Real-Time Vehicle Detection from Captured Images

Soumen Santra
Department of Computer Application
Techno International Newtown
Kolkata, India
soumen70@gmail.com

Sanjit Roy
Department of Computer Application
Techno International Newtown
Kolkata, India
sanjitroy13@gmail.com

Prosenjit Sardar
Department of Computer Application
Techno International Newtown
Kolkata, India
prosenjitsardar19111995@gmail.com

Arpan Deyasi
Dept. of Electronics and Communication Engineering
RCC Institute of Information Technology
Kolkata, India
deyasi_arpan@yahoo.co.in

Abstract— **Objective of this paper is to primarily detect moving vehicles from real-time captured images so that we can later use them for a variety of purposes like detecting the number of vehicles that are present at a road at a certain point, understanding the motion of the vehicles, analyzing vehicle density at any traffic signal at any specified point of time; and based on our previous knowledge can make a good assumption where the vehicle will be heading towards. Signal-to-noise ratio, PSNER and mean-square-error are computed for each captured image using after detection. After establishing vehicle detection scheme, we can also make an attempt to read vehicle plate numbers further a much broader attempt to automate a traffic flow system using image recognition and probability distribution.**

Keywords—Vehicle detection; Moving object; Real-time data capture; Image recognition; Vehicle density; Probability distribution

I. INTRODUCTION

Real-time image capture and detection is one of the subjects of research [1-3] which gained focus rather than conventional areas of image processing due to its immense possibility of industrial detection. Research and exploration in this field becomes possible due to the momentum gain in the study of neural network, and a few specific areas are responsible for this [4-6]. Data captured at real-time has immense opportunity to analyze vehicle density, motion detection, and moreover, reading plate numbers of vehicles which will help to collect the revenue from the perspective of Government if they bypass their toll-tax at different cross-over/checking points. Obviously, this method is much fruitful than the images captured by CCTV, and data can be used for multiple purpose. A probability distribution can be generated which will further help for neural network analysis with deep learning, and that data may be used for traffic diversion whenever required, if collected at different time of the day at a single point/node. Among the various algorithms used by neural networks, Yolo algorithm is favored [7-10] for this specific purpose, as it has already proved to be beneficial for several image capturing purposes, and also to extract the data for further analysis [11-12]. In this work, we have extended this concept not only for real-time image capture and data analysis, but also used this data for future density prediction with probability distribution.

II. YOLO ALGORITHM

We use the YOLO algorithm to detect vehicle from images. YOLO (You Only Look Once) can detect any object from an image; we are specifically using it for vehicle detection here. Historically the detection mechanisms are worked considering and analyzing various parts of the images at numerous times with different scales in arbitrary order, and thereafter, image classification methodology is redesigned for object detection. Obviously, these methods are very time consuming and also unproductive, precisely if size the image is very large, and different parts have different resolutions.

Compared to this, our choice of algorithm, i.e., YONO is based on a totally different approach. The most noticeable feature is that it scans the total image only once, and it visits the network only one time. After that, object is detected very easily. This approach speaks in favor of the name, and is properly justified. There are other notable frameworks which can also produced better output such as Faster R-CNN and SSD. These techniques are also used by several researchers.

DNN (Deep Neural Network) is such type of framework which was a part of opencv_contrib repository at the initial level. Just in two years before, it has been transferred to the master branch of opencv repository. This transfer gives user the advantage to make a run of the interface on already trained deep learning modules but inside opencv. One important point in this context may be mentioned that DNN module is never used for the purpose of training. Whenever images or videos are required to run, this module is used.

Prior to DNN modules, Caffe and Torch models were very popular as this model can support various existing libraries and frameworks. One major example in support of the statement is the addition of TensorFlow module. The addition of YONO or Darknet is made recently due to the demand of complex applications with the objective of reducing run-time. In this paper, we have to detect common objects which are seen day-to-day, for which opencv DNN module is used along with trained YONO model.

III. ALGORITHM AND CODE

```
import cv2

import argparse

import numpy as np
```

978-1-7281-0070-8/19/$31.00 ©2019 IEEE

```
#add geodesics
% here we have used the method of ArgumentParser ( )
help = 'path to yolo pre-trained weights')
ap.add_argument('-cl', '--classes', required=True,
help = 'path to text file containing class


names')
args = ap.parse_args()
def get_output_layers(net):
layer_names = net.getLayerNames()
output_layers = [layer_names[i[0] - 1] for i in net.getUnconnectedOutLayers()]
return output_layers
def draw_prediction(img, class_id, confidence, x, y, x_plus_w, y_plus_h):
label = str(classes[class_id])
 color = COLORS[class_id]
cv2.rectangle(img, (x,y), (x_plus_w,y_plus_h), color, 2)
cv2.putText(img, label, (x-10,y-10), cv2.FONT_HERSHEY_SIMPLEX, 0.5, color, 2)
image = cv2.imread(args.image)
……..
…….
Width = image.shape[1]
Height = image.shape[0]
scale = 0.00392
classes = None
with open(args.classes, 'r') as f:
classes = [line.strip() for line in f.readlines()]
COLORS = np.random.uniform(0, 255, size=(len(classes), 3))
---------------------------------------------------
blob = cv2.dnn.blobFromImage(image, scale, (416,416), (0,0,0), True, crop=False)
---------------------------------------------------
outs = net.forward(get_output_layers(net))
class_ids = [ ]
confidences = [ ]
boxes = [ ]
conf_threshold = x [#arbitrary number]
nms_threshold = y [#arbitrary number]
for out in outs:
for detection in out:
scores = detection[n:]
class_id = np.argmax(scores)
confidence = scores[class_id]
if confidence > x:
center_x = int(detection[0] * Width)
center_y = int(detection[1] * Height)
w = int(detection[2] * Width)
h = int(detection[3] * Height)
x = center_x - w / 2
y = center_y - h / 2
……
…….
class_ids.append (class_id)
confidences.append (float(confidence))
boxes.append ([x, y, w, h])
……..
……
indices = cv2.dnn.NMSBoxes(boxes, confidences, conf_threshold, nms_threshold)
for i in indices:
i = i[0]
box = boxes[i]
x = box[0]
y = box[1]
w = box[2]
h = box[3]
draw_prediction(image, class_ids[i], confidences[i], round(x), round(y), round(x+w), round(y+h))
cv2.imshow("object detection", image)
cv2.waitKey()
cv2.imwrite("object-detection.jpg", image)
cv2.destroyAllWindows()
```

