Turkish Journal of Computer and Mathematics Education Vol.12 No. 11 (2021), 2662- 2676
Research Article

An Efficient Algorithm for Real-Time Vehicle Detection Using Deep Neural Networks

[1]Sri Jamiya S, [2]Esther Rani P
[1]Research Scholar, Electronics and Communication Engineering,
Vel Tech Rangarajan Dr. Sagunthala R&D Institute of Science and Technology,Avadi, Chennai. India.
[2]Professor, Electronics and Communication Engineering,
Vel Tech Rangarajan Dr. Sagunthala R&D Institute of Science and Technology,Avadi, Chennai. India.

Article History: Received: 11 January 2021; Revised: 12 February 2021; Accepted: 27 March 2021; Published online: 10 May 2021

Abstract: Vehicle detection is one of the major tasks in the field of Computer vision and Intelligent video surveillance systems. In this paper, we present a novel approach to detect vehicles and classify them. The proposed system is based on the YOLOv3 object detection algorithm. The backbone network we used for Feature Extraction is pretrained DenseNet121. The YOLOv3 network is further tweaked on the detection layer by pacing an extra prediction scale to form the desired architecture of our research. We also incorporated Distance IoU loss and DIoU-NMS in the network which further boosts the accuracy of the network. This network has four scales of prediction layers, with the fusion of these layers, networks predictive function increases. The proposed network is tested by modifying various parameters to find the optimal results. Our network is proven to be effective in the detection of vehicles during night time, long-distance vehicles, and occluded vehicles. Our experiments on PASCAL VOC 2007, 2012 and COCO datasets achieves desired results in the improved network. We also created a dataset based on night-time vehicle images from Traffic CCTV footages to maximize detection accuracy during extreme weather conditions, especially at night. The Improved YOLOv3-Net model with input size 608 × 608 accomplishes high mAP of 82.8 % which is a promising result in detecting real time videos.

Keywords: Vehicle Detection, Object Detection, Deep Neural Network, Convolutional Neural Networks, YOLOv3, DenseNet, Distance IoU

1 INTRODUCTION

Vehicle detection has great significance in designing an autonomous vehicle driving framework and in the Traffic management system. Vehicle detection and recognition are always focused on locating vehicles in the frames but for major analysis further processing is important. At present, deep learning is at the highest level for detecting objects. Many researchers prove that by deep learning we can detect the object in real-time with the highest level of accuracy. Machine learning networks are accurate in prediction in the same way as deep learning but machine learning requires structured data whereas deep learning depends on learning information from network layers. To detect objects these networks always rely on convolutional neural networks (CNNs), and deep CNNs (DCNNs). Some variants of CNN are ResNet101, VGG16, and R-CNN [1-3]. These networks achieved significant performance in Object detection Contests. R-CNN models are the first ones to present object detection in a deep neural network using a technique of selective search algorithm to propose the valiant region of interest proposals. Followed by this model Fast R-CNN, Faster R-CNN and R-FCN algorithms predict objects using these functions [4-6].

A deep Multi-scale CNN (MS-CNN) is an enhanced form of CNN which has two sub-networks [7], a detection network followed by region proposal network, like R-CNN framework to find desired bounding boxes in the image and then classifier is used on these detected bounding boxes. One of the best optimizing techniques used in Computer vision is bounding box regression to predict better localization. Most of the tasks like multiple object detection, tracking of objects, object localization and instance segmentation depends especially on precise bounding box regression. For improving deep learning neural networks either its backbone for feature extraction is improved or a better technique to extract these features must be followed.

To improve accuracy many networks replace regression loss functions which mainly depends on Intersection over Union (IoU). It is used to compare two boundary boxes, ground truth and predicted boxes of an image. IOU stores the properties of an object by comparing its height, width, and bounding box area into the region property and measures a standard area of focus. This feature makes IOU invariant to scale. Due to this valuable property, it is used to evaluate segmentation, object detection, and object tracking which depends on this metric. Huge datasets like MS-COCO [8] and ImageNet [9] try to define bounding boxes of ground truth in a better way. But in some cases, it is not visible clearly. Hence making bounding box labels and calculating regression functions are hard which makes the neural network vulnerable to predict objects precisely.

Object detection is an important task that includes two other processes, object classification and tracking. Some object detectors like Cascade RCNN, Faster RCNN, and Mask R-CNN depends on the regression function to locate the objects by bounding boxes [5, 10, 11]. Then by optimization, duplicate bounding boxes are ignored and the objects are detected based on the highest scores. These networks are slow in predictions due to their complex pipelines. Usually, object detection networks highly depend on its Feature extraction modules. The feature extraction modules are often termed as backbone networks due to their essential process in object detection. ResNet architecture is used as a base in Faster RCNN which is evolved from R-CNN.

Datasets also play a vital role in making the network to be highly efficient. If we feed a large number of images from a dataset into a neural network its learning capability increases. Labeling, making annotations, and calculating bounding boxes for the training set of a dataset is a cost-effective and time-consuming process. When training with CPU, computation time takes much longer as it could even take months for computing large datasets. Due to the capabilities such as GPU and TPU, the training process of deep neural networks takes minimum time when compared with CPU computing.

For real-world applications of autonomous driving, the detections must be very accurate at the same time it needs to be quick but existing networks are so much helpful for this task. This shows that the previous techniques are not up to the mark in accuracy and speed of detection which limits the Self-driving vehicles to come into existence. Object detection algorithms are mainly focused on enhancing autonomous vehicle development. So, the detection of objects should be accurate and fast.

2 RELATED WORKS

In recent years there are numerous researches are performed to maximize the accuracy of Object detection. It is much handy in designing vehicle detection frameworks. The networks and algorithms which are notable for vehicle detection are listed here.

Object detection algorithms process the image to locate and classify them, then it draws bounding boxes with labels depending on the confidence score. The two main approaches of object detections are based on region proposals and then based on regression and classification. The methods which belong to the proposal of regions are R-CNN [3], Fast R-CNN [4], Faster R-CNN [5], FPN [12], SPP-net [13], and Mask R-CNN [11]. The second method which belongs to regression and classification based networks are YOLO [14], YOLOv2 [15], DSSD [17], SSD [18], YOLOv3[16]. These two variants of the object detection network, one stage detection network, and two-stage detection networks improve object detection accuracy and speeds year by year. R-CNN Framework and SPP-Net [13] generates the Region Proposal for feature map and it needs a one-time computer computing. Region proposal is used by Faster R-CNN [5] instead of selective search as an alternative, this improves the End-to-End accuracy and speed. R-FCN [6] reduces computer computing time with a score map of sensitive positioning. These networks are two-stage detection networks that are accurate but extremely slow in detection. In place of region proposal technique YOLO and SSD performs regression of bounding boxes and classification of objects [14, 17]. They consist of a detection model followed by a feature extraction network. YOLO is the trendsetter in detecting objects very fast when compared with other networks. It detects objects in a single evaluation using class probabilities and predicting bounding boxes, the entire network architecture is simple that they are made up of many convolutional networks. YOLO detects objects faster because it does all the works at the same time, its prediction is based on the problem solving of regression instead of classification. YOLO adopts one feed-forward CNN to detect object location in the frame and also records information about the class, hence the process is faster.

