A Novel Method for Multiple Object Detection on Road Using Improved YOLOv2 Model

P.Gunasekaran[1], A.Azhagu Jaisudhan Pazhani[2] and T.Ajith Bosco Raj[3]
[1, 2]Department of ECE, Ramco Institute of Technology, Rajapalayam, Tamilnadu, India.
[3]Department of ECE, PSN College of Enginnering and Technology, Tirunelveli, India
E-mail: mailtogunasekar@gmail.com, alagujaisudhan@gmail.com, ajithboscoraj@gmail.com

Keywords: AI, Object, Vehicle, Convolutional, VOC, COCO, KITTI, YOLO

Received: December 30, 2021

Object detection is a branch of machine vision and image processing that deals with instances of a certain class of semantic items. One of the most significant habits of object detection in intelligent transportation schemes is vehicle detection. Its aim is to extract clear-cut vehicle-type information from photographs or videos of automobiles. A fully convolutional network (FCN) is employed in sophisticated driver assistance systems for high performance and quick object identification (ADAS). A novel vehicle detection model employing YOLOv2 is presented to tackle the difficulties of prevailing vehicle detection, such as the absence of vehicle-type recognition, stumpy detection accuracy and sluggish speed. The detection model is trained using the VOC and COCO datasets, and the detection enactment is evaluated quantitatively using KITTI training pictures. In addition, the performance of the YOLOv2 model was compared to that of prior models.

Povzetek: Razvita je nova metoda zaznavanja več objektov na cesti s pomočjo YOLOv2 modela.

1 Introduction

Moving object detection is a computer technique that compacts with recognizing occurrences of semantic matters of a precise class (such as humans, automobiles, etc.) in a digital picture or video. It is connected to computer vision, image processing, and neural networks. Vehicle detection and pedestrian detection are two well-studied fields. In the field of machine vision, moving object detection has a variety of applications, including picture retrieval and video monitoring.

While new research datasets have increased the number of training sets and testing instances to get closer to real-world situations, detectors' capacity to process big data sets in an acceptable period of time has become a significant concern in addition to accuracy. It is not just the number of classes that matters, but also the training examples.

Detecting moving items in a video clip entail finding them in the frame. Item detection is required by every tracking technique, whichever in all frame or when the object first shows in the video. Various backdrop removal approaches from the literature were simulated for moving object detection. Background subtraction uses the relative difference between the current image and the reference updated backdrop over time. Background subtraction that works well should be able to deal with fluctuating lighting conditions, background clutter, shadows, camouflage, bootstrapping, and foreground segmentation in real time.

The tracking of moving objects in video images has flickered a lot of interest in machine vision. Surveillance systems, navigation systems, and object identification all flinch with object tracking. Object tracking is extremely important in a real-time environment because it allows for an improved sense of refuge through visual information, security and surveillance to recognize people, analysis of customer shopping behavior in retail spaces, video abstraction to attain involuntary annotation of videos, generation of object-based synopses, traffic management to examine flow, and design futuristic video effects.

Huieun Kim et al. offered "On-road object identification using Deep Neural Network" [4], which advocated SSD as a quicker object detection method than R-CNN by 41 frames per second. The model is built on SSD and tweaked with the KITTI dataset, which is made up of on-road environment object classes (SSD is a pre-trained model by Pascal VOC pictures). This work proposes an on-road object identification method based on SSD that overcomes the difficulties of detecting on-road objects using a camera in instantaneous and allows for robust object detection. It creates appearance characteristics from input pictures using convolutional layers and trains object position in 2D image coordinates by calculating loss of object box position (IoU) during the training step. SSD, on the other hand, has the disadvantage of overlooking tiny things due to its grid methodology.

The furthermost representative FCN-based object identification approaches are region-based fully convolutional networks (R-FCN), single shot multi-box detector (SSD), and you only look once (YOLO). To obtain good detection performance, these approaches

need a large amount of labeled training data. Most deep learning-based detection algorithms train the classification model using millions of ImageNet classification datasets and fine-tune it by detection training data such as tens of thousands of PASCAL VOC and COCO datasets [13]. The detection approaches based on deep learning, on the other hand, need a high level of computing complication to train the detection models.

