Vehicle Counting and Traffic Congestion Detection Using Yolov3

Sama Aswith Reddy

Student, Department of Information Technology, J.B. Institute of Engineering and Technology.

ABSTRACT

The purpose of this project is to utilize Python modules and a sophisticated deep learning algorithm to identify, categorize, track, and tally moving vehicles from highway CCTV footage. Additionally, the system predicts traffic congestion by analyzing the number of vehicles in consecutive video frames. When congestion is detected, the application automatically notifies the traffic police who receive a message on their mobile, prompting them to address the existing traffic jam in the area. The project's core involves a vision-based vehicle detection and counting system that relies on the YOLOv3 model and openCv-python library to achieve its objectives.

Keywords: Opencv, YOLOv3, Numpy, Detecting, Tracking, Counting, Frames, Classes.

1. INTRODUCTION

Intelligent vehicle detection, classification, and counting are becoming increasingly important in the field of highway management. This work is carried out to detect and classify the vehicles using the OpenCV module from Python which performs image processing and the pre-trained yolov3 algorithm which performs the detection and classification of vehicles on the images and videos. And later based upon the number of vehicles detected we predict the traffic congestion. This project has dual functionality which includes the prediction of traffic congestion mentioned above and also includes tracking of the vehicles, i.e. giving unique IDs to individual vehicles and counting those individual vehicles. Here counting is based on four types of vehicles, namely cars, motorcycles, trucks, and buses. Finally, it displays the number of different types of vehicles and increments it appropriately as the video progresses.

2. LITERATURE SURVEY

Traffic monitoring systems, such as intelligent traffic light systems [1], vehicle speed monitoring [4], parking lot monitoring [5-6], and monitoring of traffic violations [8], are one area where CV technology has been used for a few tasks. Every previous task begins with determining the location of each vehicle, such as a car. Due to this, the object detection process is crucial in this work. An existing machine learning way, for example, grey image scaling, binarization of images, and background subtraction [2],[3] or sometimes edge detection [1], is required to complete this task. Without a doubt, this method has drawbacks and limitations such as the fact that when the vehicle's shadow appears in the image, the detection may not be completely accurate. If there are changes in the road surface, such as road repair, etc. can also lessen the vehicle detection accuracy because they interrupt the process of image subtraction.

Before commencing the research, we conducted several stages, including literature reviews to analyze past studies and identify existing problems. This helped us determine additional code and functionalities needed to solve these issues and design an efficient system. We utilized YOLOv3 for vehicle detection and classification. During the testing phase, we recorded videos in Full HD resolution from various angles to ensure accurate performance measurement. Performance was evaluated based on the system's accuracy in detecting and counting vehicles. The results were then compared and correlated with the actual number of vehicles counted by humans. This evaluation process helps assess the system's effectiveness and reliability.

3. SYSTEM ARCHITECTURE

The system consists of three main modules: Object Detection, Traffic Congestion Predictor, and Vehicle Counter. The Object Detection module uses YOLOv3 to detect vehicles in each frame of the video and returns coordinates and class IDs. The Traffic Congestion Predictor module determines congestion based on the number of vehicles detected within specified thresholds. The Vehicle Counter module counts and displays different vehicle types in the video by incrementing the count when a vehicle is close to a borderline.

Generally, the machine learning and deep learning projects should be built and developed on GPUs and more specifically GPU from Nvidia i.e. Nvidia Geforce MX150 or even GTX version from Nvidia. But due to unavailability, we are just running it on a CPU system with an i5 processor. The project runs in this system successfully. However, it runs slowly. This CPU takes around 0.3-0.4 seconds to run a single frame.

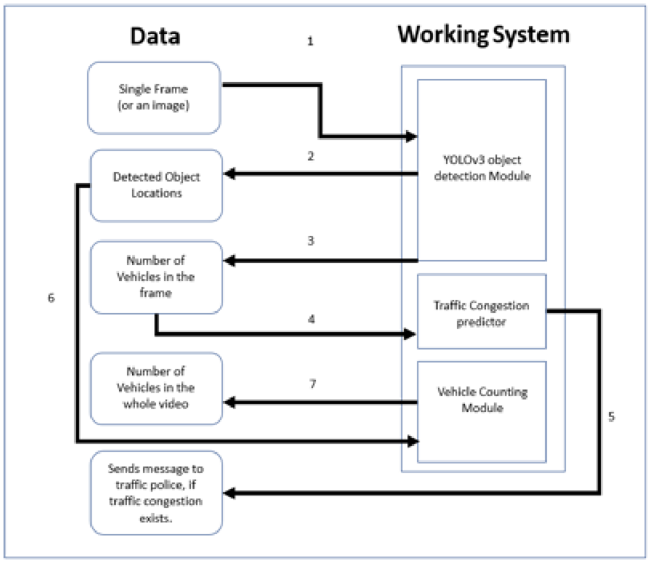

Data
Working System
1
Single Frame (or an image)
2
Detected Object Locations
YOLOv3 object detection Module
3
Number of Vehicles in the frame
6
4
Traffic Congestion predictor
Number of Vehicles in the whole video
7
Vehicle Counting Module
5
Sends message to traffic police, if traffic congestion exists.