As an alternative to the region proposal approach in two-stage detectors, YOLO predicts objects by distinguishing all frames into grids [14]. The output bounding box coordinates are calculated from the YOLO output layer feature maps; class scores and the objectless score, therefore YOLO identifies more objects in one speculation. Hence, the observation speed is very faster than the standard techniques. Although the method is effective by figuring out each grid unit in the image makes more localization errors, and then the accuracy is very low in object detection, and hence it is inappropriate for Artificial intelligent driving applications. To resolve those issues, YOLOv2 has been projected [15]. YOLOv2 progresses the accuracy in recognizing objects than YOLO utilizing convolutional layer batch normalization and draw a boundary box, fine-grained attributes, and multi-scale training but the accuracy in identifying objects is low for occluded objects.

The notable study related to recent improvements in the Object detection networks includes YOLOv3[16], RetinaNet[18], EfficientNet[19] and CornerNet[20].

YOLOv3 is upgraded from YOLOv2 with some modifications. The anchor boxes are defined by a clustering algorithm, then using these bounding boxes are predicted [16]. It implements logistic regression instead of softmax for detecting objects using the objectness score of bounding boxes. The notable feature of YOLOv3 is it uses multiple

scales of prediction layers. It acquired a good accuracy score on the COCO dataset with a higher frame rate in detection.

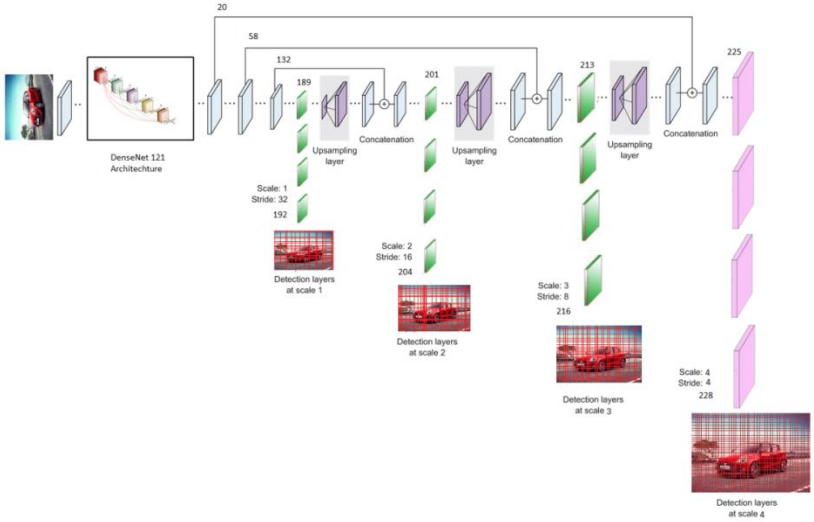
RetinaNet deals with the issue with single-step detection accuracy over two-step algorithms [18]. This issue is based on with relation of class in the foreground and background during the detector training phase. Retina networks achieve higher accuracy by mainly focusing its detector training with focal loss. Scaling of a neural network is an important and efficient task in achieving a maximum level of accuracy. In EfficientNet [19] a new advanced scaling mechanism is present to scale the depth, width, and resolution of the networks.

CornerNet is a new advanced algorithm in object detection. It eliminates previous methods of anchor boxes technique for the detection of objects [20]. It proposes an approach of detecting objects as key points instead of matching anchors with bounding box over them. It also enhances the corner points localization by a new type of pooling called corner pooling.

3 PROPOSED WORK

3.1 YOLOv3 Improvement

We choose YOLOv3 because of its quick and real-time predictions. YOLOv3 is fast but not so accurate in predictions. The YOLO algorithms are mainly designed for object detection process with simple network architectures. Redmon et al [14] proposed YOLO in 2015 which predicts objects quickly by processing the image in one flow, the network uses GoogleNet as the base. In the YOLOv2 custom baseline network called darknet19 is used. YOLOv3 gets a few upgrades such as darknet53 backbone and detection layer scales [15, 16]. The notable feature of YOLOv3 is, it makes three different scales of predictions. The feature maps consist of three detection kernel size of 1x1 with variable sizes and in various places. We mainly focus on improving the network by replacing its backbone structure, then adding Distance IOU loss function [21] and by optimizing YOLO detection layers by implementing an extra scale to propose more anchor boxes and predictions than the original YOLOv3 network structure. The entire proposed architecture diagram is illustrated in Fig. 1.



YOLOv3 determines the anchor boxes using the k-means clustering algorithm. It uses 9 Anchor boxes, one Anchors for every prediction scale layers. The anchors are arranged from larger dimensions to smaller ones. Thus, the foremost scale predicts the larger objects. YOLOv3 struggles with much smaller objects thus it fails to locate vehicles from longer distances to overcome these issues we added an extra scale with three smaller dimensional anchors.

3.2 DenseNet-121 as Backbone Network

The robust vehicle detector must have a good baseline network, a neural network needs to extract better and more features. The commonly used backbone networks are ResNet50, ResNet101, GoogleNet , Vgg16, and Darknet53 [1, 2, 16, 22]. These networks have the best benchmark performance in COCO, VOC, and ImageNet Datasets. As for the detection of vehicles we want to have a feature extractor to be deep at the same time it should not be complex. Hence, the proposed work is designed on DenseNet121[23] a pretrained convolutional neural network as a backbone. The

layers are interconnected by each layer in a multi layered perceptron model. It is pretrained on ImageNet Dataset [9], which contains weights of different features of Images in the dataset. It has a depth of 121 convolutional layers. The advantages of DenseNet incudes, they reduce the issues on the vanishing-gradient, strong distribution of features, reuse of features, and less number of parameters.

Let's consider if an image x_0 is passed into a neural network which has L layers with non-linear transformation $H_l(.)$, then *l* is the index of the layer. ResNet the classic feed-forward network, adds a skip connection that bypasses the non-linear transformation function its equation (1) is given below [1].

$$x_l = H_l(x_l - 1) + x_{l-1} \quad (1)$$

In dense network to increase the information in the layer, it uses direct end to end connections. The l^{th} gets the information of all the preceding layers

$$x_l = H_l[(x_0, x_1, \ldots\ldots, x_l - 1)] \quad (2)$$

In Densenets the downsampling is done at Dense Blocks these blocks are divided by Transition layers, The Transition layer consists of 1×1 convolutional layer, an average pooling layer with batch normalization. The weights present in the transition layer also spread on the Dense layers. For ease of network utility, we converted all of the average pooling layer into 2×2 max pool layer. The model looks less complex by the arrangement of Batch normalization is done before each of the convolutional layers. The growth rate of the network is denoted by hyperparameter k which makes the DenseNet powerful enough to get state of the art results.

The final classification layer of fully connected and pooling layers are removed to connect the proposed detection layers for detection. There are much deeper network arrangements are also featured than 121 layer network such as DenseNet-169, DenseNet-201, and DenseNet-264 [23]. We don't want to make our net so much deeper hence 121 layer arrangement is perfect to make our vehicle detection.

