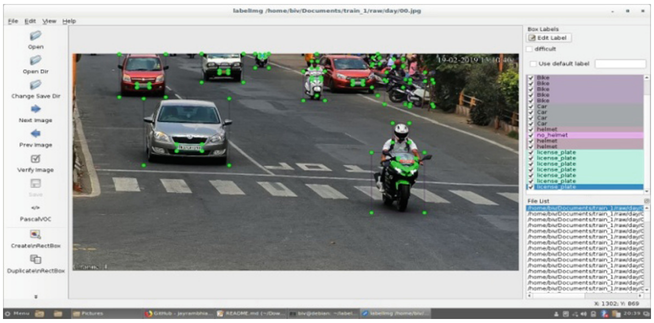


Fig. 5. Annotating the image

Image After Prediction
The process of training should have created a weight file with the details of the similar images. The YOLO compares the present image with the details available in the generated weight files and delivers an accurate predicted output. Now the algorithm will predict the type of object based on the dataset which is fed into the training process.

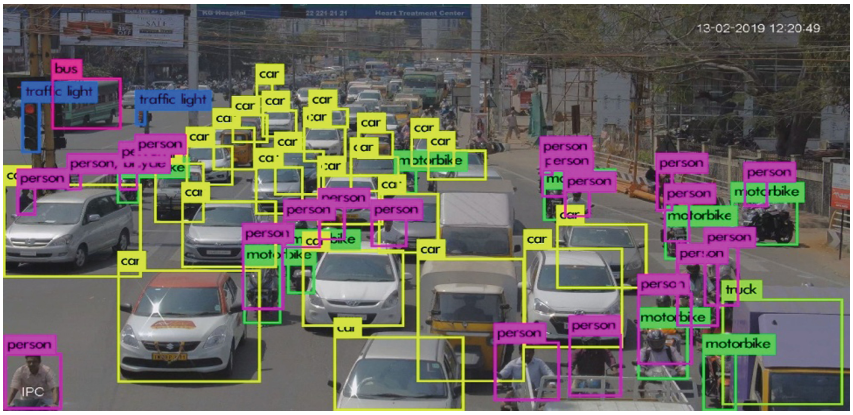


Fig. 6. Predicted output with bounded class and names (sample 1)

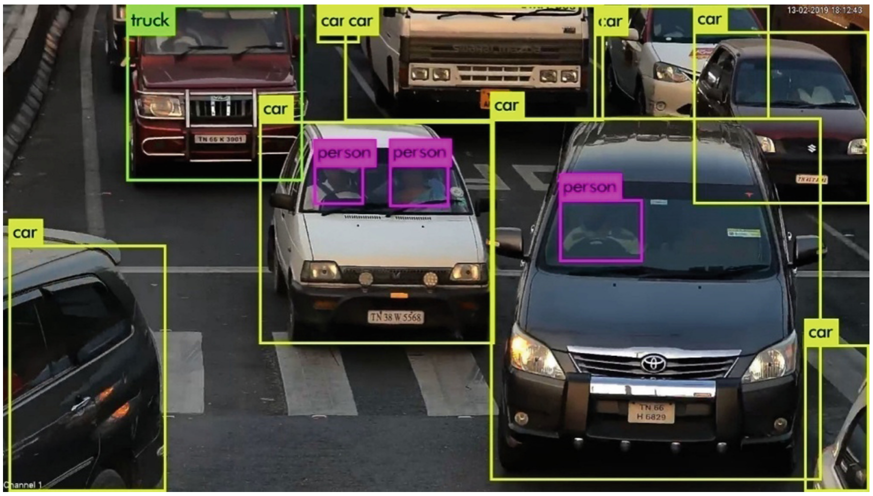


Fig. 7. Predicted output with bounded class and names (sample 2)

The objects can be identified up to 150 mm using the camera. This range varies with respect to the camera. The objects belonging to similar class are marked by a similar colour boundary as shown in the Figs. 6, 7 and 8.

