Grenze International Journal of Engineering and Technology, June Issue

Encoder-Decoder Approach toward Vehicle Detection

Pushan Deb[1], Sunny Kumar[2] and Dr Preethi N[3]
[1-3]Department of Data Science, Christ University Lavasa, Pune, India
Email: pushan.deb@msds.christuniversity.in, sunny.kumar@msds.christuniversity.in, preethi.n@christuniversity.in

***Abstract*—Vehicle Detection algorithms run on deep neural networks. But one problem arises, when the vehicle scale keeps on changing then we may get false detection or even sometimes no detection at all, especially when the object size is tiny. Then algorithms like CNN, fast-RCNN, and faster-RCNN have a high probability of missed detection. To tackle this situation YOLOv3 algorithm is being used. In the codec module, a multi-level feature pyramid is added to resolve multi-scale vehicle detection problems. The experiment was carried out with the KITTI dataset and it showed high accuracy in several environments including tiny vehicle objects. YOLOv3 was able to meet the application demand, especially in traffic surveillance Systems.**

***Index Terms*— Surveillance video, vehicle detection, codec, convolutional neural network, YOLOv3, moving object detection and tracking.**

I. INTRODUCTION

Road accidents and crime are increasing day by day. An intelligent Road Monitoring System is becoming the need of the hour. Vehicles need to be identified with a license plate so that further research can be made on a specific vehicle to identify the driver and provide proper evidence to law enforcement.[1] Because of the increasing number of network cameras, locally manufactured visual data, and Netizens, it is difficult yet essential to analyze a large amount of background data at once. Moving object detection (MOD) is a technique for extracting dynamic foreground elements from video frames, such as moving pedestrians or automobiles, and removing the background that isn't moving. Due to the recent success of convolutional neural networks (CNN), there is a great deal of interest in deep learning-based object identification algorithms, and numerous models have achieved cutting-edge results [2]. Particularly opposed to artificially manufactured methodologies, deep learning techniques utilize proposal generation methods like MultiBox, DeepBox, and region proposal networks (RPNs) can provide fewer candidates of superior value.

YOLOv3 Algorithm is a pre-trained model on COCO dataset with over 80 classes that it can detect with a MAP to 69% under sunny weather condition. As YOLOv3 is a pre-trained model, the detection is very fast and can be implemented in live traffic conditions. As shown in fig 1, It can be shown that the distance of the item and the size of the vehicle are inversely related. This may lead to incorrect detection or even faulty detection in some contexts.

This paper improves the YOLOv3 network to address this problem. Given that features like SSD, YOLOv3, and FPN all use feature pyramid structures at the detection stage, this study proposes a novel multi-level feature pyramid structure introduced to the codec module to recognise vehicle targets at various forms. The multilayer characteristics that the backbone network had retrieved were first merged into basic features. The essential

Grenze ID: 01.GIJET.9.2.1

properties listed above are then transmitted to the codec module via its decoder layer as the feature of the detection object. We eventually combine the multi-level properties of the backbone network with equivalent scales at the decoder layer to produce a feature pyramid for target identification. In section II. Related works are explained with the main algorithms. In Section III Proposed methodologies are explained and the various methods that are applied. In Section IV Conclusion and further works are given.

II. Related Works

A. Fully Supervised Object Detection

The fully supervised technique, which employs bounding box annotation for object recognition, may be split into two primary categories: single-stage detectors and multiple-stage detectors. The most popular single-stage object identification technique is called YOLO [1], which uses a Dark net architecture for real-time (30 fps) grid-based object recognition and predicts the centre of the item as well as the width and height of the bounding boxes for each grid cell. The author proposed Draknet19, which, inspired by ResNet [3], offers quicker detection (100 fps) by giving feature concatenation from the preceding layer, for a later version of YOLO, also known as YOLO9000. However, YOLO and YOLOv2 have trouble detecting tiny objects. The author suggested Darknet 53 as a replacement for YOLOv3[4], which is a bit slower (45 fps) than YOLOv2 but significantly more effective at identifying small things since it applies detection to objects on three separate sizes. Similar to YOLO, the single-shot multibox detector (SSD) is a grid-based real-time (59 fps) detection technique.