Table 1 DENSENET 121

	Type	Filters	Size	Output
	Convolution	64	7 × 7 / 2	112 × 112
	Pooling		3 × 3 max pool, stride 2	56 × 56
6 ×	Dense Block (1)	128 32	1 × 1 conv 1 × 1 conv	56 × 56
	Transition Layer (1)	128	1 × 1 conv 2 × 2 average pool	56 × 56 28 × 28
12 ×	Dense Block (2)	128 32	1 × 1 conv 1 × 1 conv	28 × 28
	Transition Layer (2)	256	1 × 1 conv 2 × 2 average pool	28 × 28 14 × 14
24 ×	Dense Block (3)	128 32	1 × 1 conv 1 × 1 conv	14 × 14

	Transition Layer (3)	64	1 × 1 conv 2 × 2 average pool	14 × 14 7 × 7
16 ×	Dense Block (4)	128 32	1 × 1 conv 1 × 1 conv	14 × 14
	Classification Layer		7 × 7 global Avg pool 1000D fully connected softmax	1×1

For training the detector the weights from feature extractor are necessary. Apart from the baseline structure during training, we also used pretrained weights of DenseNet trained from the ImageNet dataset. The input image size of the extractor is 416 × 416 after the images are scaled to this size by the input layer of the network, these images are passed into the first layer of convolution with 7 × 7 sizes with stride 2 and filter 64. Then a max-pooling is applied with size 3 × 3. After the image is passed into pooling its features are extracted by the following Dense Blocks and Transition layers. These extracted features are passed into the detection model of our network. The DenseNet-121 filter sizes, layers are listed in Table 1. This feature Extractor has strong gradient flow, computational and parameter performance advantage, and more diversified feature flow across the layers.

4 ARCHITECTURE OVERVIEW

The proposed Network structure consists of the Input layer, feature extraction network, and a detection network. The networks foremost step is to scale the dataset input image size as 416 × 416 then, these scaled images are fed into DenseNet-121 for Feature Extraction. As we already stated, we replaced YOLOv3 baseline network Darknet53 with DenseNet121 and its workflows, now we look into the modifications on the detection part of the network.

4.1 Detection Network

In the detection part, the extracted features of Densenet-121 and its pretrained weights are used to train the robust vehicle detector. The dataset images are transformed into feature maps of size 13 × 13 from the feature extractor then it is passed into detection layers of YOLO followed by few convolutions. We reduced the special dimensions from 14 × 14 from final dense blocks into 13 × 13 to match the detection layers for efficient results. Four feature maps are formed from combining three feature maps of size 26 × 26, 52 × 52, and 104 × 104. Detection is performed on four scales with the upsampling of three feature maps transmitted from the parallel scales of the network.

4.2 Prediction Layers

The four prediction layers of the network present on 192, 203, 216, and 228th layers. The first detection is done by the 93rd layer before that images are downsampled by the network forming a feature map of 13 x 13 x 75 for 1 x 1 detection kernel with stride 32, if the image input is 416 x 416. This feature map is depth concatenated with the 20th layer feature map.

Then for the second detection in the 205th layer, the images before that layers are upsampled few convolutional layers are added which delivers a feature map of 26 x 26 x 75 with stride 16. In this section, depth concatenation is made with the 58th layer. Similarly for the third and fourth detection scale with the stride of 8, 4 forming feature maps of 52 x 52 x 75 and 104 x 104 x 75. For the third detection layer, layer 192 depth concatenation is preformed with the 132nd layer. The total layers in our network are 228 with BFLOPS 58.336 of operations.

The detection kernel is defined by 1 x 1 x (Bboxes x (5+Classes)). The number of bounding boxes a cell in the feature map can predict is denoted as Bboxes. Bounding box attribute is 4 and 1 Objectness score is 5. In YOLOv3 network according to Pascal VOC dataset Bboxes=3, Classes=20 , so the kernel size is 1 x 1 x 75. This feature map is calculated on VOC dataset with 20 classes on it. For detecting vehicles, we don't need other classes except for bus, car from the VOC dataset.

4.3 Anchor Boxes:

The accuracy of detecting vehicles depends on the prediction grid of the Anchor boxes by the scaling layers. The Anchor boxes are predefined metrics. YOLOv3 defined its Anchor boxes by k-means clustering for COCO and VOC

datasets. Clustering anchors are better than handpicked set of anchors. The k-means clustering can be termed as in equation (3) [16],

$$E = \sum_{j=1}^{k} \sum_{i=1}^{n} \left\| x_i^{(j)} - c_j \right\|^2 \quad (3)$$

Here, the image is x, the average vector of Ci is denoted as μi and the cluster center is termed as k. The Anchor boxes are calculated by setting various values for k. we ran the clusters only to choose an extra three anchors for the network. We used the same set of Anchor boxes for the 3 prediction layer as it is. We choose another 3 anchors for the 4th prediction layer with smaller dimensions for long-range vehicles. The four prediction scale with 3 anchors each are (5 × 11), (5 × 23), (13 × 18), (10 × 13),(16 × 30), (33 × 23), (30 × 61), (62 × 45), (59 × 119), (116 × 90), (156 × 198) and (373 × 326). YOLOv3 predicts 10,647 bounding boxes per image, In Improved YOLOv3 network, with an extra scale we can predict 43,095 bounding boxes with an image input size of 416 x 416. Thus the network predicts bounding boxes 4 times increase in number than YOLOv3. This will increase the probability of predicting smaller vehicles.

4.4 Loss Function

The most important step in vehicle detection is Bounding box regression. Though Ln-norm loss is mostly used for bounding box regression which is not related to use with Intersection over Union (IoU). Some works related to improving IoU metrics are IoU loss [24], generalized IoU (GIoU) [25], and Distance-IoU (DIoU) [21]. In our work, we choose to implement DIoU as it's proven to be effective than others. In the target box and predicted box, the normalized distance is implemented which is quick in the training period. It deals with some of the geometric factors too using Complete IoU (CIoU) loss is implemention [21]. The Distance-IoU is based on bounding box center point coordinates. It is based on Ln-norm and IoU loss with few modifications in it as defined in equation (4).

$$\int_{\mathrm{DIoU}} = 1 - \mathrm{IoU} + \frac{\rho^2(\mathrm{b},\mathrm{b}^{\mathrm{gt}})}{\mathrm{c}^2} \quad (4)$$

The equation for DIoU is shown above, b is the predicted bounding box center point and b^{gt} is the ground truth center point, the penalty term is used to minimize the distance between these two. The diagonal length of a small box covering these two boxes is denoted as denominator c.

It is invariant to scale and also must solve a few other issues wit overlap area, central point distance, and aspect ratio. These terms are addressed by the CIoU loss function its equation is given in equation (5) below [21],

$$\int_{\mathrm{CIoU}} = 1 - \mathrm{IoU} + \frac{\rho^2(\mathrm{b},\mathrm{b}^{\mathrm{gt}})}{\mathrm{c}^2} + \alpha \mathrm{v} \quad (5)$$

4.5 Datasets

We used PASCAL VOC 2007, 2012 and COCO dataset [8, 26]. Pascal VOC contains 20 different classes, a total of 9963 images. For vehicle detection, we need only car and bus classes from this dataset. At first, we trained with all 20 classes to record the mAP score of the Improved YOLOv3-Net, our main approach is to detect only vehicles. From the COCO dataset, we picked three classes car, bus, and truck then calculated its Average precisions. We also manually created 4500 labeled frames of vehicle images taken from traffic CCTV footage in day and night conditions. These videos are collected from different online sources of live CCTV traffic camera videos which contains images of car, bus, and Truck. GRAM Road-Traffic Monitoring (GRAM-RTM) dataset [35] images are used to evaluate the detection capability of the Improved network.

