Australian Journal of Science and Technology

ISSN Number (2208-6404)
Volume 6; Issue 4; December 2022

Original Article

Vehicle detection and classification technique using convolutional neural network

Nguyen Ve Binh, Thai Vu Hien*

Faculty of Advanced Science and Technology, University of Danang, University of Science and Technology, Vietnam

ABSTRACT

Vehicle object detection and recognition are one of the important research directions in the field of computer vision. Solving this problem can help to improve traffic management, which is particularly helpful in busy Asian cities. Automatic vehicle recognition also leads to building successful autonomous cars. While classic machine learning methods are able to detect and recognize vehicle objects, it is convolutional neural networks (CNNs) that have really shown the capability of machine learning to effectively address the vehicle detection and recognition. In this paper, we use the YOLOv5 algorithm that is refined on a custom dataset to detect and classify five vehicle classes on the road: bicycles, motorbikes, buses, cars, and trucks. Although there is still room for further development, the experiment of this paper shows promising results. In particular, the model is able to detect and classify the vehicles effectively and provides a value of 0.75 for mAP50, and 0.548 for mAP50-95. Besides, a simple application is also proposed for real-time vehicle detection and classification.

Keywords: Classification, deep learning, vehicle detection, YOLOv5

Submitted: 05-12-2022, **Accepted:** 14-12-2022, **Published:** 25-12-2022

INTRODUCTION

Object detection is a well-known technique for locating items in an image or video. Object detection is mostly accomplished through the use of machine learning and deep learning approaches. Vehicle object detection is an important branch of object detection, which is also the foundation of driverless, intelligent transportation, vehicle tracking, and other fields. We have a lot of models such as R-convolutional neural networks (CNNs)[1] and F-CNN[2] to detect and recognize vehicles. The algorithm used in this research is YOLO, as illustrated in Figure 1. It uses a single neural network to process an entire image. The image is divided into regions and the algorithm predicts probabilities and bounding boxes for each region. YOLO is well known for its speed and accuracy and it has been used in many applications such as healthcare, security surveillance, and self-driving cars. In this paper, based on the YOLO architecture, we propose a model to detect and recognize or classify vehicles: buses, cars, motorbikes, bicycles, and trucks. The experiment provides satisfactory results, although there is still room for improvement. Furthermore, we also create a vehicle detection and recognition application based on the proposed deep learning model.

METHODS

YOLOv5 Architecture

YOLOv5 was designed by Glenn Jocher and released in 2020 (shortly after the release of YOLOv4[3]). The open source is available on github. YOLOv5 is a state-of-the-art, real-time object detector, and YOLOv5, which is based on YOLOv1-YOLOv4. Continuous improvements have made it achieve top performances on two official object detection datasets: Pascal visual object classes[4] and Microsoft COCO.[5] In the YOLOv5 architecture, it has three parts that are backbone, neck, and head. Now we look at the details of each part of YOLOv5. Up to the day of writing this paper, there is no research paper that was published for YOLOv5 as mentioned here, hence, the illustrations used below are unofficial and serve only for explanation purposes. YOLOv5 is a single-stage object detectors, whose architecture is composed of

Address for correspondence: Thai Vu Hien, University of Danang, University of Science and Technology, Vietnam.
E-mail: tvhien@dut.udn.vn

three components: Backbone, neck, and a head to make dense predictions, as shown in Figure 1.

Backbone

The backbone is a pre-trained network used to extract rich feature representation for images. This helps reduce the spatial resolution of the image and increases its feature (channel) resolution.

The architecture of a C3 module is given in Figure 2a. The C3 module uses the CSP Block.[6] CSP Net solves the problems of repeated gradient information in large-scale backbones and integrates the gradient changes into the feature map, thereby decreasing the parameters and floating-point operations per second of model, which not only ensures the inference speed and accuracy but also reduces the model size.

