YOLO: Detection And Counting Vehicles

Thanh T.VO
Advanced Program in Computer Science
Faculty of Information Technology
University of Science, VNU-HCM
Email: vtthanh@apcs.vn

Tuan M.PHAM
Advanced Program in Computer Science
Faculty of Information Technology
University of Science, VNU-HCM
Email: pmtuan@apcs.vn

Thanh T.THAI
Advanced Program in Computer Science
Faculty of Information Technology
University of Science, VNU-HCM
Email: ttthanh2017@apcs.vn

***Abstract*—Traffic has increased enormously with economic development. Due to the increasing urban population and hence the number of cars, the need for controlling the traffic on the streets, highway, and roads is vital. Vehicle detection has been a part of the traffic surveillance system for many years. So Joseph Redmon et.al wants to find ways to help statistic and calculate the vehicles on the road. It helps distribute vehicles and solve traffic congestion. So that Joseph Redmon apply YOLO for this problem and we use YOLO in this paper but we more add counting vehicles for YOLO. The result of that can be used as a reference.**

***Index Terms*—vehicle detection, counting, traffic congestions**

I. Introduction

In recent years, vehicle traffic in the world in general and in VietNam in particular is increased and many problems have appeared [2]. For example, traffic jams, accidents, air pollution and so on. Traffic congestion has been a significant challenge problem, especial is detection and counting vehicles.

Our system uses YOLO to detect and counting vehicle on video. Our goal is counting the number of cars from a video or can count vehicles on the road with a camera in real- time. After it has been done, To increase the accuracy of counting cars, the team will only count in an area where the vehicle identification is the most obvious.

To solve this problem, M.Tursun and G. Amrullar proposed a method using an optimized virtual loop [15]. Another method was proposed by M. Lei, et. al. [9]. They used surveillance cameras and mounted at relatively high places to acquire the traffic video stream. Their methods are adaptive background estimation and Gaussian shadow elimination. And another method based on employing adaptive background subtraction and Kalman filtering for road/vehicle detection and tracking was proposed by Bas et. al [3].

We use YOLO [12] to detect car on the street because YOLO is simple and sees the entire image during training and test time so YOLO makes less than other systems like R-CNN [6], Fast R-CNN [14] to predict. But YOLO stills lags behind state of the art detection systems in accuracy. While it can quickly identify objects in images but it has not high accuracy like R-CNN.

Now we can use YOLO to detect vehicles and count vehicles in frames. To count cars as well as count other objects, there are currently many ways. It means that from a given frame there is a moving vehicle that subtracts from its background which we will get the vehicle's shape and then count. Another method uses the detection models that identify the vehicle on the road then count it. YOLO is a very fast detection model and it can detect object real-time so YOLO very suitable for counting vehicles. To identify an object in a photo, YOLO divides that image into the SxS grid, if the center of an object falls into a grid cell, that grid cell is responsible for detecting that object. Each grid cell predicts B bounding boxes and confidence scores for those boxes. Then classifies each grid, predicts bounding boxes, class probability. And giving the confidence of that prediction.

II. Related Work

Object detection is a core problem in computer vision. So that object detection has a lot of ways to detect such as Haar [13], HOG [4]... Moreover we compare YOLO with top detection frameworks.

Deformable parts models [5] (DPM) use a sliding window approach to object detection. DPM uses a disjoint pipeline to extract static features, classify regions, predict bounding boxes for high scoring regions. Yolo replaces all of these disparate parts with a single convolutional neural network. The network performs feature extraction, bounding box prediction, nonmaximal suppression, and contextual reasoning all concurrently. Instead of static features, the network trains the features in-line and optimizes them for the detection task. Yolo architecture leads to a faster, more accurate model than DPM.

R-CNN [10] use region proposals instead of sliding windows to find objects in images. Selective Search generates potential bounding boxes, a convolutional network extracts features, an SVM scores the boxes, a linear model adjusts the bounding boxes, and non-max suppression eliminates duplicate detections. Each stage of this complex pipeline must be precisely tuned independently and the resulting system is very slow. But YOLO shares some similarities with R-CNN. Each grid cell proposes potential bounding boxes and scores those boxes using convolutional features. However, Yolo puts spatial constraints on the grid cell proposals which helps mitigate multiple detections of the same object. Yolo also proposes far fewer bounding boxes, only 98 per image compared to about 2000 from Selective Search.