Table 2 Experimental Platform Configuration

Computing Machine	Configuration
Operating System	Ubuntu 18.04.3 LTS
GPU	NVIDIA T4 Tensor Core GPU
RAM	16
GPU acceleration library	CUDA10.0, CUDNN7.4

5 TRAINING AND TESTING OF NETWORK

We have trained the network on numerous methods. First, we trained the 418 × 418 network on the PASCAL VOC dataset with 20 classes, After evaluating its results we removed all other classes from Pascal VOC except car and bus and started training along with CCTV dataset. Then we trained the detector with vehicle class of COCO dataset and CCTV video frames we have collected. The CCTV dataset video frames are manually labeled. These frames significantly improve our detection accuracy in dark areas as we included night time vehicle images. We also implemented an input image size of 608 × 608 model of the network, which is more effective than 418 × 418 model, mainly due to pixel differences and the features associated with it. Our Experimental platform configuration is listed in Table 2. The entire experimental process is explained in the following sections.

5.1 Data augmentation:

Data Augmentation is used to train networks with a maximum amount of data without increasing the dataset. In our work we used flipping, cropping, and blur images. We also implemented data transform function to transform datasets into network readable format.

5.2 Optimization:

In our network, a widely used optimization algorithm SGD stochastic gradient descent algorithm is present to optimize the vehicle detection network. The parameters for the convolutional and pooling layers are carefully initiated. The learning rate of our model is 0.001 for 40k iterations in training each model. We used a batch size of 32 in 416 × 416 image input network and mini-batch of value 16. We placed a weight decay of 0.0005 and a momentum of 0.9. This configuration is set for the first method in the training process. Different approaches and training options also carried out to find the best detection accuracy.

TABLE 3 Comparison of various Network models on PASCAL VOC 2007. This table is taken from YOLO9000 paper [15].

Pascal VOC 2007	**mAP**
Fast R-CNN [4]	70.0
Faster R-CNN VGG-16 [5]	73.2
Faster R-CNN ResNet [1]	76.4
YOLO [14]	63.4
SSD300 [17]	74.3
SSD500 [17]	76.8
YOLOv2 [15]	76.8
Improved YOLOv3-Net (ours)	**77.4**

In Table 3, we have calculated the mAP of all the 20 classes in the PASCAL VOC dataset for Improved YOLOv3-Net. The training took approximately 1 hour for 1000 iterations. We have made 40k to 45k iterations. Improved YOLOv3-Net performs better when compared with one stage and two stage detectors. We get 77.4 % mAP on Pascal VOC, nearly 2% mAP rise than YOLOv2 and also outperform other detectors which are shown in Fig 2. Although we get better results this training with all classes is not an ideal choice for vehicle detection problems. We don't want classes like bird, dog, persons, and other unwanted classes for vehicle detection. There is no need for a vehicle detector to learn about these things. So we tried different approaches of training network with few vehicles classes car and bus from the Pascal VOC COCO and CCTV dataset with few parameter tweaks.

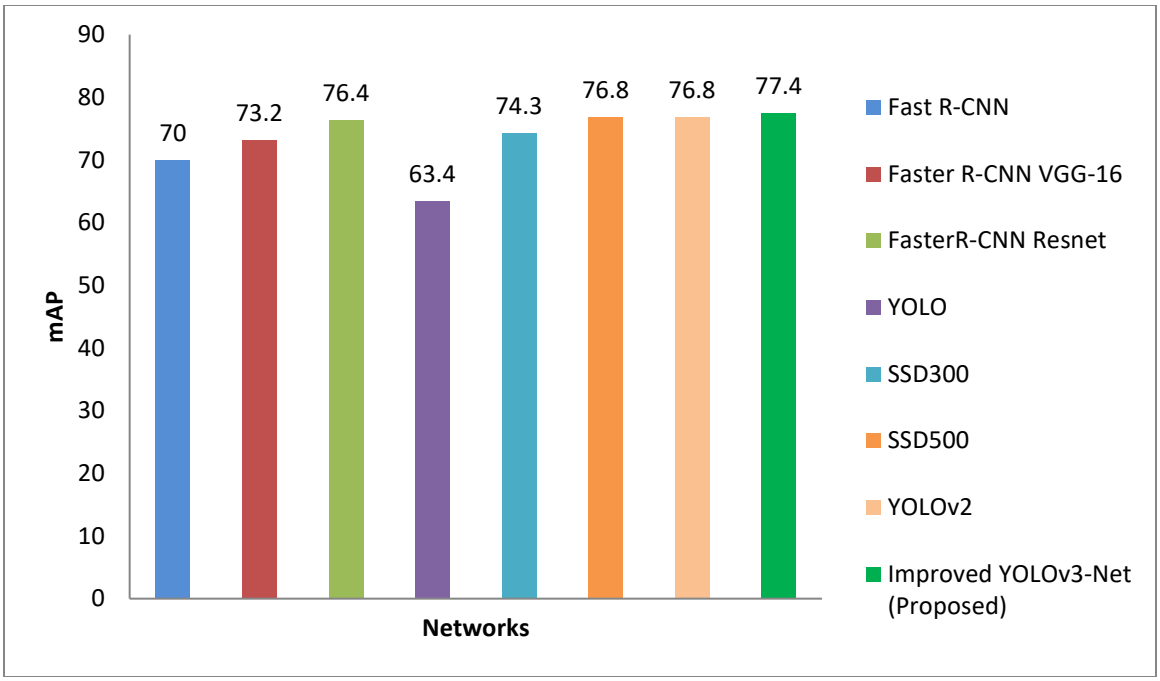


Fig 2 Graph showing the comparison between the proposed network with different networks

To get the best results for object detection, datasets plays a crucial role. Designing a network and make it up running is easy. The hard thing is collecting data to train the built network. Numerous public datasets are available but it is not designed to address our needs to detect vehicles. To detect vehicles precisely, we required more data so we created our own set of data to boost accuracy. We trained our network on PASCAL VOC with CCTV video frames, then COCO data with CCTV video frames, and combining both datasets with CCTV footage frames. We trained our network with a limited number of classes includes car bus and truck. In table 4, Improved YOLOv3-Net is compared with different datasets and with two input sizes models 416 × 416 and 608 × 608 are used. When input image size increases the performance of the network increases, due to pixel size and more features from the images are transferred across the layers.

TABLE 4 Comparison of Proposed Network with YOLOv3 and YOLOv2 trained on Pascal VOC and CCTV frames dataset

NETWORK MODEL	BACKBONE NETWORK	DATASET	AP (car)	AP (bus)	AP (Truck)	mAP
YOLOv2-416	Darknet19	Pascal VOC + CCTV frames	78.9	67.3	72.3	72.8
YOLOv3-416	Darknet53	Pascal VOC + CCTV frames	69.4	78.6	77.3	75.1
Improved YOLOv3-Net-416 (ours)	**DenseNet-121**	Pascal VOC + CCTV frames	79.6	74.3	82.4	78.7
Improved YOLOv3-Net-608 (ours)	**DenseNet-121**	Pascal VOC + CCTV frames	86.2	79.8	77.5	81.1

In the first approach of vehicle detection training, Pascal VOC 2007, 2012, and CCTV video frames are used to train the detector. The classes included are bus, car, and truck. There are about 30k to 35k training iterations that were observed. The batch size of this training process is 64, the subdivision batch is set as 16, and the learning rate as 0.001. The IoU threshold of 0.50 is applied in the dataset to find out the True positive and False positive. Two different sizes of network models are observed Improved YOLOv3-Net-416 and Improved YOLOv3-Net-608. The best mAP of

78.1% for YOLOv3-Net-416 and 81.1% for YOLOv3-Net-608 was achieved by the modified network which is listed in Table 4.

