Real-time traffic monitoring and traffic offense detection using YOLOv4 and OpenCV DNN

Conference Paper · December 2021

DOI: 10.1109/TENCON54134.2021.9707406

4 authors:

Fahimul Hoque Shubho
North South University
5 PUBLICATIONS **30** CITATIONS
SEE PROFILE

Fahim Iftekhar
North South University
1 PUBLICATION **26** CITATIONS
SEE PROFILE

Ekhfa Hossain
North South University
2 PUBLICATIONS **35** CITATIONS
SEE PROFILE

Shahnewaz Siddique
North South University
50 PUBLICATIONS **234** CITATIONS
SEE PROFILE

Real-time traffic monitoring and traffic offense detection using YOLOv4 and OpenCV DNN

Fahimul Hoque Shubho, Fahim Iftekhar, Ekhfa Hossain, Shahnewaz Siddique
Department of Electrical and Computer Engineering, North South University, Dhaka-1229, Bangladesh
fhshubho@gmail.com, fahimiftekhar55@gmail.com, ekhfahossain@gmail.com, shahnewaz.siddique@northsouth.edu

***Abstract*— This paper presents a computer vision-based system for traffic offense detection. The system detects traffic offenses such as speed limit violations, unauthorized vehicles, traffic signal violations, unauthorized parking, wrong-way driving, and motorbike riders without helmets. The traffic offense detection system consists of a pipeline of four different modules. These are a vehicle detection module, a vehicle classification module, a vehicle tracking module, and a traffic offense detection module. Vehicles on the roads are detected in the vehicle detection module using visual data such as live camera feed. Next, after the vehicles are detected, they are classified into different classes using a vehicle classification module. A vehicle tracking module is developed to track the vehicle as it moves through the traffic. Lastly, we have implemented a traffic offense detection module that analyzes traffic patterns and detects different types of traffic violations in real-time. The entire system is implemented using OpenCV Deep Neural Network (DNN) module. We have used YOLOv4 to detect vehicles on the roads with high accuracy. For motorbike riders without helmets, we have used a fast YOLOv4-tiny model. The DeepSORT algorithm is used to track vehicles in real-time. Obtained accuracies are 86% in YOLOv4 for Vehicle and 92% in YOLOv4-tiny for Helmet Detection.**

Keywords*— *Computer Vision, Object Detection, Vehicle Detection, Vehicle Tracking, Traffic Offense Detection, YOLOv4, YOLOv4-tiny, OpenCV, Deep Neural Network.

I. INTRODUCTION

Accidents on the road are a big concern in the modern world. Many countries spend millions of dollars on reducing road accidents every year. Various road safety methods and measures are in place to prevent road users from being seriously injured or killed. Still, countries are facing a considerable number of accidents on the roads every day. Most of these accidents are linked to people not willing to follow the traffic rules on the road and lack of monitoring. Research has shown that improvements to road infrastructure, particularly design standards that consider all road users' safety, are critical to making roads safe. [1] shows that 112 countries have national design standards for the management of speed. But infrastructure alone cannot make the roads safer without monitoring and adherence to traffic rules. Today artificial intelligence techniques in computer vision and machine learning offer fascinating and promising solutions to improving traffic monitoring and increasing road safety.

In this research, we use modern computer vision and machine learning techniques to identify and classify vehicles and track them while continually assessing traffic parameters for potential traffic offenses. Our suggested approach is more efficient since it avoids the clutter of having separate models for various objectives. Instead, just one model detects, classifies, tracks, and assigns traffic violations, if any, in real-time, while a second model keeps an eye out for special cases. This method ensures that the system operates in a timely, clutter-free, efficient, and practical manner. Our system starts with obtaining a video feed from a traffic camera, then detecting vehicles on the road and tracking them to analyze the scenario to detect possible traffic offenses. The video feed from the video camera is first analyzed with an object detection model we trained with YOLOv4 (You Only Look Once) [2] to detect and classify the vehicles on the road. Our proposed, implemented model has a mean average precision of 87.19% in detection and vehicle classification. The YOLOv4-tiny model that we trained from scratch has shown an incredible 98.1% mean average precision and is developed to detect riders without helmets.

II. LITERATURE REVIEW

Traffic management must rely on a system for estimating traffic parameters in real-time. Currently, there is a lot of work going on to detect traffic parameters in the research field. Ou et al. [3] proposed parallel process techniques to detect traffic offenses. Their work implemented real-time traffic violation detection in a monitoring stream using simultaneous video streams from multiple cameras. The system was designed to detect three types of vehicles- speeding vehicles, blacklisted vehicles, and plate-cloned vehicles. The system was implemented and evaluated using both real and synthetic traffic data.