A FCN-based object identification approach that enhances performance in a road environment was suggested in the publication "High Performance and Fast Object Detection in Road Environments" [9]. Although the SSD input network is slighter than that of YOLO, the processing time is significantly longer. The classification-specific layer design and the amount of default boxes account for the performance disparity. The VGG-16 model castoff in SSD requires around four times more processing resources than the Darknet-19 model used in YOLO in the classification-specific layer.

Hui-Lee Ooi et al. used an object detector to evaluate the MOT in urban traffic sceneries with road users of various vehicle sizes, whereas earlier work in this area has used background removal or optical flow to excerpt the items of interest regardless of size. The work involves a review of a common model object detector for tracking in urban traffic divisions, as well as the addition of label information to describe the items in the scene. The label information should be a valuable signal to differentiate and associate the objects of interest through frames, resulting in a more precise trajectory, due to the diversity of objects prevalent in urban landscapes. This is documented in the paper “Multiple Object Tracking in Urban Traffic Scenes with a Multiclass Object Detector” [5].

Due to its efficiency and accuracy, a deep-learning object identification model from the Region-based Fully Convolutional Network (R-FCN) framework is used to recognize the road users in each frame. The top performing technique on the MIO-TCD localization contest led to the selection of this detector. The pre-trained model is refined further by using the MIO-TCD dataset to deliver labels for the various road users seen in traffic scenes, each of which falls into one of eleven classes or labels.

The work "Survey of Pedestrian Detection for Advanced Driver Assistance Systems" [3] focuses on one form of ADAS in particular, pedestrian protection systems (PPSs). This study focuses on pedestrians since, according to accident data, 70% of persons engaged in car-to-pedestrian collisions were in front of the vehicle, with 90% of them moving. As a result, PPSs frequently employ forward-facing sensors. The foreground segmentation algorithm detects moving people. The INRIA Person Data set, which is now fairly popular for general human categorization assessment but comprises a significant number of samples derived from high resolution pictures, was employed in this model.

The work "The Object Detection Based on Deep Lea rning" [15] provides an overview of object detection and discusses the relationship and differences between the co nventional and deep learning methods. The study focuses on the framework design, model working principles (YO LO, SSD), realime model performance analysis, and dete ction accuracy.

In the work "Integrated Real-Time Object Detection for Self-Driving Vehicles," [11], the authors propose combining Fast R-CNN with YOLO to obtain real-time performance with around half the YOLO localization inaccuracy. The ImageNet 2012 dataset was used to pre-train the model. This may lower the likelihood. However, the model has trouble identifying tiny items that are close together.

To attain the greatest accuracy result and speed, the work "Comparative study of Object Detection Algorithms" [12] focuses on three distinct models, namely SSD, faster R-CNN, and R-CNN. On the COCO dataset, these models are trained and their performance indicators are evaluated. The test is run on the same hardware and includes a variety of model combinations.

Mendes, et al., proposed a method to detect object, when an object centroid passes over a region of interest, ROI ID and region type (in/out) are saved into object properties [8]. This ID will be used along the vehicle’s lifetime over next frames to determine its route. At this moment, object ID is stored in a result set to prevent duplicate in counting. This is necessary because ROI is a polygon (not a single line) and an object will pass over the region in multiple sequential frames

For improving school bus routing and scheduling, see the study "Improving efficiency of school bus routing using AI based on bio-inspired computing" [2]. The accuracy of the School bus routing problem may be enhanced further by utilizing a genetic algorithm that compacts with data preparation, routing, and bus stop selection (SBRP). The author of the work "Moving Object Tracking in Video" [18] proposes a technique for isolating moving objects in video sequences, followed by a rule-based tracking system. The introductory testing findings show that the algorithm works even in difficult conditions like a new track, a halted track, a track collision, and so on.

Azhagu Jaisudhan Pazhani1., et al., proposed Faster R-CNN which comprise of a combination of Faster R-CNN with enhanced ROI pooling, named as FrRNet-ERoI frame-work. It is pipeline process meant to establish the result as detected object for given test image. The network comprises of two sections namely region proposal network and fast R-CNN [16].