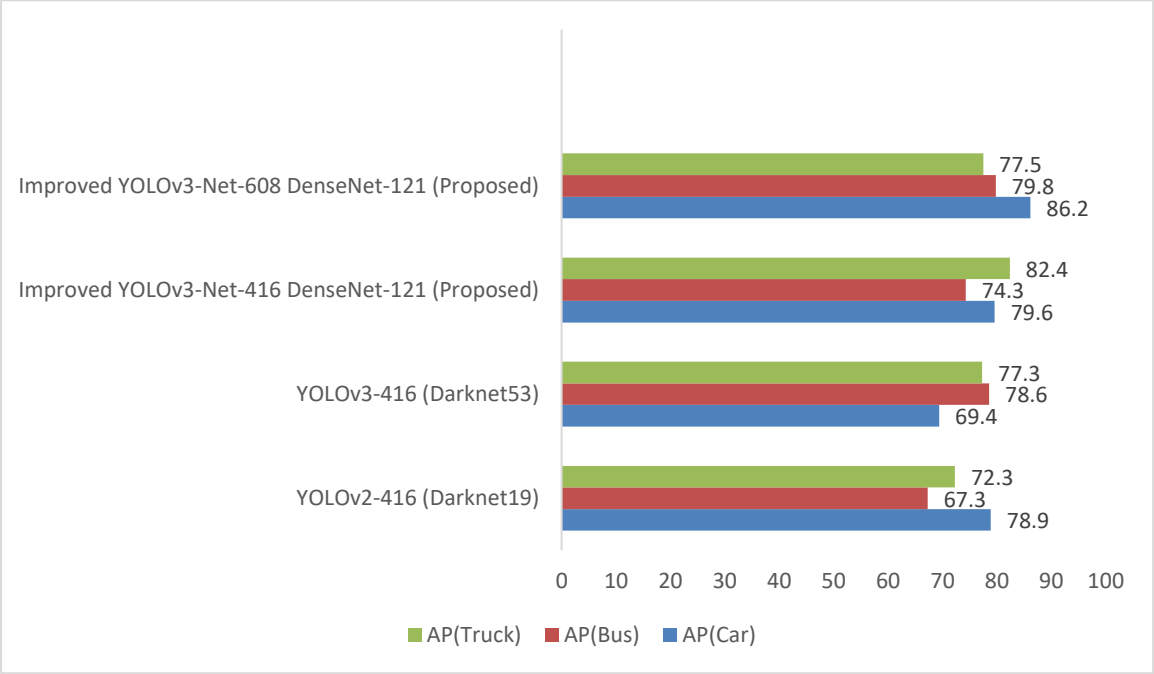


Fig 3 Comparison Graph of Improved YOLOv3-Net Average Precisions for Car, Bus and Truck Classes in PASCAL VOC and CCTV video frames Dataset

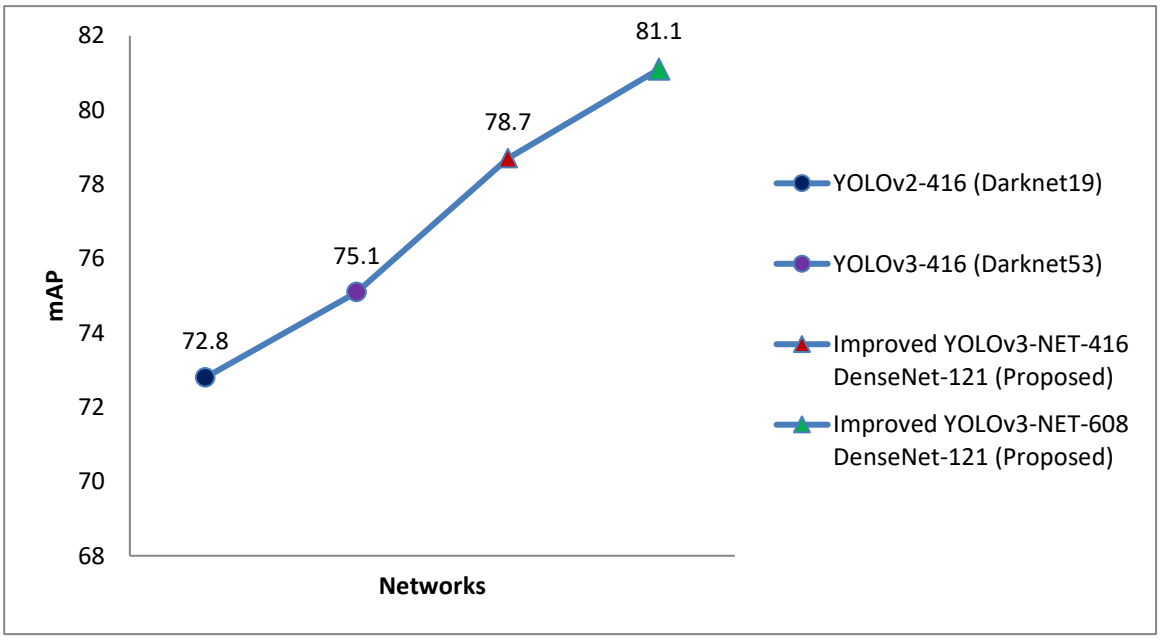


Fig 4 Comparison Graph of Improved YOLOv3-Net mAP with YOLOv2 and YOLOv3 in PASCAL VOC and CCTV video frames Dataset

In order to compare the effectiveness of our model, we used YOLOv2 and YOLOv3 networks both achieves less accuracy than our modified model which is listed in Table 4. The YOLOv2 model has a backbone of Darknet19 and

YOLOv3 has Darknet53 in our modified model Densenet-121 is used which is useful in extracting more features than Darknet architecture. Comparison graphs for Average and mean Average Precision for Proposed network with YOLOv2 and YOLOv3 is shown in Fig 3 and Fig 4.

TABLE 5 Comparison of Proposed Network with YOLOv3 and YOLOv2 trained on COCO and CCTV frames dataset

NETWORK MODEL	BACKBONE NETWORK	DATASET	AP (car)	AP (bus)	AP (Truck)	mAP
YOLOv2-416	Darknet19	COCO + CCTV frames	68.8	61.4	50.7	60.3
YOLOv3-416	Darknet53	COCO + CCTV frames	72.3	59.7	66.2	66
Improved YOLOv3-Net-416 (ours)	**DenseNet-121**	COCO + CCTV frames	76.6	65.4	72.8	71.6
Improved YOLOv3-Net-608 (ours)	**DenseNet-121**	COCO + CCTV frames	78.2	73.4	68.7	73.4

In the second approach, we performed the training process with the COCO dataset and CCTV video frames with the same three different vehicle classes car, bus, and Truck. The batch size for training the network is 80, the mini batch is 32, and the learning rate as 0.001. we set the weight decay of 0.0005 and tried some downsampling earlier than the Pascal VOC training process. The training process includes 40k to 50k iterations. The mAP results are quite low when compared with the Pascal VOC dataset due to some loss in the training process. We also used DIoU function to increase accuracy to some extent. The mAP of 73.4% is achieved by Improved YOLOv3-Net-608, along with all other compared networks mAP are listed in Table 5 and its corresponding graphs are plotted in Fig 5 and Fig 6.