B. Object Counting

Counting items in a scene that are visible to a model is known as object counting. Numerous techniques for doing item counting using clustering, either detection or regression, are available. Counting methods based on clustering are frequently used to count objects. These methods usually come after an unsupervised learning pipeline that divides features into categories based on an object's appearance. For instance, Tu et al.[13] employed expectation-maximization to count persons based on their shoulders and faces. Rabaud and Belongie published a counting method based on the observation of feature points over time. Since they often need a video sequence to do so, clustering-based algorithms find it difficult to follow feature points in the still picture.

C. The Combined Loss Function

The mean squared error regression and binary cross-entropy classification are used to create the final loss function for training.

D. Bounding Box Formation Technique

In the subsections that came before, the structure of the recommended network and its training loss function was discussed. This subsection described how to create a bounding box. Class activation maps (CAMs) and regression activation maps are used during the inference phase (RAMs). As previously mentioned, two more 1X1 N convolutional layers are added in order to generate the CAMs and RAMs for the test picture, where 1X1 signifies the kernel size and N is the number of classes.

1) The predictions from the classification output are thresholded by a value to check if a class instance is present. The best value of this threshold for testing empirically is 0.5 since multi-label categorization is frequently binary.
2) If a class has a value greater than the threshold value, the regression predictions are used to calculate the number of class instances for an item. At this point, further thresholding is carried out in order to determine the precise (integer) number of instances. If the regression forecast for the class's instance count is larger than 0.5, it is considered that this class has one instance (a threshold chosen based on the standard mathematical rounding concept in which values less than 0.5 are mapped to zero and values greater than 0.5 are mapped to one).
3) Once the number of occurrences for each class in the picture has been established, the regression activation map (a 28-28 grid) is normalized by the number of objects in order to eliminate the noise blobs. The largest of the remaining blobs is then chosen as the proper single-cell center. Because RAMs are learned by global average pooling, the prior RAM filtering method works well because this produces high activations for the region corresponding to the visual characteristic of the class with the highest level of activation; this one-cell activation is taken as the center of the object.
4) To do non-maximum suppression, the next step is to create the bounding box using the threshold CAMs (we used 1.0, 0.9999, 0.999, 0.99, and 0.9 for each set of classes). The extensions of the objects are

found in CAMs, and they can be scattered throughout the globe as multiple little portions or they can overlap. There are four types of class activations that can affect how the bounding box develops based on the amount of instances of each existing class:

- If a class has exactly one instance, there is just one instance of each item in the related CAM (a 28 28 grid). All of the active cells are counted. The locations of the top and leftmost activated cells on the vertical and horizontal axes, respectively, of the grid, are taken as the top left corner of the bounding box, and the locations of the bottom and rightmost are taken as the bottom right corner, regardless of the shape of the region in the CAM (i.e., whether it is unitary or fractionated in the CAM's grid)
- If there are several instances of a class and the number of instances equals the number of unique regions in the CAMs, then each centre has a corresponding instance represented by a linked area in the CAM. Each region's bounds are determined by the grid position's lowest and maximum indices.