Fast and Faster R-CNN [7] focus on speeding up the R-CNN framework by sharing computation and using neural networks to propose regions instead of Selective Search.

III. Proposed Method

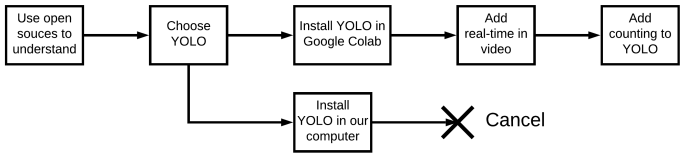

Fig. 1. To do list

Counting cars as well as counting other types of objects is an interesting topic. The authors divide it into two parts. The first part is detecting and the second part is counting. Initially, we detect an object on a photo with a few obvious objects, then on the video. After detecting completed, we count the number of vehicles and display results as well as storing the data for later using. For detecting vehicle, we decided to first try to make their own algorithms. Then if it is impossible or too difficult to solve, we will use some efficient algorithms available and try to understand it and make it better.

The authors decided to do manually to understand the datasets by testing with some of our simple comments about them. For detecting, we tried to analyze each image to find the identities of the vehicle. We try to draw observations as more as possible. E.g., the shape of the vehicle is slightly similar to a rectangle shape, wheels of a vehicle are four black circles, the license plate of the vehicle is white and has black characters in it, which is the basic identities seen from the image. Then we looked a little deeper into the image of the vehicle, which is the brightness of the image, the reflection of the car glass, the separation between the vehicle and the frame, etc. In each comment, the authors try to implement an algorithm and try to put in the experiment, but there are many errors and technical limitations. In conclusion, our manual methods could not solve the problem and get the expected results. So after that, our decision is using the available source code and tries to improve it.

First, the authors used the open sources on GitHub to understand image recognition. Those open sources are like [8], [11].. Then the authors try to use more networks like R-CNN, Faster-RCNN, YOLO. And the authors decided to use YOLO because time to image recognition is fast, stable and easy to install. But during the installation, the authors had the problem that YOLO used a lot of computer resources - the our computer only has Intel i5-7200U 2.5GHz chipset, RAM 8GB, NVIDIA GTX 940MX card with 2GB and SAMSUNG 128GB SSD hard drive. With our computer, the authors only use YOLO to detect cars in image but time to test is slow. When the authors try to test with videos, our computer very hot and usually crash when video has a lot of cars to detect. To solve this problem, the authors chose Google Colab because it has a Linux machine with a more powerful configuration to train and run YOLO. In addition, the authors also installed "PyTube" [1] to download videos from youtube to check YOLO can identify the car on each frame. And finally, we add the vehicle counting function on each frame to YOLO.

In YOLO, YOLO divides the input image and frame of video into an S S grid. If the center of an object falls into a grid cell, that grid cell is responsible for detecting that object. Each grid cell predicts B bounding boxes and confidence scores for those boxes. The confidence scores reflect how confident the model is contains an object and how accurate thinks the box is that it predicts. If no object exists in YOLO, the confidence score will be set zero

Each bounding box consists of 5 predictions: x, y, w, h and $confidence$. The (x, y) coordinates is the center of box and w, h is width and height of bounding box. Moreover, the $confidence$ prediction represents between the predicted box and any ground truth box.

Moreover, YOLO uses convolutional neural network and evalutes on PASCAL VOC detection dataset. When the model is included in YOLO, YOLO will separate the objects from the background and then classify the objects by PASCAL VOC dataset and ground truth. In VOC, it has 20 object classes with 5717 images train, 5823 images validation and 10540 images test. Moreover, it has folder Annotations with XML format and format has $x_{min}, y_{min}, x_{max}, y_{max}$ for bounding box

The authors count cars by counting vehicles on each frame thanks to the object segmentation of YOLO. When each frame appears, the authors will let YOLO analyze each frame and show how many cars in each frame. Moreover, the authors want to improve YOLO detect correctly because the authors know YOLO fast but not correctly like R-CNN or Fast-RCNN.