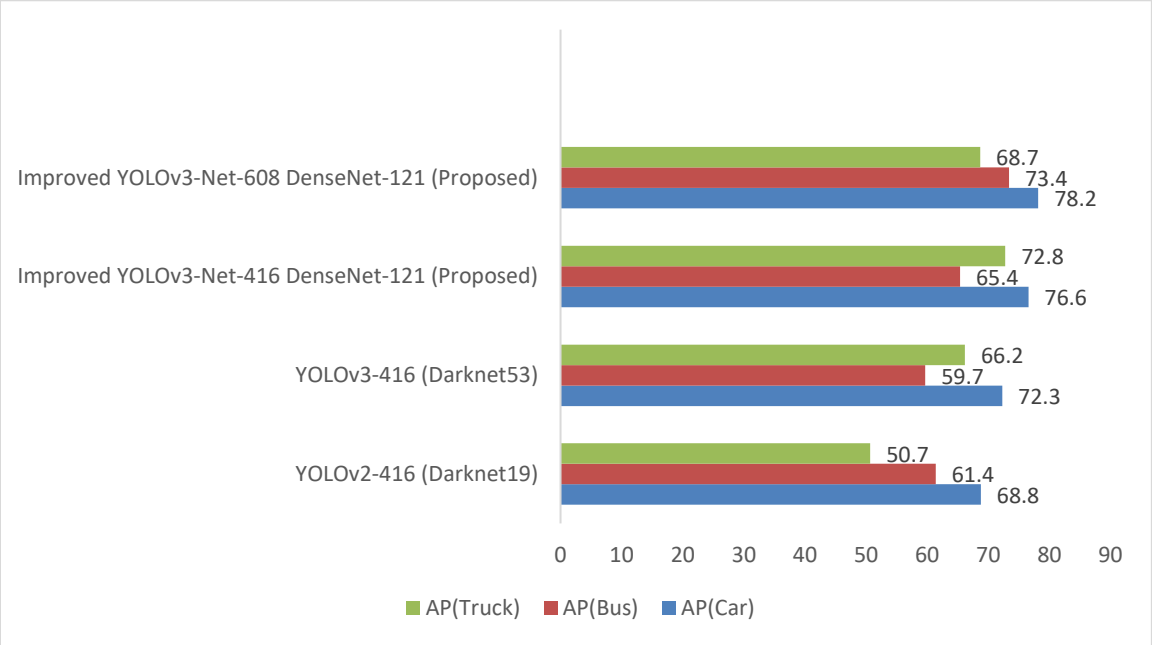


Fig 5 Comparison Graph of Improved YOLOv3-Net Average Precisions for Car, Bus and Truck Classes in COCO and CCTV video frames Dataset

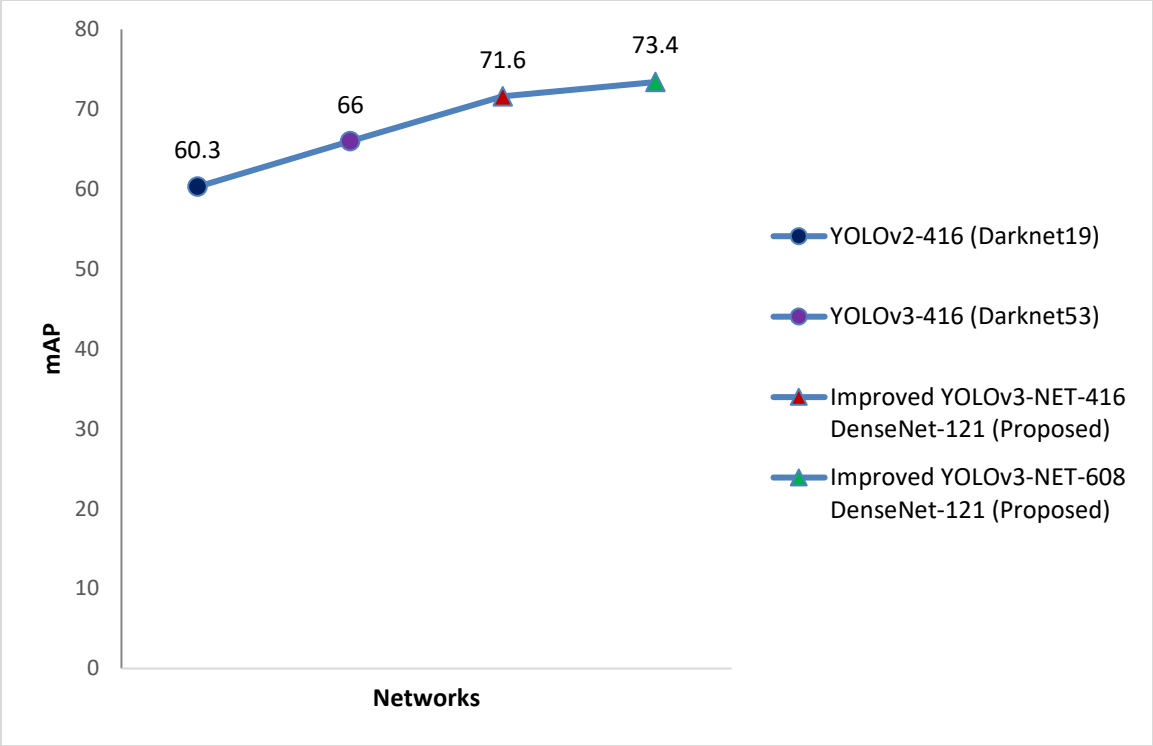


Fig 6 Comparison Graph of Improved YOLOv3-Net mAP with YOLOv2 and YOLOv3 in COCO and CCTV video frames Dataset

TABLE 6 Comparison of Proposed Network with YOLOv3 and YOLOv2 trained on COCO and CCTV frames dataset

Network	Backbone	Dataset	AP (car)	AP (bus)	AP (Truck)	mAP
YOLOv2-416	Darknet19	Pascal VOC+ COCO + CCTV frames	72.5	78.2	75.7	75.4
YOLOv3-416	Darknet53	Pascal VOC+ COCO + CCTV frames	78.3	75.8	79.4	77.8
Improved YOLOv3-Net-416 (ours)	**DenseNet-121**	Pascal VOC+ COCO + CCTV frames	84.6	78.2	76.9	79.9
Improved YOLOv3-Net-608 (ours)	**DenseNet-121**	Pascal VOC + CCTV frames	88.5	83.1	76.8	82.8

In the final network model, we trained the networks with all the datasets which are mentioned above. By combining these datasets during training the network's learning capability increases due to the availability of large numbers of features. In training, we used 40k to 55k iterations with the same batch size and other parameters from the last training process. The training of 608 × 608 networks took approximately 2 hours for 1000 iterations, a total of about 40k to 50k iteration weights were analyzed. It is trained on three different datasets with vehicle classes alone. We have to stop and use our backup weights for training again for several times to analyze the best weights for high accuracy. The network size is larger than the original YOLOv3, hence training time took a bit longer. From Table 6 we can find that Improved YOLOv3-Net mAP looks promising. It achieves mAP of 82.8% on a combination of all 3 datasets and in Fig 7 and Fig 8, illustrates its mAP graphs. The network predicts many bounding boxes for a single vehicle, to cancel out extra boxes from their objectness score we used thresholding of 0.2 to 0.5. Any boxes below this threshold

are ignored by the network. In Fig 9, the detection capability of Improved YOLOv3 on GRAM Road-Traffic Monitoring (GRAM-RTM) dataset is shown.