IV. RESULTS

These are the following outputs, the image on top is the original image and the image below it is the image with the vehicle and/or people detected.

These results show that the algorithm is capable of detecting vehicles from images but there are certain problems like for instance it captures pictures of vehicles on vehicles to be vehicles. It also captures the passer by peoples, as pedestrians and are also capable of counting them.

Fig 1a: original image of bus

Fig 1b: Detected image of bus

Fig 2a: original image of bus

Fig 2b: Detected image of bus

Fig 3a: original image of bus

Fig 3b: Detected image of bus

Our analysis is justified through the error table where we have computed SNR, PSNR and MSE for all the output images. The result is summarized in table-I.

TABLE I. SNR, PSNR AND MSE CALCULATION FOR OUTPUT IMAGES

Output image	SNR	PSNR	MSE
Fig 1b	11.3924	17.6362	58.8577
Fig 2b	11.7565	17.7471	68.5107
Fig 3b	12.6695	17.863	67.0549

V. CONCLUSION

This algorithm is capable in detecting vehicles with a very high sense of precision and also detects nearby people as pedestrians who are willing to cross roads or are just passerby. SNR of all the detected images are very close, and the same conclusion can also be applied for MSE. Detecting both vehicles and pedestrians in a transportation system is crucial.

REFERENCES

[1] S. Hussmann, A. Hermanski, "Real-time Image Processing of TOF Range Images using a Single Shot Image Capture Algorithm", IEEE International Instrumentation and Measurement Technology Conference Proceedings, 2012

[2] K. Hang, “Real-time Image Acquisition and Processing System Design based on DSP”, 2nd International Conference on Computer and Automation Engineering, 2010

[3] A. Mazare, L. M. Ionescu, A. I. Lita, G. Serban, “Real Time System for Image Acquisition and Pattern Recognition using Boolean Neural Network”, 38th International Spring Seminar on Electronics Technology, 2015

[4] C. C. Nguyen, G. S. Tran, T. P. Nghiem, N. Q. Doan, D. Gratadour, J. C. Burie, C. M. Luong, “Towards Real-Time Smile Detection Based on Faster Region Convolutional Neural Network”, 1st International Conference on Multimedia Analysis and Pattern Recognition, 2018

[5] J. H. Kim, H. G. Hong, K. R. Park, “Convolutional Neural Network-Based Human Detection in Nighttime Images Using Visible Light Camera Sensors”, Sensors, vol. 17, p. 1065, 2017

[6] F. A. Machot, M. Ali, A. H. Mosa, C. Schwarzlmüller, M. Gutmann, K. Kyamakya, “Real-time raindrop detection based on cellular neural networks for ADAS”, Journal of Real-Time Image Processing, pp 1–13, 2016

[7] J. Du, “Understanding of Object Detection Based on CNN Family and YOLO”, IOP Conf. Series: Journal of Physics, 1004, p. 012029, 2018

[8] M. Radovic, O. Adarkwa, Q. Wang, “Object Recognition in Aerial Images using Convolutional Neural Networks”, Journal of Imaging , vol. 3, p. 21, 2017

[9] J. Zhang, M. Huang, X. Jin, X. Li, “A Real-Time Chinese Traffic Sign Detection Algorithm based on Modified YOLOv2 Algorithms”, vol. 10, p. 127, 2017

[10] X. Wu, B. Xu, X. Chen, C. Jin, “Visual Target Detection Based on YOLO Network Structure”, Boletín Técnico, vol. 55(9), pp. 6-13, 2017

[11] J. Lu, C. Ma, L. Li, X. Xing, Y. Zhang, Z. Wang, J. Xu, “A Vehicle Detection Method for Aerial Image based on YOLO”, Journal of Computer and Communications, vol. 6, pp. 98-107, 2018

[12] X. Li, Y. Liu, Z. Zhao, Y. Zhang, L. He, “A Deep Learning Approach of Vehicle Multitarget Detection from Traffic Video”, Journal of Advanced Transportation, vol. 2018, Article ID 7075814, 2018