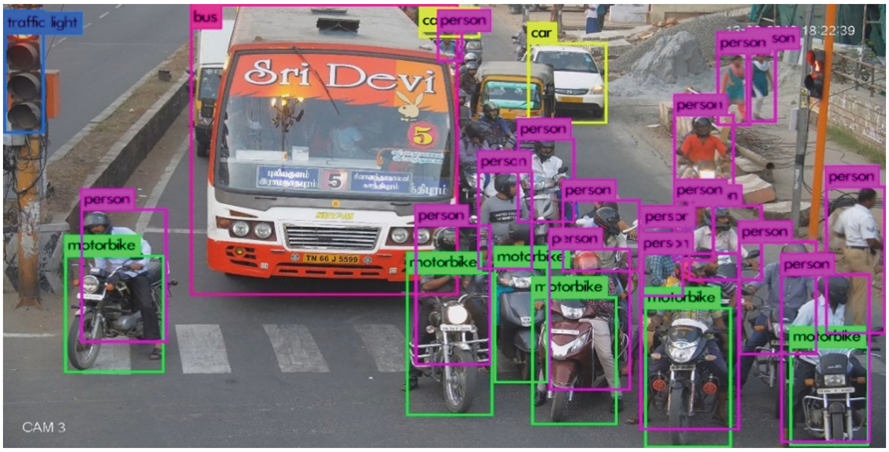
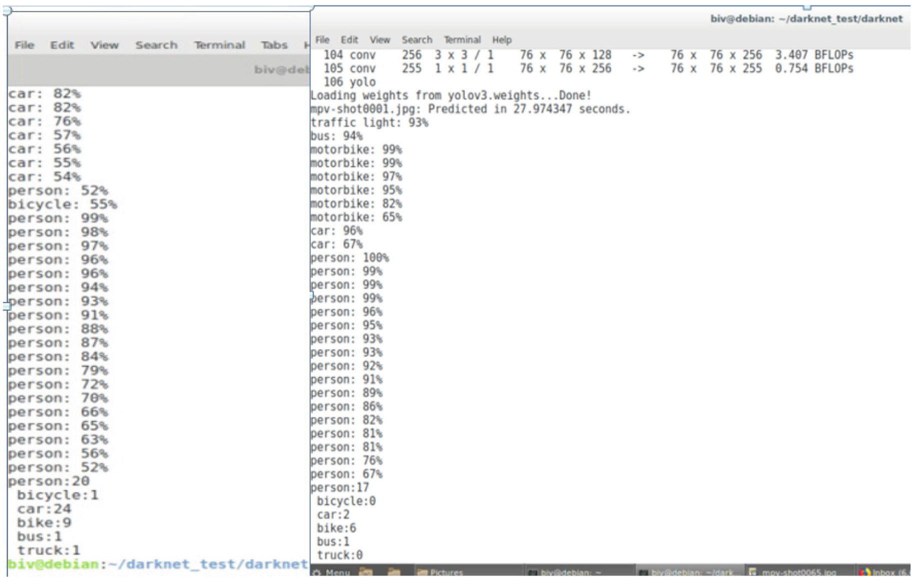


Fig. 8. Predicted output showing numbers of each classes and increased confident level (sample 1)

The predicted output consists of objects being bounded by a box and named correspondingly which is shown in Fig. 9.