Figure 1. The architecture of the vehicle counting and traffic congestion detection system.

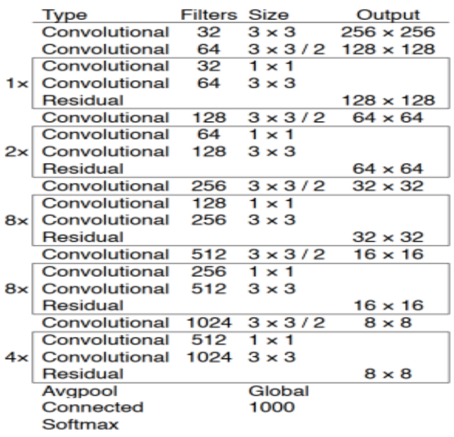
Opencv considers videos as frames where we need to process individual frames to complete a video. Some of the video frames are shown below.

Figure 2. Screenshots of videos

4. YOLOV3 ALGORITHM

CNN plays a central role in object detection in the real world, particularly when used with the OpenCV module for image and video processing. Various CNN architectures are employed in computer vision (OpenCV) to detect and classify objects. For example, "Faster R-CNN" [3], "Region-based Fully CNN" [4], "Regional based fully CNN (R-FCN)" [5], "YOLO" [6], "YOLOv2" [7]. Among many of these detection and classification techniques, you only look once the version 3 (YOLOv3) model is determined as the most suitable algorithm because of its high accuracy.

YOLOv3 is an advanced Deep learning algorithm that employs CNN architecture. It's an improvement over YOLOv2 and is known as Darknet-53 due to its 53 convolutional layers. This version enhances detection accuracy and significantly improves GPU utilization compared to previous models. Moreover, YOLOv3 has the advantage of accepting input images of any size, making implementation easier.



	Type	Filters	Size	Output
	Convolutional	32	3 × 3	256 × 256
	Convolutional	64	3 × 3 / 2	128 × 128
1×	Convolutional	32	1 × 1	
1×	Convolutional	64	3 × 3	
1×	Residual			128 × 128
	Convolutional	128	3 × 3 / 2	64 × 64
2×	Convolutional	64	1 × 1	
2×	Convolutional	128	3 × 3	
2×	Residual			64 × 64
	Convolutional	256	3 × 3 / 2	32 × 32
8×	Convolutional	128	1 × 1	
8×	Convolutional	256	3 × 3	
8×	Residual			32 × 32
	Convolutional	512	3 × 3 / 2	16 × 16
8×	Convolutional	256	1 × 1	
8×	Convolutional	512	3 × 3	
8×	Residual			16 × 16
	Convolutional	1024	3 × 3 / 2	8 × 8
4×	Convolutional	512	1 × 1	
4×	Convolutional	1024	3 × 3	
4×	Residual			8 × 8
	Avgpool		Global	
	Connected		1000	
	Softmax			

Figure 3. Darknet-53 Architecture

Steps of YOLOv3 model:

- Takes a frame from the whole video as input.
- It is passed through the first convolutional layer with arbitrary padding and filter size(kernel).
- The size of the image might get shrunk based on the padding value and filter used.
- When using the pre-trained YOLOv3 model, the weights and biases of different layers are already set to detect the desired vehicles in the given frame. As the image passes through the YOLOv3 model, the activations are generated after each layer based on the pixel intensities of the frame and the neural network's weights and biases.
- These activations are sent as input to the next layer and the next layer in the next step and so on.
- Finally, in the last layer, the activations in terms of probabilities are evaluated based on all those previous different types of layers (convolutional, pooling, fully connected).

- Different type of vehicles has different class IDs and we link those class IDs with the evaluated probabilities to determine the type of the vehicle.
- The same process will be repeated till the full length of the video.

A few detected vehicles are shown in Figure 4, which includes the detection and classification of cars, buses, trucks, and motorcycles.

Vehicle Counting using a non-tracking Approach:

Initially, we used a simple approach to count the number of vehicles in the frames, where we had used a borderline (which can be drawn using opencv) and the vehicle count is incremented only if the centroid of any vehicle is close to that borderline and even classification can also be done using this approach i.e. if the distance between the vehicle and the borderline is less than or equal to some threshold value we consider it as a vehicle and we increment the count of vehicles. The threshold value can be determined from the given video's resolution after few observations.