The work "Moving Object Tracking in Video Using Matlab" [1] discusses a tracking approach without background extraction. Since, when removing backdrop from a video frame, if there are little moving objects in that frame, they form a blob in thresholding, which causes confusion while tracking that blob because it is of no use. The author covers video tracking in computer vision in his work "Video-Based People Tracking" [7], which includes design criteria and a study of solutions ranging from simple window tracking to tracking complicated, deformable objects by learning shape and dynamics models.

Markus Schreiber., et al., proposed a sequential processing of the GNSS raw data i.e. each measurement

is processed as a single measurement. Given n pseudo range measurements at one time, n estimation steps are carried out successively [10]. The alternative would be to take a measurement vector including all measured values present at one time and process them in a single measurement step.

2 Object Detection Models Based on region proposal

The extraction of region candidates and the construction of deep neural networks are the two key tasks in the deep learning object identification based on region proposal.

2.1 R-CNN

One of the earliest models to employ convolutional neural networks for object detection was the R-CNN model. R-purpose CNN's is to yield in an image and properly recognize the key items in the image. R-CNN does exactly what it sounds like it should: it proposes a lot of boxes in the picture and checks to see if any of them belong to an item. R-CNN uses a procedure called Selective Search to generate these bounding boxes, or region proposals.

The Regions of Interest (RoI) are created first. The RoIs are category-agnostic bounding boxes with a high probability of covering an intriguing object. Selective Search is the approach employed in the study to generate them; however other region creation methods can be used instead. The characteristics from each area suggestion are then extracted using a convolutional network. The bounding box's sub-image is twisted to match the CNN's input size before being sent to the network. Following the network's extraction of topographies from the input, the topographies are sent into support vector machines (SVM), which do the final classification. Starting with the convolutional network, the approach is trained in steps. The SVMs are fitted to the CNN features once the CNN has been trained. Finally, the region proposal creating method is trained.

The R-CNN approach is significant since it was the first feasible key for object detection with CNNs. Because it was the first, it has a number of flaws that succeeding systems have addressed. R-three CNN's key issues are: First, as previously said, training is divided into many stages. Second, training is too expensive. Topographies are retrieved from every region proposal and kept on disc for both SVM and region proposal training. This will take days to compute and hundreds of gigabytes of storage. Third, and probably most importantly, object detection is sluggish, taking about a minute per image even when using a GPU. This is due to the fact that the CNN forward calculation is done independently for each item suggestion, even if they come from the same picture or overlap.

2.2 Fast R-CNN

Girshick's Fast R-CNN, published in 2015, is a more real solution for object recognition. Instead of doing the forward pass of the CNN sequentially for each RoI, the fundamental concept is to conduct it for the whole picture [14].

The technique takes an image and computes areas of interest from it as input. The RoIs are created using an external mechanism, same like in R-CNN. A CNN with numerous convolutional and max pooling layers is used to process the image. After these layers, the convolutional feature map is formed and fed into a RoI pooling layer. The feature map is used to derive a fixed-length feature vector for each RoI. The feature vectors are then fed into fully connected layers, which are coupled to two output layers: a softmax layer that generates probability estimates for object classes, and a real-valued layer that generates bounding box co-ordinates based on regression.

The region proposer was still a bottleneck with Fast R-CNN that needed to be addressed. To detect the locations of objects, the first step is to create a set of potential bounding boxes or regions of interest to test. These suggestions were developed in Fast R-CNN employing Selective Search, a somewhat slow method that was discovered to represent the entire process' bottleneck.

2.3 Faster R-CNN

Faster R-CNN discovered a solution to make the region proposal phase nearly free. Faster R-CNN exposed that region recommendations were grounded on picture attributes that had previously been estimated during the CNN's forward pass (first step of classification). A single CNN is employed in this model to perform both region recommendations and classification. Only one CNN has to be educated, and region suggestions may be made for absolutely little cost. Faster R-CNN creates the Region Proposal Network by layering a Fully Convolutional Network on top of the CNN's characteristics.

