Vehicle Detection Using YOLOv4-tiny Model

Abir-Al-Arafat (id: 18-36138-1)[a], Snigdho Dip Howlader (id: 18-36177-1)[a], Suparan Sharma (id: 18-37275-1)[a], Nihal Abedin Annon (id: 18-37301-1)[a]

[a]*Department of Computer Sciences, American International University-Bangladesh*

Abstract

YOLOv4-tiny is proposed based on YOLOv4 to simplify the network structures and reduce parameters, which makes it suitable for developing on the mobile and embedded devices. It improves the real-time of object detection and it is a fast object detection method. The YOLOv4-tiny network was tailored for real-time implementation by significantly reducing the model size, detection time, number of computational parameters, memory consumption and refining the network for three-channel images to compensate for the loss of accuracy due to light-weighting. The aim of this paper is to train a vehicle dataset using YOLOv4-tiny model so that the vehicles can be detected by the machine using bounding boxes. The test experiments were completed entirely on a computer using Google Colab and achieved a Mean Average Precision of 51.60% on the Vehicle Dataset. The dataset consists of 1505 images of vehicles divided into 5 classes like ambulance, bus, car, motorcycle and truck. In the YOLOv4-tiny model experiments hyper parameter were also carried out on flipped and cropped values and brightness data augmentation. In terms of computational speed the YOLOv4-tiny's scores are better. After running the YOLOv4-tiny model on the dataset, following were the results mAP 51.60%, precision 0.56, recall 0.50, F1-score 0.53, and Average IoU 44.11%. The experimental results show that the state of the art model which already has real-time object detection capabilities can be further improved for highly targeted use cases.

Keywords: Object detection, YOLOv4-tiny, Vehicle Detection, Convolutional Neural Networks (CNN)

1. Introduction

Object Detection is a sub-field of computer vision that is revolutionizing the way of identifying things. Object detection and tracking are important and challenging tasks

in many computer vision applications such as surveillance, vehicle detection and autonomous robot navigation. Object detection differs from object classification in terms of bounding box formation. We will try to draw a bounding box around the specific object in detection algorithms. There can be multiple bounding boxes in an image, detecting different objects in the image. A long-standing target in the field of vehicle detection is to develop systems that can perceive and understand a rich and huge variety of vehicles. In this project YOLOv4-tiny object detection model was trained on an open image vehicle dataset using the darknet framework. So the YOLOv4-tiny object detection model is going to be used to teach it to predict various vehicles. YOLOv4-tiny builds on the progress of YOLOv4 but emphasizes model speed and a smaller model size for inference even on smaller hardware. YOLOv4-tiny was released on June 24, 2020 by Alexey Bochkovskiy. It achieved 40.2% mAP on MS COCO which is less accurate than YOLOv4(64.9%) but it achieved 371 fps using a GTX 1080 Ti which is much faster than a full YOLOv4 model. So anyone aiming to create a fast model that might need to work in a constrained computational environment but be fast should consider YOLOv4-tiny. It can be mobile application, embedded device inference and constrained training resources. A typical object detector architecture consists of four components- the input, the backbone, the head, and the neck. The backbone is the pre-trained network taking the input image and does feature extraction. The head predicts the classes and bounding boxes of objects. The neck serves the purpose of increasing robustness by collecting feature maps from intermediate stages. These algorithms can be trained and optimized for the problem of vehicle detection. In this paper, we have used the object detection algorithm YOLOv4-tiny and trained a vehicle dataset on it.

2. YOLOv4-tiny network

The YOLOv4-tiny model is a lightweight version of the YOLOv4 model. It is a lightweight real-time detection algorithm suitable for embedded platforms. Compared with YOLOv4, the detection accuracy is reduced, but it achieves model compression while greatly improving Detection speed.

3. Literature Review

YOLO (You Only Look Once) is an algorithm developed by Joseph Redmon in 2016 to detect objects in real-time based on Convolutional Neural Network. This algorithm can detect through image input, video input, and real-time using a webcam. YOLO uses an Artificial Neural Network approach to detect objects in an image. This network divides the image into several regions and predicts each bounding box and probability for each region. Then the bounding box is compared with each predicted probability. In general, the YOLO algorithm has simple steps when compared to SSDs and the faster R-CNN. Most of the steps in this algorithm same as image classification using the CNN algorithm. YOLO performs bounding box calculations with one map feature scale. The stages of the YOLO algorithm are input images marked into several S x S grid boxes where each grid box create a few bounding

boxes of various sizes. The bounding box size is bigger than the grid box size. If the bounding box contains objects, the grid box that create the bounding box is responsible for detecting that object. One grid box can only expose one object. If more than one bounding box for a certain grid box detects an object, it is only allowed to select one object with one bounding box that has the highest level of accuracy. No Max Suppression is used to determine when many bounding boxes are detected for the same object. Yolov4-tiny is a model developed from the YOLOv4 algorithm. This model has a simpler network structure and reduces the parameters that exist in YOLOv4. With a network structure like this, YOLOv4-tiny is suitable for development on mobile devices and embedded devices.