Beymer et al. [4] proposed a model that deals with traffic and lighting conditions while using segmentation, classification, and tracking methodologies. The model focuses on vehicle segmentation and tracking and the computation of traffic parameters from tracking data. They have used a feature-based traffic tracking approach to track vehicles under congestion. The system tracks vehicle sub-features, which makes it less susceptible to partial occlusion. A network of C40 DSP (Digital Signal Processor) chips linked to a host PC has been used to construct a real-time version of the system.

Wang et al. [5] proposed an approach to use an enhanced background-updating algorithm and feature-based tracking method. In their paper, they have used video-based traffic detection through an improved background-updating algorithm, then tracking the moving vehicles by a feature-based tracking method.

Zhu et al. [6] proposed a vehicle detection and tracking volume statistics algorithm based on an improved single Gaussian model. The algorithm has three sections: moving target detection, shadows suppression, and traffic volume count. The single Gaussian model detects moving vehicles while shadow suppression is performed in the RGB feature space. Traffic volume was calculated in the virtual lanes.

Deng et al. [7] proposed a model that can be used for object segmentation, classification, and tracking methodologies to know the real-time measurements in urban roads. The model used the background subtraction method to detect stationary or slow-motion objects. Experimental results showed a diminishing 20% degradation of infrastructure capacities.

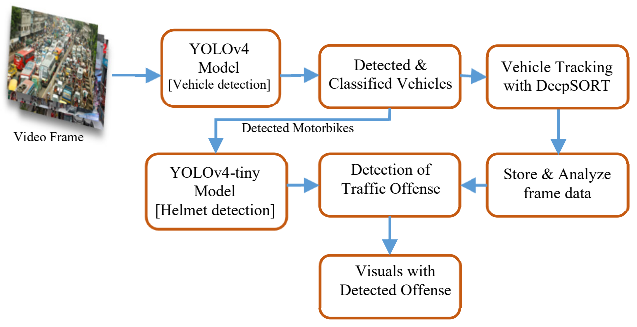
Wen-juan et al. [8] proposed a model that can locate and track the vehicles in a video and calculate the traffic flow, queue length, queue waiting time, the average speed, and other vital parameters. The model used online learning mechanisms and optical flow to track objects. Experimental results showed the accuracy of daytime detection was 98%, and nighttime detection was 92%. The average speed detection of daytime was 95%, while nighttime was 90%.

Kim et al. [9] compared different Artificial Neural Network (ANN) models for real-time vehicle type recognition. They compared deep learning-based object detection models R-CNN (Region Based Convolutional Neural Network), Fast R-CNN, Faster R-CNN, YOLO, and SSD (Single Shot Detector) in processing speed and accuracy for best performance. Faster R-CNN had less FPS (frames per second) with better accuracy, while SSD had less accuracy with better FPS. The YOLO model was a middle ground.

Zinchenko et al. [10] proposed an Artificial Neural Network (ANN) model based on Warren McCulloch and Walter Pitts' work on the simulation of nervous activity. It takes a similar approach to Kim et al. [9] in using a faster and effective model. The proposed system's primary aim is to deliver better outcomes in decreased pedestrian and vehicle waiting times, shorter travel times, and higher average vehicle velocity. The best result of vehicle detection was 94.65%, and the worst result was 88.71%.

III. Methodology

Our system monitors and detects traffic offenses by processing the video feed from traffic cameras in several stages. An object detection model first processes the video feed from the traffic camera to detect and classify the vehicles present in the video. Detected vehicles are tracked between frames, and all the available information from each frame of the video is stored temporarily and analyzed to detect any possible traffic offense. An additional object detection model processes every detected motorbike to monitor if the riders are wearing helmets. All detected offenses are stored as a visual snap for further use. Detected traffic offenses are- Overspeed Detection, Unauthorized Vehicles Detection, Traffic Signal Violation, Unauthorized Parking, Wrong-Way Driving, Motorbike Riders without Helmet. Fig. 1 represents the overview of our proposed system.



Video Frame
YOLOv4 Model [Vehicle detection]
Detected & Classified Vehicles
Vehicle Tracking with DeepSORT
Detected Motorbikes
YOLOv4-tiny Model [Helmet detection]
Detection of Traffic Offense
Store & Analyze frame data
Visuals with Detected Offense

Fig. 1. Overview of the system.

A. *Vehicle Detection, Classification and Tracking*

Detecting vehicles from the video feed of traffic cameras and classifying them is a crucial part of this system. A good amount of data and an efficient object detection model are required for this part. We decided to use multiple datasets and train YOLOv4 object detection models for detection and classification on our custom dataset.