By alternating between training for RoI generation and detection, a Faster R-CNN network is developed. Two distinct networks are first trained. These networks are integrated and fine-tuned after that. Certain layers are fixed during fine-tuning, while others are trained in turn.

A single picture is sent into the trained network. The image's feature maps are generated by the shared fully convolutional layers. The RPN receives these feature maps. The RPN generates region suggestions, which are sent into the final detection layers together with the feature maps. These layers yield the final classifications and contain a RoI pooling layer. Region suggestions are practically costless to compute thanks to shared convolutional layers.

The use of a CNN to compute region suggestions has the extra benefit of being GPU-friendly. A CPU is used to implement traditional RoI generating methods like

Selective Search. To identify the items, all of the object detection methods presented so far employs areas. The network does not look at the entire image at once, but instead focuses on different areas of it in a sequential manner. Two issues arise as a result of this:

- To extract all of the items, the programme must pass over a single image many times.
- Because there are several systems operating simultaneously, the performance of the systems that follow is influenced by the performance of the prior systems.

2.4 R-FCN

Faster R-CNN was an order of magnitude quicker than its predecessor fast R-CNN thanks to the performance boost. However, there was an issue with applying the region-specific component multiple times in an image; this issue was resolved in R-FCN, where the computation required per image was drastically reduced by cropping features from the last layer of features prior to predictions, rather than harvesting features from the same layer where the crops are predicted. When utilising Resnet101 as the feature extractor, the approach is quicker than Faster R-CNN while attaining equal accuracy ratings. In hindsight, it also respects translational invariance since it is a position sensitive cropping mechanism.

2.5 Based on regression SSD

The Single Shot MultiBox Detector (SSD) goes much farther in terms of integrated detection. There is no resampling of picture segments, and the approach does not create any recommendations. It creates object detections using a single pass of a convolutional network. The approach starts with a default set of bounding boxes, similar to a sliding window method. Offset parameters indicate how much the right bounding box encircling the item differs from the default box in the object predictions made for these boxes.

The classifier uses feature maps from multiple distinct convolutional layers (i.e. larger and smaller feature maps) as input to cope with diverse scales. The classifier is followed by a non-maximum suppression stage, which removes most boxes below a particular confidence level because the approach creates a dense collection of bounding boxes.

2.6 YOLO

To begin, create a VGG16 classifier network. Then, for object detection, replace the completely linked layers with a convolution layer and retrain it endways. YOLO uses 224 224 images to train the classifier, followed by 448 448 images for object recognition [6]. YOLOv2 trains the classifier using 224 224 images at first, but then retrains it with 448 448 images in a considerably shorter time frame. This simplifies detector training while also increasing mAP by 4%.

3 Proposed method for multiple object detection

3.1 YOLOv2 model

YOLOv2 is a more advanced version of the original YOLO. YOLO9000 is based on YOLOv2; however, it is trained on a combined dataset that combines the COCO detection dataset with ImageNet's top 9000 classes.

3.2 YOLOv2 Improvement

To improve the accuracy and speed of YOLO prediction, a number of changes are made, including:

3.3 Image resolution matters

The detection performance is improved by fine-tuning the basis model with high-resolution photos.

3.4 Convolutional anchor box detection

Rather of using fully-connected layers to predict bounding box positions throughout the whole feature map, YOLOv2 employs convolutional layers to predict anchor box locations, similar to quicker R-CNN. Class probabilities and spatial location predictions are disconnected. Overall, the modification results in a modest reduction in mAP while increasing recall.

3.5 K-mean clustering of box dimensions

Unlike the speedier R-CNN, which employs hand-picked anchor box sizes, YOLOv2 uses k-mean clustering to discover acceptable priors on anchor box dimensions on the training data. The distance metric is built around IoU scores:

$$\mathbf{dist}\ (y, z_k) = 1 - \mathbf{IoU}\ (y, z_k),\ k = 1 \text{ to } M$$

If x is a candidate for a ground truth box and ci is one of the centroids. The elbow approach may be used to choose the best number of centroids (anchor boxes) k.