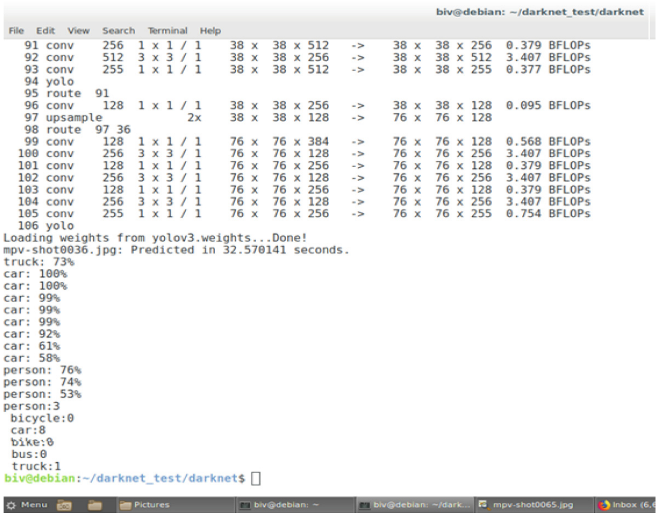

File Edit View Search Terminal Tabs
biv@deb
biv@debian: ~/darknet_test/darknet
File Edit View Search Terminal Help
Menu
Pictures
biv@debian: ~
biv@debian: ~/dark...
mpv-shot0065.jpg
Inbox

```
car: 82%
car: 82%
car: 76%
car: 57%
car: 56%
car: 55%
car: 54%
person: 52%
bicycle: 55%
person: 99%
person: 98%
person: 97%
person: 96%
person: 96%
person: 94%
person: 93%
person: 91%
person: 88%
person: 87%
person: 84%
person: 79%
person: 72%
person: 70%
person: 66%
person: 65%
person: 63%
person: 56%
person: 52%
person:20
 bicycle:1
 car:24
 bike:9
 bus:1
 truck:1
biv@debian:~/darknet_test/darknet
```

```
 104 conv     256  3 x 3 / 1    76 x  76 x 128   ->    76 x  76 x 256  3.407 BFLOPs
 105 conv     255  1 x 1 / 1    76 x  76 x 256   ->    76 x  76 x 255  0.754 BFLOPs
 106 yolo
Loading weights from yolov3.weights...Done!
mpv-shot0001.jpg: Predicted in 27.974347 seconds.
traffic light: 93%
bus: 94%
motorbike: 99%
motorbike: 99%
motorbike: 97%
motorbike: 95%
motorbike: 82%
motorbike: 65%
car: 96%
car: 67%
person: 100%
person: 99%
person: 99%
person: 99%
person: 96%
person: 95%
person: 93%
person: 93%
person: 92%
person: 91%
person: 89%
person: 86%
person: 82%
person: 81%
person: 81%
person: 76%
person: 67%
person:17
 bicycle:0
 car:2
 bike:6
 bus:1
 truck:0
```

Fig. 9. Predicted output showing numbers of each classes and increased confident level (sample1 & sample 2)

The predicted output consists of objects being bounded by a box and named correspondingly which is shown in Fig. 10.



```
   91 conv     256  1 x 1 / 1    38 x  38 x 512   ->    38 x  38 x 256  0.379 BFLOPs
   92 conv     512  3 x 3 / 1    38 x  38 x 256   ->    38 x  38 x 512  3.407 BFLOPs
   93 conv     255  1 x 1 / 1    38 x  38 x 512   ->    38 x  38 x 255  0.377 BFLOPs
   94 yolo
   95 route  91
   96 conv     128  1 x 1 / 1    38 x  38 x 256   ->    38 x  38 x 128  0.095 BFLOPs
   97 upsample            2x    38 x  38 x 128   ->    76 x  76 x 128
   98 route  97 36
   99 conv     128  1 x 1 / 1    76 x  76 x 384   ->    76 x  76 x 128  0.568 BFLOPs
  100 conv     256  3 x 3 / 1    76 x  76 x 128   ->    76 x  76 x 256  3.407 BFLOPs
  101 conv     128  1 x 1 / 1    76 x  76 x 256   ->    76 x  76 x 128  0.379 BFLOPs
  102 conv     256  3 x 3 / 1    76 x  76 x 128   ->    76 x  76 x 256  3.407 BFLOPs
  103 conv     128  1 x 1 / 1    76 x  76 x 256   ->    76 x  76 x 128  0.379 BFLOPs
  104 conv     256  3 x 3 / 1    76 x  76 x 128   ->    76 x  76 x 256  3.407 BFLOPs
  105 conv     255  1 x 1 / 1    76 x  76 x 256   ->    76 x  76 x 255  0.754 BFLOPs
  106 yolo
Loading weights from yolov3.weights...Done!
mpv-shot0036.jpg: Predicted in 32.570141 seconds.
truck: 73%
car: 100%
car: 100%
car: 99%
car: 99%
car: 99%
car: 92%
car: 61%
car: 58%
person: 76%
person: 74%
person: 53%
person:3
 bicycle:0
 car:8
 bike:0
 bus:0
 truck:1
biv@debian:~/darknet_test/darknet$ 
```