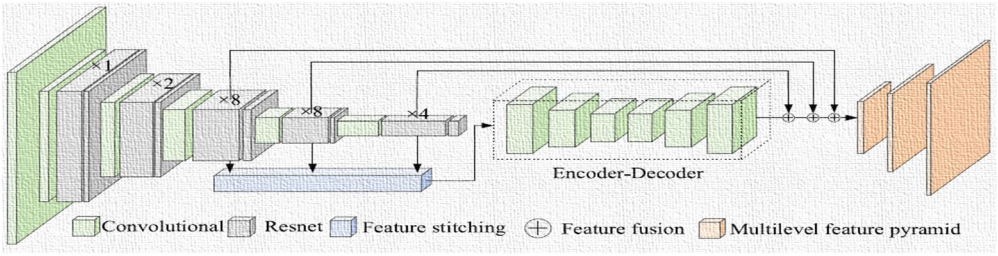


Fig.1. Vehicle Detection Model Network Structure.[3]

III. METHODOLOGY

A. Yolov3

A real-time object detection system called YOLOv3 (You Only Look Once, Version 3) recognizes particular things in films, live feeds, or still photos. To find objects, YOLO employs features that a deep convolutional neural network has learned.

Darknet-53: Darknet-19 is the name of the network architecture that YOLOv2 uses. It has 24 layers in total, including 19 convolutional layers (thus the name darknet-19) and 5 maximum clustering layers. Due to the loss of several fine-grained information during input down sampling, YOLOv2 is not particularly good at recognizing small targets. In order to get low-level features, YOLOv2 uses identity mapping to connect feature maps from the preceding layer.

Under Three Scales Detect: When the input image size is decreased to 32, 16 or 8, respectively, YOLOv3 generates predictions at each of three scales that are precisely stated. The 82nd layer is in charge of making the initial prediction. The network lowers the visual resolution for the first 81 layers until the 81st layer's pitch is 32. The size of the resultant feature map, if we start with a 416 416 image, is 13 13. Here, we employ a 1 1 detection kernel to produce a detection feature map with dimensions of 13 13 255. The layer 79 feature map is then sampled twice to a dimension of 26x26 after passing through numerous convolutional layers. The feature map of layer 61 and this feature map are then thoroughly concatenated.

Anchor Boxes: 9 anchor boxes altogether are used by YOLOv3. In each ratio, three. If you train YOLO on your own dataset, you must create 9 anchor points using K-Means clustering.

Additional Bounding Boxes: More bounding boxes are predicted by YOLOv3 than YOLOv2 for input photos of the same size. When YOLOv2's original resolution is 416 × 416, for instance, it is assumed that 13 x 13 x 5 = 845 boxes. Five boxes are found in each grid cell using five anchor points. The prediction are given below[5]: -

$$b_x = \sigma(t_x) + C_x \qquad (i)$$
$$b_y = \sigma(t_y) + C_y \qquad (ii)$$
$$b_w = pw^{e^{tw}} \qquad (iii)$$
$$b_h = ph^{e^{th}} \qquad (iv)$$

Softmax Abandoned : YOLOv3 classifies items found in photos using several labels. Previously in YOLO, the author was accustomed to using softmax level scores and regarded the class of objects encompassed in the bounding box as having the greatest score. This was altered in YOLOv3.

Loss Function

$$Loss = \lambda_{coord} \sum_{i=0}^{S^2} \sum_{j=0}^{B} 1_{ij}^{obj} \left[(x_i - \hat{x_i})^2 + (y_i - \hat{y_i})^2\right]$$

$$+\lambda_{coord}\sum_{i=0}^{S^2}\sum_{j=0}^{B}1_{ij}^{obj}\,(2-w_i\times h_i)[(w_i-\widehat{w_\iota})^2$$

$$+(h_i-\widehat{h_\iota})^2]-\sum_{i=0}^{S^2}\sum_{j=0}^{B}1_{ij}^{obj}[\widehat{C_\iota}\log(C_i)$$

$$+\left(1-\widehat{C_\iota}\right)\log(1-C_i)]$$

$$-\lambda_{noobj}\sum_{i=0}^{S^2}\sum_{j=0}^{B}1_{ij}^{obj}[\widehat{C_\iota}\log(C_i)$$

$$+\left(1-\widehat{C_\iota}\right)\log(1-C_i)]-\sum_{i=0}^{S^2}1_i^{obj}\sum_{c\epsilon classes}^{S^2}\times[\widehat{p_\iota}(c)\log(p_i(c))+(1-\widehat{p_\iota}(c))\log(1-p_i(c))] \quad \text{(v)}$$