3.6 Direct location prediction

YOLOv2 formulates the bounding box prediction in such a way that it does not deviate too far from the centre. The model training may become unstable if the box location prediction may position the box in any section of the picture, as in the regional proposal network.

3.7 Add fine-grained features

A passthrough layer is added to YOLOv2 to convey fine-grained characteristics from an earlier layer to the final output layer. This passthrough layer uses the same approach as ResNet's identity mappings to retrieve higher-dimensional information from preceding layers. This results in a 1% improvement in performance.

3.8 Multi-scale training

Every 10 batches, a new size of input dimension is randomly picked to train the model to be resilient to input photos of various sizes. The freshly sampled size is a multiple of 32 since the convolution layers of YOLOv2 down sample the input dimension by a factor of 32.

3.9 Architecture of YOLOv2 model

Between the input and output, the architecture represented in figure 3.1 has multiple hidden levels. The convolution layer, ReLU, pooling layer, and fully connected layer are all part of the hidden layer. Finally, the softmax layer is used to determine the output probability range.

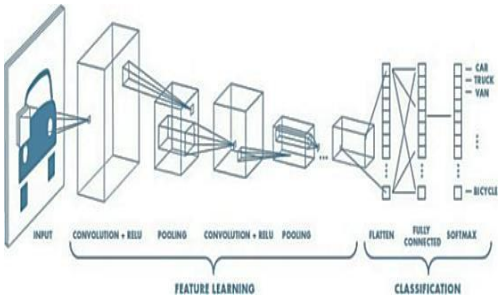


Figure 3.1: Architecture of YOLOv2 model

3.10 Convolution layer

A convolutional neural network's basic building part is the convolution layer. As you progress through the convolution layers, the filters perform dot products on the previous convolution layers' input. As a result, they're using the smaller cultured bits or edges to create larger pieces. The convolution layer, in general, is made up of many filters that extract characteristics from the input picture.

3.11 ReLU Layer

Convolutional neural networks do not have a distinct component called ReLU. The goal of using the rectifier function is to make the pictures more non-linear. The rectifier is used to further breakdown the linearity in order to compensate for the linearity imposed on an image when it is processed through the convolution function. Examine the changes in figure 3.2 as it goes through the convolution and rectification processes.

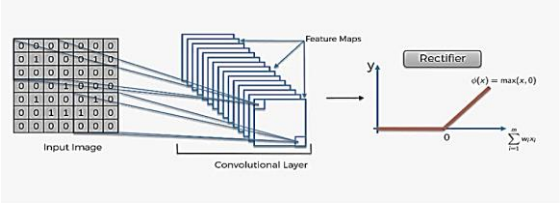


Figure 3.2: ReLU layer

3.12 Pooling layer

A CNN's pooling layer is another component. Its purpose is to gradually shrink the representation's spatial dimension in order to minimize the number of parameters and computations in the network.

3.13 Fully connected layer

Each neuron in one layer is attached to every neuron in alternative layer in fully connected layers. It works in the same way as a standard multi-layer perception neural network in theory. The image is classified using the flattened matrix.

3.14 Softmax

The softmax function in mathematics normalizes an unnormalized vector into a probability distribution. In neural networks, it's frequently used to translate non-normalized output to a probability distribution across expected output classes.