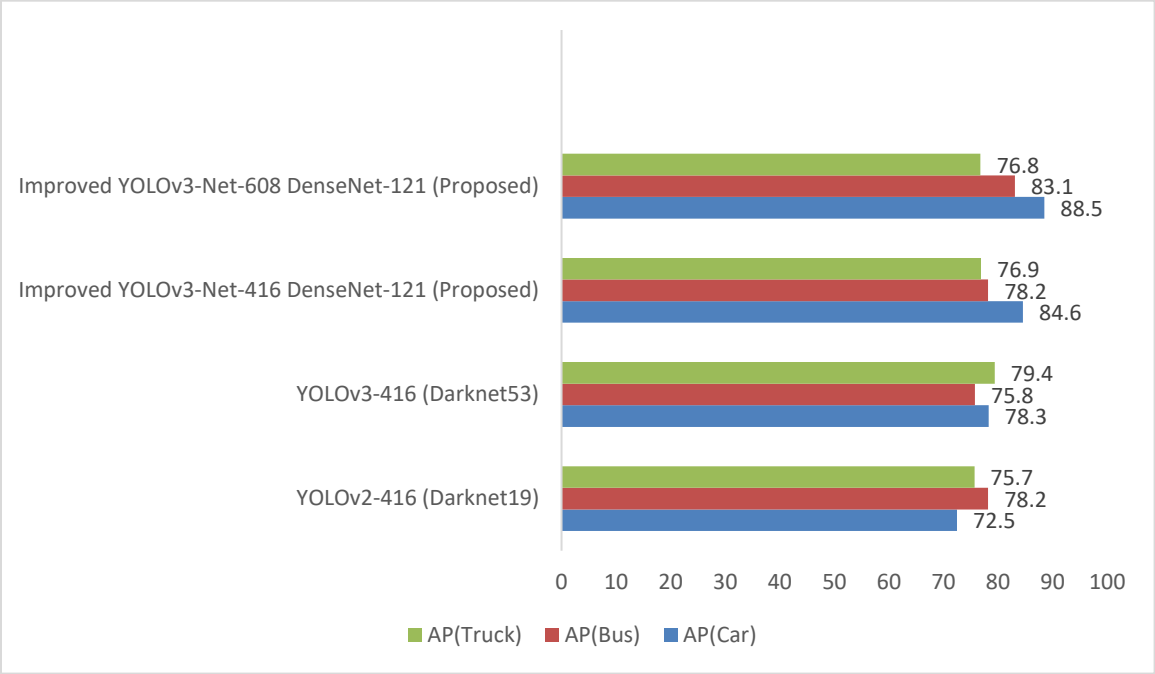


Fig 7 Comparison Graph of Improved YOLOv3-Net Average Precisions for Car, Bus and Truck Classes in PASCAL VOC, COCO and CCTV video frames Dataset

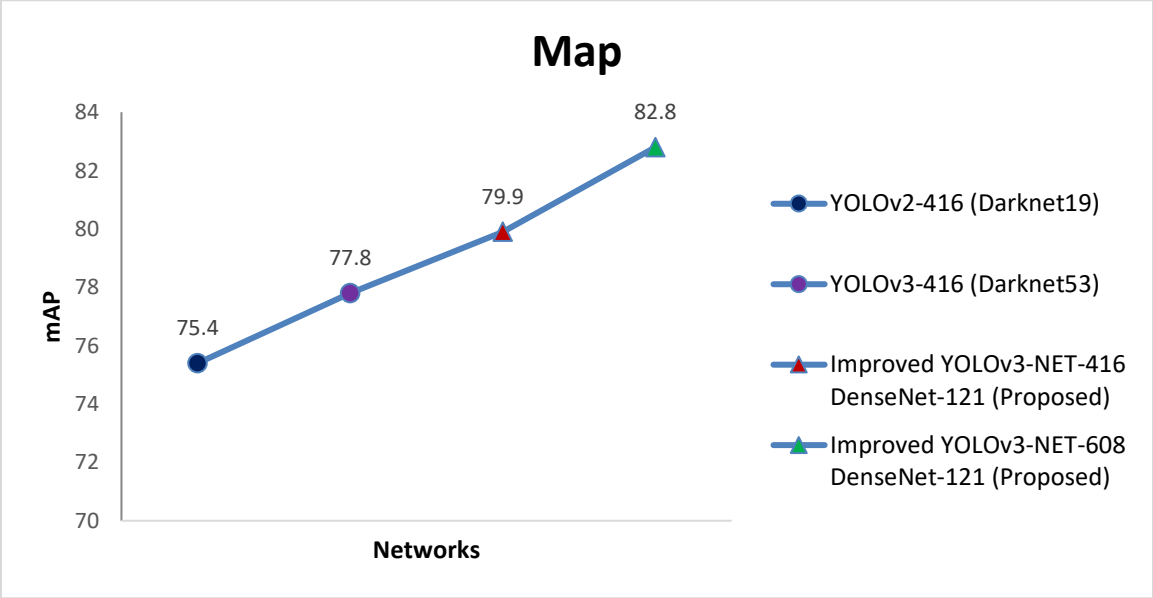
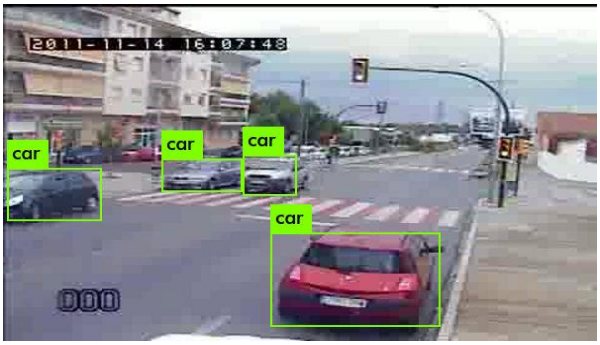


Fig 8 Comparison Graph of Improved YOLOv3-Net mAP with YOLOv2 and YOLOv3 in PASCAL VOC, COCO and CCTV video frames Dataset

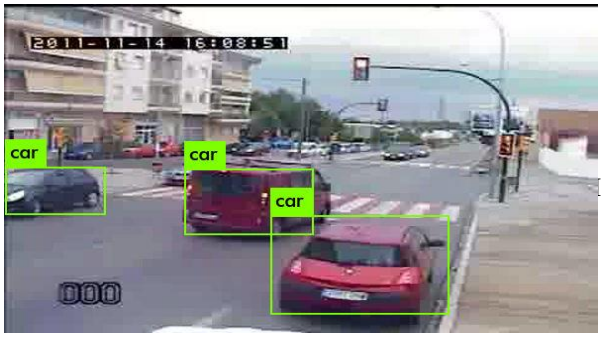
5.3 Post processing by Non-maximum Suppression

This step is used to solve the problem of predicting multiple bounding boxes for the detected vehicles. If a cell predicts a bounding box some adjacent cell block also predicts overlapping boxes to it. The overlap of bounding boxes is

rectified using NMS method. We also adopted DIoU NMS algorithm [21] into our network with a threshold of 0.4 to 0.5. It is proven to be effective and is easy to implement which further increases our vehicle detection function significantly.