4. Methodology

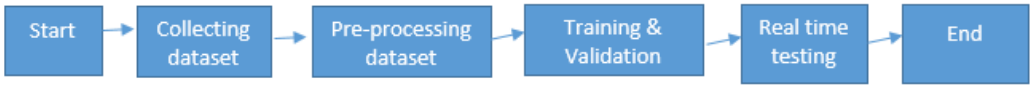

Figure: workflow

The figure above shows the steps that were followed in this project. The first step is collecting dataset for the project. The dataset is divided into two parts, the dataset for training and the other one is for validation. The data used is a dataset derived from an open image vehicle dataset. The dataset is an image with .jpg format which has a size of 416 x 416 pixels. The amount of data used in this project were 1505 with the distribution of training data as 1300 images, validation data as 125 images and testing data as 63 images. This dataset consists of 5 classes of vehicles types such as ambulance, bus, car, motorcycle and truck.

The second step is the pre-processing dataset that is labeling all images with annotation tools which produces a .txt file that contain image information. In addition to labeling, there is a data augmentation process in the pre-processing dataset. Data Augmentation is a technique of manipulation/modification of data without losing the essence of the data. In this research, data augmentation was carried out to reproduce the dataset that will be used. The augmentation used in this project was flip, crop and changing brightness of data.

The next step is the training and validating the data by using Google Colaboratory. The input of this process is the training dataset and validation dataset that has been prepared in advance. While the output of this process is a weights file which will later be used during real-time testing. After knowing the best weights model generated from the training and validation process, the next stage is real-time testing. This process involves input images in the yolov4-tiny models.

5. Result and discussions

This training and validation process was carried out through Google Colab. The training and validation process was done using the YOLOv4-tiny configuration. The configuration

parameters used are Batch 64, Subdivision 24, Network Size 416x416, Max_batches 10000, Steps 8000 and 9000 and Classes 5. The mean value of average precision, FPS (Frames per second) and GPU utilization are used to quantitatively evaluate the performance of different methods. The mAP is the mean value of average precision for the detection of all classes. FPS denotes the number of images that can be detected successfully in one second. GPU utilization denotes used GPU memory in testing the different detection methods.
YOLOv4-tiny method belongs to lightweight deep learning method. They have relatively simple network structure and few parameters. Therefore, they have better performance in FPS and not so good performance in mAP, and more suitable for deploying on the mobile and embedded devices.
The process successfully performed 9066 iterations and the total time needed to complete the training and validation process using YOLOv4-tiny is 3 hours. The YOLOv4-tiny model produces the best mAP of 53.69% and an average loss of 0.149041. Based on the results of the training and validation that has been carried out, the best results from the YOLOv4-tiny model are yolov4-obj_best.weights with AP of Ambulance 75.54%, Bus 61.67%, Car 37.94%, Motorcycle 53.32% and Truck 29.54%.

6. Conclusion

In this project, we tried to produce real-time image detection of vehicles that can show the bounding box and prediction probability of objects in each image with some pre-existing tweaks and techniques. We tried to experimentally verify the results and present the same in the paper. The model can be run using Google Colab, local PC / laptop, or with a website built using the Darknet framework. The model can be run using images as inputs. The research results show that the YOLOv4-tiny architecture has a fast computational speed and good computational performance. The YOLOv4-tiny model has a mean average precision value of 51.60%, precision 0.56, recall of 0.50, F1-score 0.53 and Average IoU 44.11%. The YOLOv4-tiny model shows that the faster a model detects, it will result in a reduction in the accuracy value and vice versa. Then, several factors that affect the performance of the YOLO algorithm architecture, namely the smaller the subdivision value used, the mAP value will increase during the training and validation process. It is suggested to use increased amount of data. Because the model's performance increases depending on the size of the dataset. If the size of the dataset gets increased, the performance of the model will also increase. Then it can increase the number of classes/types of vehicles that will be detected in real-time.

References

[1] Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You only look once: Unified, real-time object detection. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 779-788).

[2] Mahto, P., Garg, P., Seth, P. and Panda, J., 2021. Refining Yolov4 for Vehicle

Detection.

[3] Hui, T., Xu, Y. and Jarhinbek, R., 2021. Detail texture detection based on Yolov4-tiny combined with attention mechanism and bicubic interpolation.

[4] Zhao, Zhong-Qiu & Zheng, Peng & Xu, Shou-Tao & Wu, Xindong. (2019). Object Detection with Deep Learning: A Review. IEEE Transactions on Neural Networks and Learning Systems. PP. 1-21. 10.1109/TNNLS.2018.2876865.

[5]. Dai, J., Li, Y., He, K., & Sun, J. (2016). R-FCN: Object detection via region-based fully convolutional networks. Advances in Neural Information Processing Systems, 379–387.