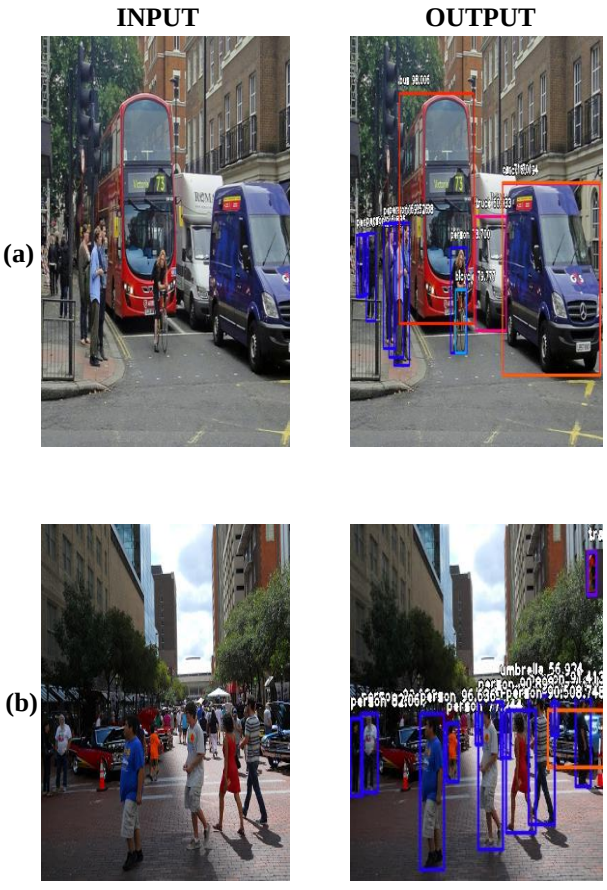
4 Results and discussion

Python was used to create the suggested work. The model is built with data from the PASCAL VOC and COCO datasets. The convolution layer, pooling layer, and activation layers such as ReLU and softmax are used to build the model at first. To excerpt the features from the input picture, all of the hidden layers are employed. Finally, to forecast the probability of prediction, the completely linked layer is added.

There are numerous models for object detection, including Faster R-CNN, SSD, and YOLO. These models are implemented and their performance is evaluated in this work. The suggested model YOLOv2 is created, and the model's performance is evaluated using various input photos.

The YOLOv2 model was designed to identify pedestrians in a road environment. It has also been improved to detect a variety of items such as a bicycle, automobile, bus, motorcycle, and truck. The item is recognized and the likelihood of prediction is displayed via anchor boxes.

4.1 YOLOv2



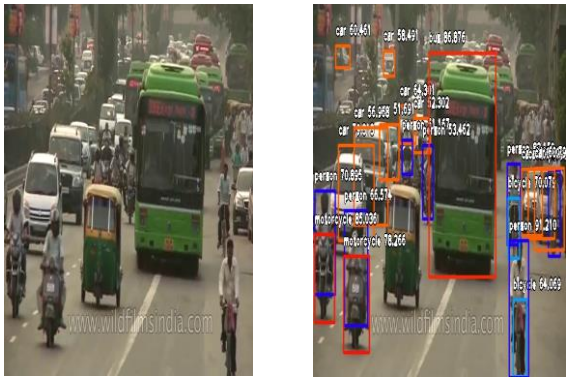


Figure 4.1: Input and Prediction output for YOLOv2 model

Figure 4.2: Output for person detection in a video

Figure 4.3: Output for multiple object detection in a video

Figure 4.2 demonstrates how the YOLOv2 model recognizes just pedestrians in a video, but figure 4.3 shows how the model detects numerous things in the input video, such as a bus, bicycle, automobile, and person.

Table 1: Performance result of various models

Objects / Models	Bus	Person	Bicycle	Car
Faster R-CNN	99.5	76.0	81.9	99.4
SSD	98.0	73.7	79.7	71.7
YOLO	100	96.3	93.1	99.8
YOLOv2	98.5	99.6	97.0	98.5

The accompanying table 1 shows the chance of detection for various models of bus, person, bicycle, and automobile. Figure 4.4 depicts a comparative study of several models.