B. Fast R-CNN

By categorizing object suggestions with a deep ConvNet, the Region-based Convolutional Network approach (RCNN) provides excellent object detection accuracy. On the other hand, R-CNN has substantial disadvantages: Training involves several stages. A ConvNet on object suggestions is initially tuned using R-CNN using log loss. Then SVMs and ConvNet features are matched. These SVMs take the place of the SoftMax classifier that was trained through fine-tuning as object detectors. It is at the third training step when bounding-box regressors are learnt.

C. Faster R-CNN

A step up from Fast R-CNN is Faster R-CNN. Due to the region proposal network, Quicker R-CNN is faster than Fast R-CNN, as the name suggests (RPN). A fully convolutional network that produces suggestions with various sizes and aspect ratios is the region proposal network (RPN). The RPN uses the language of neural networks with an emphasis on object detection (Fast R-CNN) what to watch out for.

Anchor boxes were proposed in this study as an alternative to pyramids of photos (several instances of the same image at various scales) or pyramids of filters (i.e. multiple filters with varied sizes). An anchor box is a reference box with a specific scale and aspect ratio. The same region can have multiple sizes and aspect ratios if there are many reference anchor boxes. This may be compared to a pyramid constructed of reference anchor boxes. The detection of objects with different scales and aspect ratios is made possible by the process of mapping each region to a distinct reference anchor box.

D. Mask R-CNN

Mask R-CNN, sometimes known as Mask RCNN, is the most advanced Convolutional Neural Network (CNN) for instance and picture segmentation. Faster R-CNN, a region-based convolutional neural network, served as the foundation for Mask R-CNN. Knowing the idea of image segmentation is a prerequisite for understanding how Mask R-CNN operates. The task of computer visions, the technique of dividing a digital image into several parts is called image segmentation (sets of pixels, also known as image objects). In this segmentation, borders and objects are located (lines, curves, etc.).

E. Pyramid Network

With the accuracy and speed of a pyramid notion in mind, a feature extractor known as the Feature Pyramid Network (FPN) was developed. It replaces detectors like Faster R-feature CNN's extractor for object identification and creates several feature map layers (multi-scale feature maps) with greater quality information than the traditional feature pyramid.

Dataset: The dataset used was Kitti Dataset and COCO Dataset and our own collected data from live city traffic. The Kitti Dataset consists of snippets got from a 360-degree camera, recorded in highway and roads of rural areas in Karlsruhe. The dataset almost depicts as if the video is a surveillance video.

Analysis: The picture that shows beneath the traffic film is quite similar to a real image that was really taken on the road by a car's camera and is included in the KITTI data collection. According to three distinct KITTI data set requirements, as shown in Table 2, the AP of the algorithm utilized in this work is 95.04%, 92.39%, and 87.51%, respectively. These outcomes outperform YOLOv3, whose results were, respectively, 2.49%, 3.68%, and 9.73%. Because the convolutional feature MAP at the bottom is only up sampled once by the YOLOv3 detection model's top stage before performing feature stitching.

TABLE I. AVERAGE PRECISION IN THREE DIFFERENT DIFFICULTY LEVELS UNDER THE KITTI DATASET

Algorithm Name	Average Precision %			Time
	Easy	Moderate	Hard	
R-CNN	32.23	26.04	20.93	-
Faster-R-CNN	87.90	79.11	70.19	142
YOLOv3	95.04	92.39	87.51	34