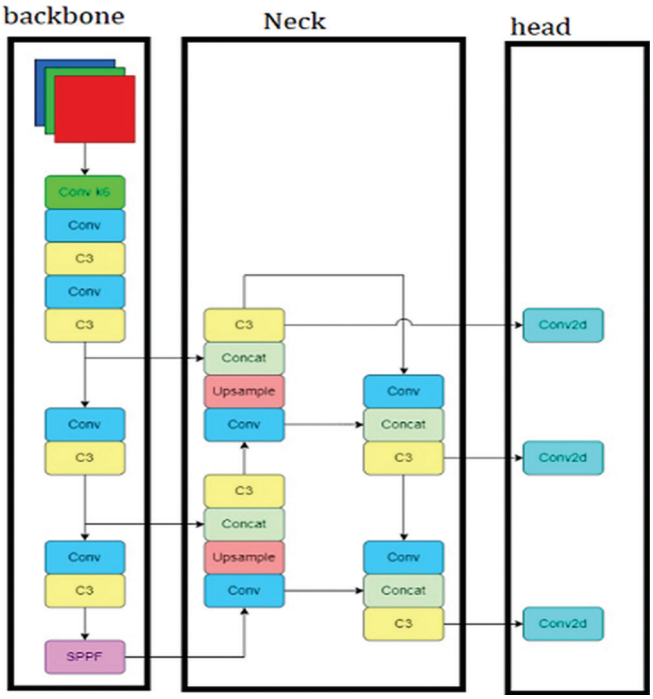

Figure 1: YOLOv5 architecture

The architecture BottleNeck1 is represented in Figure 2b. This bottleneck structure exploits the res-net architecture[7] that helps neural networks prevent degradation.

Spatial pyramid pooling fast (SPPF)

SPP[8] was released in 2014. At that time, object detection still used fully-connected layers for output. As usual, the input image has to be fixed-size to have good result detection but with the appearance of SPP, the input image can be of arbitrary size/scale. SPP will create a fixed-length vector as a bag of words. The neural network in Figure 1 uses SPPF that we illustrate in Figure 3. The difference between SPP and SPPF is that SPPF uses max pooling layers series, instead of max pooling series as SPP. SPPF is 2 times faster than SPP.

PANet

The YOLOv5 uses a path aggregation network (PANet)[9] as its neck to boost information flow. PANet adopts a new feature pyramid network structure with enhanced bottom-up path, which improves the propagation of low level features. At the same time, adaptive feature pooling, which links feature grids and all feature levels, is used to make useful information in each feature level propagate directly to the following subnetwork. PANet improves the utilization of accurate localization signals in lower layers, which can obviously enhance the location accuracy of the object.

Head

The head of YOLOv5, namely, the YOLO layer, generates three different sizes (18 × 18, 36 × 36, 72 × 72) of feature maps to achieve multi-scale prediction, enabling the model to handle small, medium, and big objects.

The equations to compute the target coordinates for the bounding boxes have changed from previous versions. Readers can find more details in the corresponding papers.

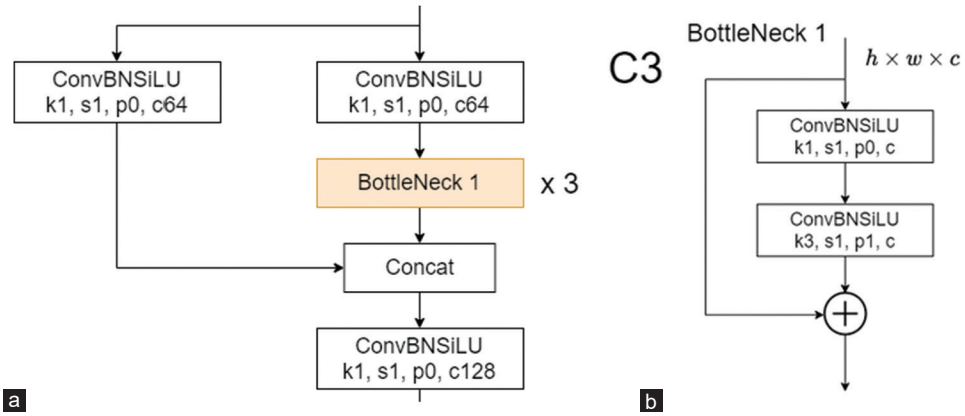

Figure 2: C3 module (a) and bottle neck architecture (b)