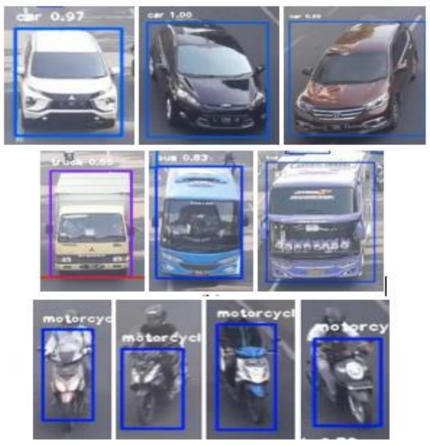


Figure 4. Detection and classification of vehicles

This simple technique of vehicle counting has drawbacks, particularly in cases of slow-moving vehicles due to traffic congestion. Overcounting can occur when the same vehicle appears in multiple close positions to the counting line in consecutive frames. To mitigate this issue, we adopted a different approach, as illustrated in Figure 6. By examining three consecutive frames and considering the centroids' proximity and frame differences, we accurately determine if a vehicle is the same across frames. This improved method ensures more precise and reliable vehicle counting results. This algorithm will surely decrease the error rate of vehicle counting. The drawback mentioned before is no longer present using the current approach.

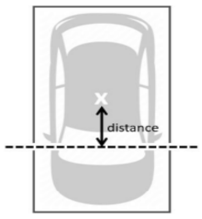


Figure 5. Vehicle Counting based on Centroid-Borderline Distance

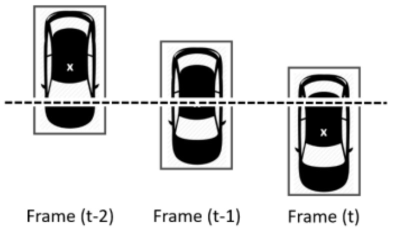


Figure 6. Non-tracking counting of vehicles

The system's performance can be measured by correlating the actual count of vehicles (by humans) and the count which is done by our system. The accuracy in terms of percentage is calculated as shown below:

A= (1-(|real_count - system_count|) /real_count) * 100%

where A is the accuracy and real_count is the actual number of vehicles counted by humans and system_count is the number of vehicles counted by the system.

Final Result and Debate

The traffic congestion detection and vehicle counting system developed as part of this project uses various videos as said before. Actually, the videos taken are of 1080p resolution with 30 frames per second as the rate. As we know, we use a pre-trained YOLO version 3 model, so training the neural network is not necessary.

In the first scenario, we used two different videos with full HD resolution and 30 frames per second as the ratio. This means the video is divided into 6000 frames approximately. In Figure 7 we can see an example of a detected frame and also counting.

This system achieved an accuracy of 95% which is so far the highest, in the 1st video, which was recorded by us from the front side, as shown in Table 1. The types of motorcycle and car have the most counting accuracy with only 8 percent and 0 percent of error on average across all videos, respectively. On the other hand, the truck has really bad accuracy, and even the bus too.

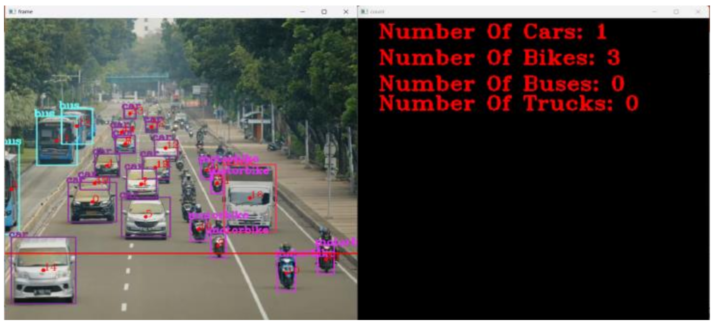

Number Of Cars: 1
Number Of Bikes: 3
Number Of Buses: 0
Number Of Trucks: 0

Figure 7. Vehicle Detection, Classification, and Counting

Table 1: First Scenario Results

Video	Vehicle	Counting		Counting Error	Overall Accuracy
		Real	System		
Video 1	Car	41	41	0	95.33%
	Motorbike	24	25	+1	
	Bus	9	11	+2	
	Truck	8	11	+3	
Video 2	Car	27	29	+1	91.21%
	Motorbike	34	35	+1	
	Bus	12	14	+2	
	Truck	10	14	+4	
				Average Accuracy:	93.27%

In the second attempt employed, 15 frames per second as the ratio with the same resolution as the first. As explained in Table 2, the counting accuracy decreases by approximately 10% and 5% in the first and second videos respectively. Here the second video was the winner with 86.7%.