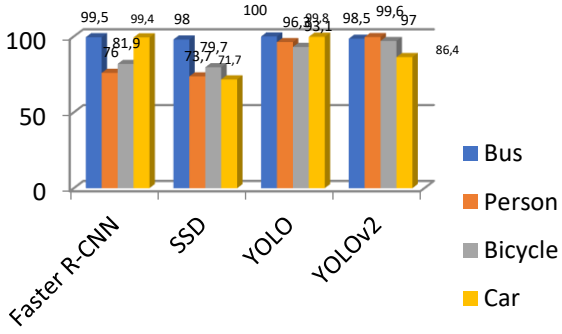


Figure 4.4: Comparison study of probability of detection for the various models

5 Conclusion

Currently, many deep learning frameworks, including TensorFlow, provide multiple versions of pre-trained object identification models. The goal of this work is to identify many items in a stable environment. Using YOLOv2, high accuracy in object recognition and tracking is achieved. YOLOv2 takes an efficient technique by first predicting the portions that contain the essential data and then classifying them using CNN. It just looks at the image once, which increases the speed of object detection. To detect the items in the video, the pre-trained object detection model is used. The likelihood of detecting various items is used to calculate the detection model's performance. In Pascal VOC detection dataset, the YOLOv2 model yields detection probabilities of 98.5 percent (bus), 99.6 percent (person), 97 percent (bicycle), and 86.4 percent (vehicle), whereas competing systems, such as the enhanced version of Faster R-CNN and SSD, only obtain lower results.

References

[1] Bhavana C. Bendale, et al., (2012). "Moving Object Tracking in Video Using MATLAB", International Journal of Electronics Communication and Soft Computing Science and Engineering ISSN: 2277-9477, Vol 2, Issue 1.

[2] Gawande P.V, Lokhande S.V, (2018). "Improving efficiency of school bus routing using AI based on bio inspired computing: A Survey", International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056 ,p-ISSN: 2395-0072, Volume: 05 Issue: 03.

[3] Gerónimo, D., et al., (2010). "Survey of Pedestrian Detection for Advanced Driver Assistance Systems", IEEE Transactions on Pattern Analysis and Machine Intelligence, doi:10.1109/tpami.2009.122.

[4] Huieun Kim, et al., (2016). "On-road object detection using Deep Neural Network", IEEE International Conference on Consumer Electronics-Asia (ICCE-Asia), DOI: 10.1109/ICCE-Asia.2016.7804765.

[5] Hui-Lee Ooi, et al., (2018). "Multiple Object Tracking in Urban Traffic Scenes with a Multiclass Object Detector", published on 13th International Symposium on Visual Computing (ISVC), Cornal University, arXiv:1809.02073 [cs.CV].

[6] J. Redmon and A. Farhadi. (2017). "YOLO9000: Better, Faster, Stronger" Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition.

[7] Marcus A. Brubaker, et al., (2010). "Video-Based People tracking", hand book of ambient intelligence under smart environments, pp 57-87.

[8] Mendes, et al., (2015). "Vehicle Tracking and Origin-Destination Counting System for Urban Environment" in proceedings of the International Conference on Computer Vision Theory and Applications.

[9] Minsung Kang, Young-Chul Lim, (2017). "High Performance and Fast Object Detection in Road Environments", Seventh International Conference on Image Processing Theory, Tools and Applications (IPTA), DOI:10.1109/IPTA.8310148.

[10] M. Schreiber, et al., (2016). "Vehicle localization with tightly coupled GNSS and visual odometry", in Proc. IEEE Intelligent Vehicles Symposium.

[11] Naghavi, S. H., et al., (2017). "Integrated real-time object detection for self-driving vehicles" 10th Iranian Conference on Machine Vision and Image Processing (MVIP). doi:10.1109/iranianmvip.2017.834234.

[12] Nikhil Yadav, et al., (2017). "Comparative Study of Object Detection Algorithms", International Research Journal of Engineering and Technology (IRJET), e-ISSN: 2395-0056, p-ISSN: 2395-0072, Vol 4.

[13] O. Russakovsky, et al., (2015). "ImageNet Large Scale Visual Recognition Challenge", in computer vision. vol. 115, no. 3, pp. 211–252.

[14] S. Ren, K. He, et al., (2015). "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks" Nips, pp. 1–10.

[15] Tang, C., et al., (2017). "The Object Detection Based on Deep Learning", 4th International Conference on Information Science and Control

Engineering (ICISCE), DOI:10.1109/icisce.2017.156.

[16] Azhagu Jaisudhan Pazhani1. et al., (2021). "Object detection in satellite images by faster R-CNN incorporated with enhanced ROI pooling (FrRNet-ERoI) framework" Earth Science Informatics, https://doi.org/10.1007/s12145-021-00746-8.