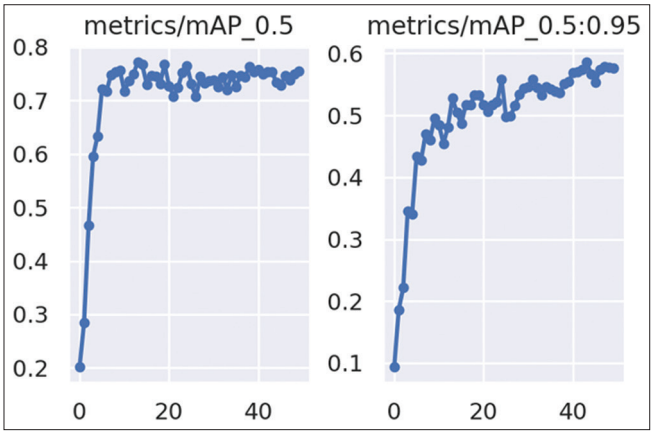

Figure 3: Mean average precision

Loss function

Yolov5 returns three outputs: the class of objects, their bounding boxes, and the objectness scores Thus, it uses binary cross entropy to compute the classes loss and the objectness loss. CIOU (Complete Intersection over Union)computes the location loss. The formula for the final loss is given by the following equation:

$$L_{obj} = 4.0 \cdot L_{obj}^{small} + 1.0 \cdot L_{obj}^{medium} + 0.4 \cdot L_{obj}^{large} \quad (1)$$

$$L_{ciou} = \sum_{i=0}^{S^2} \sum_{j=0}^{B} I_{i,j}^{obj} \left[1 * IoU + \frac{\rho^2\left(b, b^{gt}\right)}{c^2} + av \right] \quad (2)$$

Where

$$a = \frac{4}{\pi^2}\left(arctan\left(\frac{w^{gt}}{h^{gt}}\right) - arctan\left(\frac{w}{h}\right)\right)^2$$

$$v = \frac{v}{\left(1 - IoU\right) + v}$$

S^2, B: Feature map scale and prior frame.
ρ: Euclidean distance.
$I_{i,j}^{obj}$: If there is a target at the first box of the grid, take 1 and 0, respectively.
c: The diagonal distance between the predicted box and the actual box closure area.
b, w, h: Central coordinates and width of the prediction box.
b^{gt}, w^{gt}, h^{gt}: Center coordinates and width height of the actual frame.

$$Loss = \lambda_1 * L_{cls} + \lambda_2 * L_{obj} + \lambda * L_{bo} \quad (3)$$

Where
L_{cls}: the category loss.
L_{obj}: the confidence loss.
L_{box}: the bounding box loss.

Activation function

Choosing an activation function is crucial for any deep learning model, for YOLOv5, the authors went with SiLU and Sigmoid activation function. SiLU[10] stands for Sigmoid Linear Unit and it is also called the swish activation function. It has been used with the convolution operations in the hidden layers, while the Sigmoid activation function[11] has been used with the convolution operations in the output layer.

EXPERIMENTS AND RESULTS

Data Set

For vehicle object detection, we used the YOLOv5-m network, and for network training, we used our dataset. There is no optimal result for dataset partitioning in network training. Our dataset partitioning approach is conventional. We divided the dataset into a 60% training set and a 20% validation set and 20% test set. Our dataset contains vehicles, and the images for the training, validation, and test sets were chosen arbitrariness from the dataset.

Training

The model was trained on 1030 samples with five classes that are bicycles, motorbikes, buses, cars, and trucks. The training set was divided into 16 batches and the number of epochs was 49.

The loss function is used to determine the training state of the model in the current iteration and to calculate the difference between the predicted and true values during the iteration. The YOLOv5 loss function is calculated as equation (3).

Looking at the loss curve shows, we see that at 49 epochs, the curve essentially stops decreasing, and the network training is essentially complete. The value of the box loss function of the training set decreased from an initial 0.08–<0.1, object loss decreased from 0.04 to 0.015, and class loss decreased from 0.0.35 to 0.01.

Validation

There are 106 samples for validation. The validation loss is the same as the training loss. From the validation results, we see that the validation loss is approximate to the training loss.

Precision and recall

The precision measures how accurate a model is at recognizing an object. The recall rate is how much a model searches for the entire object when recognizing the object. Figure 4 shows the variation of the precision and recall during the training of the model according to the number of epochs. The last epoch