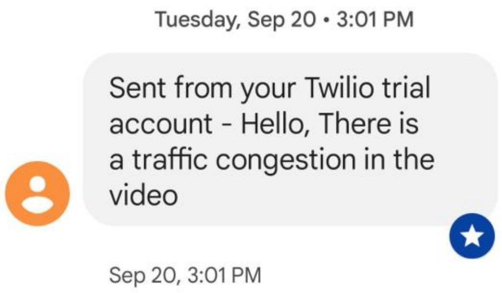
The proposed system performs best when using a video with the highest frame-per-second ratio. A higher frame-per-second enhances overall performance by processing more data, while a lower frame-per-second ratio may result in data loss. Car detection and counting are more accurate compared to other vehicles, even with a lower frame-per-second ratio. The YOLOv3 model excels at accurately recognizing cars, including small car objects located far from the camera (Figure 7). However, we observed a flaw in the system, where a few cars are detected as two different types of vehicles simultaneously, leading to the overcounting of the number of trucks in both test cases. This issue should be addressed for further improvements.

Table 2: Second Scenario Results

Video	Vehicle	Counting		Counting Error	Overall Accuracy
		Real	System		
Video 1	Car	41	39	-2	86.58%
	Motorbike	24	18	-6	
	Bus	9	10	+1	
	Truck	8	10	+2	
Video 2	Car	27	24	-3	86.7%
	Motorbike	34	29	-5	
	Bus	12	13	+1	
	Truck	10	12	+2	
				Average Accuracy:	86.64%

The second part of the project is traffic congestion detection, which can be seen in Figure 8. If the congestion exists for a specific amount of time (usually between 5-10 min, which varies depending upon the width and type of the road), then a message will be sent to the respective traffic police mobile as shown in Figure 9.

Figure 8. Vehicle Detection, Classification, and Counting


Tuesday, Sep 20 • 3:01 PM

Sent from your Twilio trial account - Hello, There is a traffic congestion in the video

Sep 20, 3:01 PM

Figure 9: Message in mobile phone.

5. CONCLUSION

This system utilizes YOLOv3 without vehicle tracking to achieve vehicle counting and traffic congestion detection. The counting is based on the distance between the vehicle's bounding box centroid and the borderline, resulting in a high accuracy of 95.33% with front-side videos. YOLOv3 plays a crucial role in vehicle detection, particularly for cars, which have the highest accuracy among all vehicles. The frame-per-second ratio significantly impacts the project's performance. Overall, the system demonstrates excellent accuracy in its functions. However, future improvements are necessary to enhance the system further.

6. FUTURE ENHANCEMENTS

This work can be used in CCTV surveillance rooms where the live CCTV footage can be used for traffic management, which is really crucial in today’s world due to the increased number of vehicles on highways. Even this project can be used to predict traffic congestion, which might be due to any unfortunate accident or any vehicle failures on the highways. And also the traffic congestion existing in that area can be cleared once the traffic police receive the message.

7. REFERENCES

1. J. T. G. Nodado, M. A. P. Abugan, A. C. Aralar, and H. C. P. Morales, “Intelligent Traffic Light System Using Computer Vision with Android Monitoring and Control,” TENCON 2018 - 2018 IEEE Reg. 10 Conf., no. October, pp. 2461–2466, 2018.
2. A. J. Kun and Z. Vámossy, “Traffic Monitoring with Computer Vision,” in 7th Int’l Symposium on Applied Machine Intelligence and Informatics, 2009, pp. 131–134.

3. Z. Iftikhar, P. Dissanayake, and P. Vial, “Computer Vision Based Traffic Monitoring System for Multi-track Freeways,” in 2014 Int’l. Conf. on Intelligent Computing, 2014, pp. 339– 349.
4. Krishna, M. Poddar, M. K. Giridhar, and A. S. Prabhu, “Automated Traffic Monitoring System Using Computer Vision,” in 2016 Int’l. Conf. n ICT in Business Industry & Governance, 2016.
5. T. Paula, C. Florina, R. Brad, L. Br, and M. Greavu, “An Image Feature-Based Method for Parking Lot Occupancy,” Futur. Internet, vol. 11, no. 169, pp. 1–17, 2019.
6. T. Fabian, “A Vision-Based Algorithm for Parking Lot Utilization Evaluation Using Conditional Random Fields,” in 2013 Int’l Symposium on Visual Computing, 2013, pp. 222– 233.
7. W. Wu, O. Bulan, E. A. Bernal, and R. P. Loce, “Detection of Moving Violations,” Comput. Vis. Imaging Intell. Transp. Syst., vol. 1, pp. 101–130, 2017.