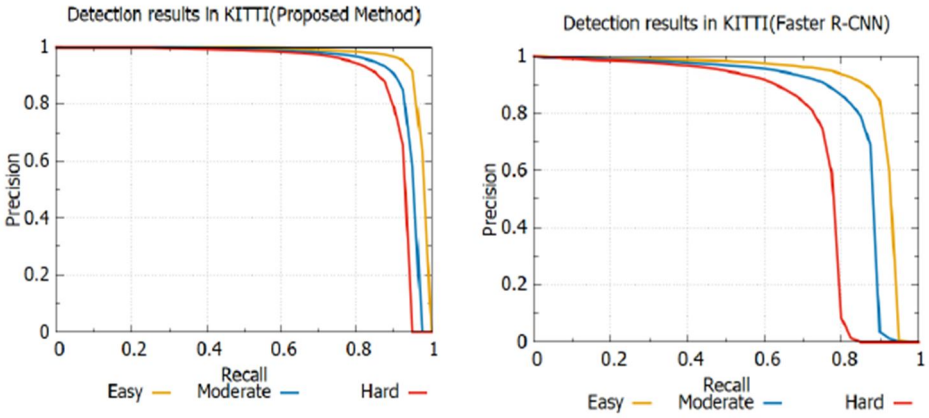


Fig 3.a & 3.b. P-R Diagram in YOLOv3 and Faster R-CNN in three difficulties

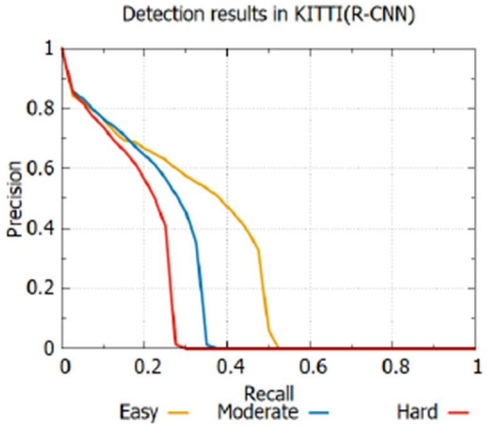
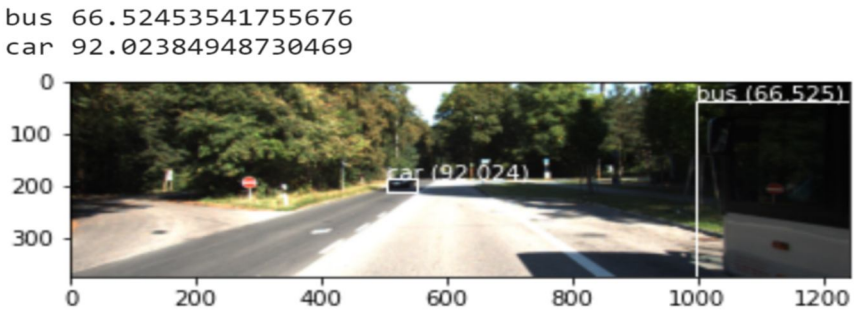
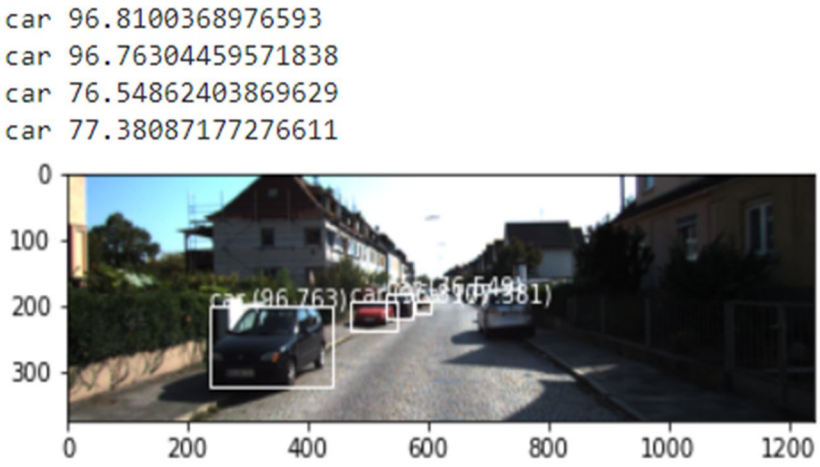