5.4



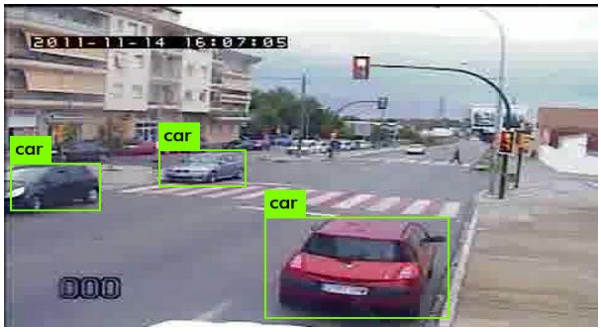


Fig 10 Detection of vehicles in a CCTV video. Right – A small car is undetectable by the detector

In Fig 10, A small vehicle in distance is correctly detected by the improved network as a car. These errors mostly depend on pixel ratio. If an object is not occupied by certain pixels it is not detected by the detector. Most of the error occurs due to the poor quality of the video feed. To minimize this kind of errors the network requires more optimizing techniques.

6 Conclusion and Future works

In this research work, a new Vehicle detection algorithm is proposed using deep neural network based on YOLOv3. The proposed method uses DenseNet-121 as a Feature Extraction network replacing darknet53. We also adopted loss function Distance IOU and NMS-DIOU into the network to increase accuracy. Our network predicts objects on four different scales. The network achieves a higher mAP of 82.8% on PASCAL VOC and COCO dataset with CCTV night frames of vehicles. The proposed network is highly efficient in detection of vehicles regardless of numerous conditions.

From different datasets, we find that our detector is effective when training with CCTV video frames along with datasets of Pascal VOC and COCO. The detector is quick as it predicts a single image in less than 30ms. Even though this network achieves ideal results, it lacks in predicting few vehicles such as small and dark natured vehicle types. In

the future works, we want to make a dataset especially based on small size long-distance vehicle images to detect more long-range vehicles. In this network, the prediction of vehicles is very low when tested on extreme darkness. So, we want to mainly focus on these two issues.

References

1. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep Residual Learning for Image Recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 26 June–1 July 2016.
2. Simonyan, K.; Zisserman, A. Very Deep Convolutional Networks for Large-Scale Image Recognition. In Proceedings of the International Conference on Learning Representations, San Diego, CA, USA,7–9 May 2015.
3. Girshick, R.; Donahue, J.; Darrell, T.; Malik, J. Rich feature hierarchies for accurate object detection and semantic segmentation. In Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition, Columbus, OH, USA, 24–27 June 2014; pp. 580–587.
4. Girshick, R. Fast R-CNN. In Proceedings of the 2015 IEEE International Conference on Computer Vision(ICCV), Santiago, Chile, 7–13 December 2015; pp. 1440–1448.
5. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. arXiv 2016, arXiv:1506.01497v3.
6. Dai, J.; Li, Y.; He, K.; Sun, J. R-FCN: Object detection via region-based fully convolutional networks. In Proceedings of the 2016 IEEE International Conference of Advances in neural information processing systems, Barcelona, Spain, 5–8 December 2016; pp. 379–387.
7. Cai, Z.; Fan, Q.; Feris, R.S.; Vasconcelos, N. A unified multi-scale deep convolutional neural network for fast object detection. In Proceedings of the 2016 European Conference on Computer Vision, Amsterdam, The Netherlands, 8–16 October 2016; pp. 354–370.
8. Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays,Pietro Perona, Deva Ramanan, Piotr Doll´ar, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In European conference on computer vision, pages 740–755.Springer, Cham, 2014.
9. Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li,and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on, pages 248–255.IEEE, 2009.
10. Zhaowei Cai and Nuno Vasconcelos. Cascade r-cnn: Delvinginto high quality object detection. arXiv preprintar Xiv:1712.00726, 2017.
11. K. He, G. Gkioxari, P. Doll´ar, and R. Girshick. Mask r-cnn. In Computer Vision (ICCV), 2017 IEEE International Conference on, pages 2980–2988. IEEE, 2017.
12. T.-Y. Lin, P. Dollar, R. Girshick, K. He, B. Hariharan, and ´ S. Belongie. Feature pyramid networks for object detection. In CVPR, 2017.
13. K. He, X. Zhang, S. Ren, and J. Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. In ECCV. 2014.
14. Joseph Redmon, Santosh Divvala, Ross Girshick, and AliFarhadi. You only look once: Unified, real-time object detection. In Proceedings of the IEEE conference on computervision and pattern recognition, pages 779–788, 2016.
15. Redmon, J.; Farhadi, A. YOLO9000: better, faster, stronger. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Venice, Italy, 22–29 October 2017; pp. 7263–7271.
16. Redmon, J.; Farhadi, A. YOLOv3: An incremental improvement. arXiv 2018, arXiv:1804.02767 .
17. Wei Liu, Dragomir Anguelov, Dumitru Erhan, ChristianSzegedy, Scott Reed, Cheng-Yang Fu, and Alexander CBerg. SSD: Single shot multibox detector. In European conferenceon computer vision, pages 21–37. Springer, 2016.
18. T. Lin, P. Goyal, R. B. Girshick, K. He, and P. Dollár, “Focal loss for dense object detection,” in The IEEE International Conference on Computer Vision (ICCV), 2017.
19. Tan, M., and Le, Q. V. 2019. Efficientnet: Rethinking model scaling for convolutional neural networks. arXiv preprint arXiv:1905.11946.
20. H. Law and J. Deng. Cornernet: Detecting objects as paired keypoints. In ECCV, pages 765–781, 2018.
21. Z. Zheng, P. Wang, W. Liu, J. Li, R. Ye, and D. Ren, “Distance-IoU loss: Faster and better learning for bounding box regression,” in The AAAI Conference on Artificial Intelligence (AAAI), 2020.

22. Szegedy, Christian, Liu, Wei, Jia, Yangqing, Sermanet, Pierre, Reed, Scott, Anguelov, Dragomir, Erhan, Dumitru, Vanhoucke, Vincent, and Rabinovich, Andrew. Going deeper with convolutions. CoRR, abs/1409.4842, 2014.
23. G. Huang, Z. Liu, K. Q. Weinberger, Densely connected convolutional networks, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 4700–4708
24. D. Zhou, J. Fang, X. Song, C. Guan, J. Yin, Y. Dai, and R. Yang, "IOU loss for 2d/3d object detection," in 2019 International Conference on 3D Vision (3DV). IEEE, 2019, pp. 85–94.
25. H. Rezatofighi, N. Tsoi, J. Gwak, A. Sadeghian, I. Reid, and S. Savarese, "Generalized intersection over union: A metric and a loss for bounding box regression," in Proc. IEEE Conf. Computer Vision and Pattern Recognition, 2019, pp. 658–666.
26. M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman. The pascal visual object classes (voc) challenge. International journal of computer vision, 88(2):303– 338, 2010.
27. Guerrero-Gómez-Olmedo, R., López-Sastre, R.J., Maldonado-Bascón, S., Fernández-Caballero, A.: Vehicle tracking by simultaneous detection and viewpoint estimation. In: Ferrández Vicente, J.M., Álvarez Sánchez, J.R., de la Paz López, F., Toledo Moreo, F.J. (eds.) IWINAC 2013, Part II. LNCS, vol. 7931, pp. 306–316. Springer, Heidelberg (2013)