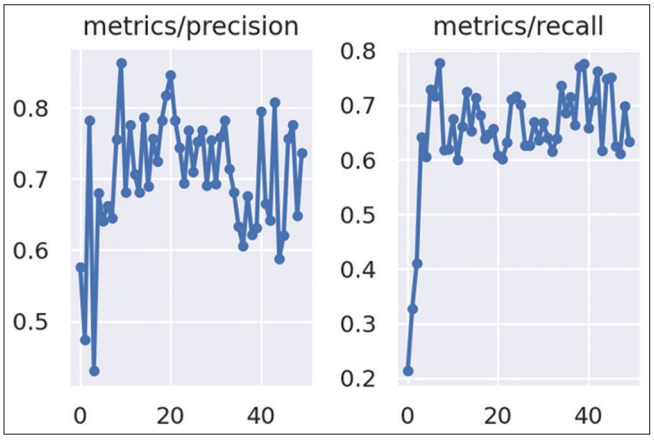

Figure 4: Precision and recall

Figure 5: Application of vehicle detection and classification.

achieved by the model during training is 0.75, and the highest recall is 0.65.

Mean average precision (mAP)

The mAP is an evaluation metric that assesses network performance in the object detection field. mAP@0.5 is the area under the P-R curve of the network when setting the detection IOU ratio threshold to 0.5. mAP@0.5:0.95 is the average value of the area under the P-R curve of the network when setting the detection positive case intersection and ratio threshold from 0.5 to 0.95, calculated individually at a step size of 0.05. Thus, mAP@0.5:0.95 is harder to achieve. Figure 3 shows the mAP curve during training. The final mAP@0.5 achieved by the algorithm is around 0.75, and the mAP@0.5–0.95 is 0.56.

Confusion matrix

Confusion matrix is a very popular measure used while solving classification problems. It can be applied to binary classification as well as for multiclass classification problems.

The experimental results show good predictive performance on car, motorbike, bus, and bicycle with accuracy of 0.83, 0.79, 0.89, and 0.75, respectively. For trucks, our model does not detect very well with an accuracy of only 0.25. This limited result may be due to the lack of training data for the class "truck."

Application

The model was used to build an application [Figure 5] which can access cameras through IP addresses.

This demo was made on Vo Nguyen Giap street, Da Nang City, Viet Nam. In Figure 5, the model can detect vehicles on the road with high confidences, correct labels, and correct bounding boxes that are assigned to each detected object.

CONCLUSION

In this paper, we study the vehicle detection model for five classes. The model does not detect the truck well. We can improve by adding more data about the truck so that the model can learn better. We can rely on this model to be able to build some applications such as driverless, intelligent transportation, vehicle tracking, and recording its speed.

REFERENCES

1. Girshick R, Donahue J, Darrell T, Malik J. Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation. In: 2014 IEEE Conference on Computer Vision and Pattern Recognition. New Jersey: IEEE; 2014. p. 580-7.
2. Zhao W, Fu H, Luk W, Yu T, Wang S, Feng B, *et al*. F-CNN: An FPGA-Based Framework for Training Convolutional Neural Networks. In: 2016 IEEE 27th International Conference on Application-specific Systems, Architectures and Processors (ASAP). London, 2016. p. 107-14.
3. Bochkovskiy A, Wang CY, Liao HY. YOLOv4: Optimal Speed and Accuracy of Object Detection. arXiv: Computer Vision and Pattern Recognition; 2020.
4. Available from: http://www.host.robots.ox.ac.uk/pascal/VOC/voc2012 [Last accessed on 2022 Dec 01].
5. Available from: https://cocodataset.org [Last accessed on 2022 Dec 01].
6. Wang CY, Liao HY, Wu YH, Chen PY, Hsieh JW, Yeh IH. CSPNet: A New Backbone that can Enhance Learning Capability of CNN. In: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW). Seattle, WA, 2020. p. 1571-80.
7. He K, Zhang X, Ren S, Sun J. Deep Residual Learning for Image Recognition. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). Las Vegas, NV: IEEE; 2016.
8. He K, Zhang X, Ren S, Sun J. Deep Residual Learning for Image Recognition. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR); 2016. p. 770-8.
9. Liu S, Qi L, Qin H, Shi J, Jia J. Path Aggregation Network for Instance Segmentation. In: 2018 IEEE/CVF Conference on

Computer Vision and Pattern Recognition. Salt Lake City, UT, 2018. p. 8759-68.
10. Elfwing S, Uchibe E, Doya K. Sigmoid-weighted linear units for neural network function approximation in reinforcement learning. Neural Netw 2018;107:3-11.
11. Dubey SR, Singh SK, Chaudhuri BB. Activation functions in deep learning: A comprehensive survey and benchmark. Neurocomputing 2022;503:92-108.