Fig.4. P-R Diagram in R-CNN in three different Difficulties



(a)

The above Graphs show that the model worked with high accuracy and with high speed in YOLOv3 model rather than Faster R-CNN or R-CNN. The self-collected images also showed the same results in YOLOv3 model. The difficulties that were set for the images were of three levels easy moderate and hard. as showed in the fig 3a and 3b. The difficulties differed with according to several properties like rainy, sunny, cloudy, dark, bright, distorted images etc. In all the difficulties YOLOv3 gave very high accuracy.



(b)

Fig 5a & 5b. Vehicle Detection YOLOv3

IV. Conclusion

In this study, the YOLOv3 network model is applied to the problem of vehicle recognition in videos of traffic surveillance. It was shown that during the actual detection phase, small scale autos frequently go undetected. To efficiently and effectively construct multi-scale features that can adapt to the identification of multi-scale target vehicles, we present a unique feature pyramid module built on the basis of YOLOv3 and based on encoding and decoding. After being tested on the KITTI dataset, the impact has been increased. Good detection results have been reached for vehicle targets of various sizes, especially for the identification of microscopic targets. The accuracy is significantly better than the YOLOv3 algorithm and can better meet the requirements of practical applications.

References

[1] BINGXIN HOU et al, A Fast Lightweight 3D Separable Convolutional Neural Network with Multi-Input Multi-Output for Moving Object Detection, 2021

[2] Yiping Gong, et al, Context-Aware Convolutional Neural Network for Object Detection in VHR Remote Sensing Imagery, 2020

[3] FENG HONG, et al, A Traffic Surveillance Multi-Scale Vehicle Detection Object Method Base on Encoder-Decoder, 2020

[4] Hong-Mei Sun, Rui Sheng Jia, Finding every car: a traffic surveillance multi-scale vehicle object detection method, [Springer Science + Business Media, LLC, part of Springer Nature 2020]

[5] Shi, L,;Zhang, F,;Xia,J,;Xie,J, Z,;Liu,R. Identifying Damaged Building in Aerial Images Using the Object Detection Method, Remote Sens. 2021,13,4231

[6] Xiu-Zhi Chen, Chieh-Min Chang, Chao-Wei Yu and Yen-Lin Chen A Real-Time Vehicle detection system under various Bad Weather Conditions Based on a Deep Learning Model without, Published: 9 October 2020

[7] Dinesh Rajan,Brett Story, Xinxiang Znang, Night Time Vehicle Detection and Tracking by Fusing Sensor Cues from Autonomous Vehicle, May-2020

[8] Kun Wang Maozhen Liu1, YOLOv3-MT:A YOLOv3 using multi-tracking for vehicle visual detection, 30 April 2021/ Published online: 4 june 2021

[9] XIAOTAO SHAO, CAIKE WEI, YAN SHEN, Feature Enhancement on CycleGan for Night time Vehicle Detection, November 27, 2020 acceted December 15,2020

[10]Mohammed Rabah, Ali Rohan, Heterogeneous Parallelization for Object Detection and Tracking in UAVs, February 7, 2020, accepted February 19, 2020 date of current version March 11,2020

[11]GIHA YOON,GEN-YONG KIM, HARK YOO, Implementing Practical DNN-Based Object Detection Offloading Decision for Maximizing Detection, Performance of Mobile Edge Devicedate of publication October 8,2021

[12]Ye Tao, Zho Zongyang, Chai Xinghua, Low-altitude small-sized object detection using lightweight feature-enhanced convolutional neural network, journal of System Engineering and Electronics, Vol. 32, 4 August 2021, pp.841-853

[13]P. Tu, T. Sebastian, G. Doretto, N. Krahnstoever, J. Rittscher and T. Yu, "Unified crowd segmentation", Computer Vision.