Fig. 10. Predicted output with bounded class and names (sample 3)

4 Conclusion

In this obligation we have determined and classified the type of objects like vehicles and human beings from the obtained images. The usage of YOLO algorithm decreased the process time and produces a more optimised output than the existing methodologies. By fixing a certain area in a captured image, people those who violate traffic rules can also be identified and a penalty can be made in a digital way. The number of vehicles in the image is found out and it can be used for traffic control. YOLO accessess to the whole image by predicting boundaries. We enforces spatial diversity in making predictions. Self driving (unmanned) cars can use this proposed technique for detecting the objects in front of them and driving without collisions. The outgrowth shows that, this process is more efficient and faster than the existing methods.

References

1. Braun, M., Krebs, S., Flohr, F., Gavrila, D.: EuroCity persons: a novel benchmark for person detection in traffic scenes. IEEE Trans. Pattern Anal. Mach. Intell. **9**(1), 1–8 (2019)
2. Chen, Z., Huang, X.: Pedestrian detection for autonomous vehicle using multi-spectral cameras. IEEE Trans. Intell. Veh. **11**(7), 1–9 (2019)
3. Purohit, N., Israni, D.: Vehicle classification and surveillance using machine learning technique. In: IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology, vol. 20, no. 34, pp. 910–914 (2018)
4. Zhao, M., Zhao, C., Qi, X.: Comparative analysis of several vehicle detection methods in urban traffic scenes. In: IEEE International Conference on Sensing Technology, vol. 7, no. 19, pp. 119–126 (2018)
5. Lin, J.P., Sun, M.T.: A YOLO-based traffic counting system. In: IEEE Conference on Technologies and Applications of Artificial Intelligence, vol. 10, no. 23, pp. 82–85 (2018)
6. Choudhury, S., Chattopadhyay, S.P., Hazra, T.K.: Vehicle detection and counting using haar featurebased classifier. In: IEEE Annual Industrial Automation and Electromechanical Engineering Conference, vol. 17, no. 11, pp. 106–109 (2017)
7. Nawaratne, R., Bandaragoda, T., Adikari, A., Alahakoon, D., De Silva, D., Yu, X.: Incremental knowledge acquisition and selflearning for autonomous video surveillance. In: IEEE Annual Conference of the IEEE Industrial Electronics Society, vol. 10, no. 15, pp. 4790–4795 (2017)
8. Saribas, H., Cevikalp, H., Kahvecioglu, S.: Car detection in images taken from unmanned aerial vehicles. In: IEEE Signal Processing and Communications Applications Conference, vol. 10, no. 8, pp. 840–845 (2017)
9. Lin, C.-Y., Chang, P., Wang, A., Fan, C.-P.: Machine learning and gradient statistics based real-time driver drowsiness detection. In: IEEE International Conference on Consumer Electronics, vol. 4, no. 12, pp. 1801–1807 (2018)
10. Zhigang, Z., Huan, L., Pengcheng, D., Guangbing, Z., Nan, W., Wei-Kun, Z.: Vehicle target detection based on R-FCN. In: IEEE Chinese Control And Decision Conference, vol. 10, no. 19, pp. 5739–5743 (2018)
11. Lee, K.H., Hwang, J.N.: On-road pedestrian tracking across multiple driving recorders. IEEE Trans. Multimedia **17**(9), 1429–1438 (2015)
12. Liu, W., Lau, R.W.H., Wang, X., Manocha, D.: Exemplar-amms: recognizing crowd movements from pedestrian trajectories. IEEE Trans. Multimedia **18**(12), 2398–2406 (2016)