IV. Conclusion

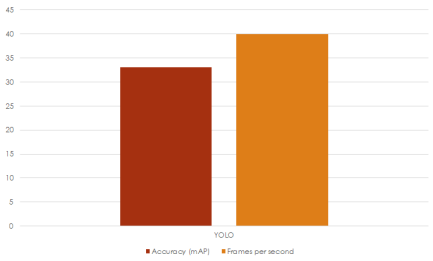

Fig. 2. Result on Pascal

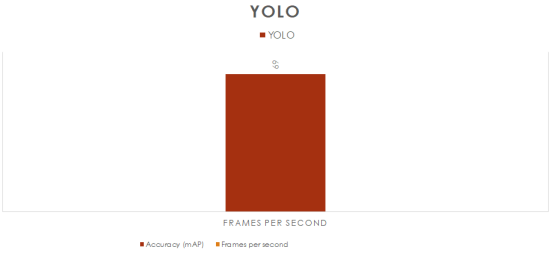

Fig. 3. Result on COCO

Look at two statistical charts above, we can see YOLO run so fast in all Pascal and COCO dataset but it does not have high accuracy. Moreover, we can draw some conclusions:

- The higher the accuracy, the slower the processing speed and vice versa.

- We need to choose a model that matches the problem we solve.

- High-speed model can be applied to problems that need real time speed.

- High-accuracy model can be applied to high precision problems. We can take it to compare with other models.

REFERENCES

[1] [Online]. Available: https://github.com/nficano/pytube

[2] "More vehicles cause traffic jam." [Online]. Available: https://www.khaleejtimes.com/nation/transport/more-vehicles-cause-traffic-jam

[3] E. Bas, A. Tekalp, and F. Salman, "Automatic vehicle counting from video for traffic flow analysis," July 2007, pp. 392 – 397.

[4] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) - Volume 1 - Volume 01*, ser. CVPR '05. Washington, DC, USA: IEEE Computer Society, 2005, pp. 886–893. [Online]. Available: http://dx.doi.org/10.1109/CVPR.2005.177

[5] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan, "Object detection with discriminatively trained part-based models," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 32, no. 9, pp. 1627–1645, Sep. 2010. [Online]. Available: http://dx.doi.org/10.1109/TPAMI.2009.167

[6] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Computer Vision and Pattern Recognition*, 2014.

[7] R. B. Girshick, "Fast R-CNN," *CoRR*, vol. abs/1504.08083, 2015. [Online]. Available: http://arxiv.org/abs/1504.08083

[8] J. Huang. Car detection and counting. [Online]. Available: https://github.com/hjptriplebee/car_detection_counting

[9] M. Lei, D. Lefloch, P. Gouton, and K. Madani, "A video-based real-time vehicle counting system using adaptive background method," *Signal-Image Technologies and Internet-Based System, International IEEE Conference on*, vol. 0, pp. 523–528, 11 2008.

[10] K. Lenc and A. Vedaldi, "R-CNN minus R," *CoRR*, vol. abs/1506.06981, 2015. [Online]. Available: http://arxiv.org/abs/1506.06981

[11] A. Muhamad. Vehicle detecion. [Online]. Available: https://github.com/AuzanMuh/vehicle_detection

[12] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun 2016. [Online]. Available: http://dx.doi.org/10.1109/CVPR.2016.91

[13] R. Reisenhofer, S. Bosse, G. Kutyniok, and T. Wiegand, "A haar wavelet-based perceptual similarity index for image quality assessment. signal processing: Image communication," vol. 61, 2018.

[14] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," in *Advances in Neural Information Processing Systems (NIPS)*, 2015.

[15] M. Tursun and G. Amrulla, "A video based real-time vehicle counting system using optimized virtual loop method," in *2013 8th International Workshop on Systems, Signal Processing and their Applications (WoSSPA)*, May 2013, pp. 75–78.