1) Dataset: A combined dataset consisting of vehicles images of Bangladesh from different sources were used. Labeled images from the PoribohonBD dataset [11] and the Dhaka-AI dataset [12] were used. Additionally, publicly available images from online were labeled and added to the dataset. The combined dataset contained 11,210 images with 46,662 labeled objects of 17 different types of vehicles. The types of vehicles are bus, rickshaw, motorbike, bike rider without helmet, car, three-wheelers, pickup, microbus, van, truck, bicycle, police car, ambulance, human hauler (leguna), wheelbarrow, auto-rickshaw, army vehicle, and horse cart. This dataset was used to train the object detection model to detect and classify the vehicles from the video feed at the first stage of the system.

Another dataset, the helmet dataset from Kaggle [13], containing 764 labeled images, was combined with an additional 502 images collected and labeled by us into a dataset of 1,266 labeled images of different helmet types. This dataset was used to train an additional object detection model to detect helmets on bike riders. Figure 2 shows the vehicle count of each class in the combined dataset images.

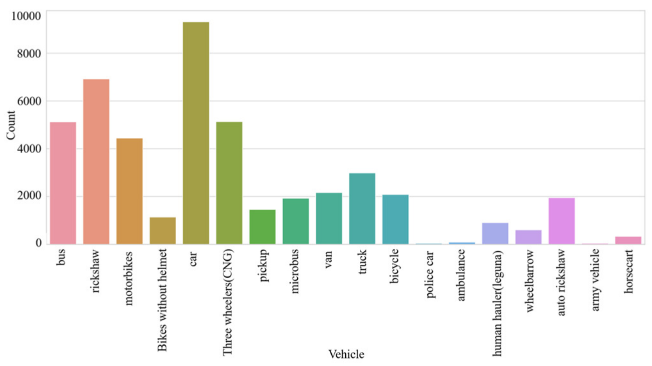


Fig. 2. Vehicle counts in the combined dataset images.

2) Data pre-processing: To prepare the dataset for object detection model training, the image labels were unified in the combined dataset. The ProibohonBD dataset and Dhaka-AI dataset had different image labels for the same class of vehicles. We prepared a list of labels for 17 types of vehicles and converted the image labels from both datasets according to it to unify them. We used a python script to unify the labels. We also labeled the images collected online according to the list. Table I shows the data difference of data labels in the datasets and the unified label for our combined dataset.

Some vehicle types such as SUV and taxi were separately labeled in the Dhaka-AI but were labeled as cars in the PoribohonBD dataset. As there were not enough images of SUVs and taxis to treat them as two separate classes, we decided to merge them into car images and modified their labels as cars in the combined dataset. Some vehicles were of the same type but labeled differently in the two datasets. As the same vehicle had multiple names locally and in datasets, we kept one name as label and merged them. We did the same for a few other labels, and it is shown in Table I.

TABLE I. IMAGE LABELS BETWEEN THE DATASETS

Class labels PoribohonBD Dataset	**Class labels Dhaka-AI Dataset**	**Class labels Combined Dataset**
Bicycle	bicycle	bicycle
Bus	bus	bus
-	minibus	bus
Car	car	car
-	SUV	car
-	taxi	car
-	minivan	microbus
-	ambulance	ambulance
Van		van
CNG	three-wheelers(CNG)	three-wheelers(CNG)
Easy-bike	auto-rickshaw	auto-rickshaw
Horse-cart	-	horsecart
Leguna	human hauler	human hauler(leguna)
Motorbike	motorbike	motorbike
-	scooter	motorbike
Rickshaw	rickshaw	rickshaw
Truck	truck	truck
Wheelbarrow	wheelbarrow	wheelbarrow
-	police car	police car
-	pickup	pickup
-	army vehicle	Army vehicle

We didn't include vehicle images of the boat and launch from the PoribohonBD dataset as these are not seen on the road. The PoribohonBD dataset and Dhaka-AI dataset had image labels in '.xml' format. To use the images from datasets labels were converted to YOLO supported '.txt' format using python script. Our collected images from online were labeled in YOLO-supported '.txt' format. We followed the same procedure for our helmet detection dataset to convert the labels into YOLO supported format.

3) Object Detection Model: For such a monitoring system, it is critical to have an object detection model that can perform efficiently while providing enough speed to monitor the traffic in real-time and accuracy to classify the vehicles with acceptable precision. YOLOv4 provides acceptable speed and accuracy in detecting objects and classifying them. A YOLOv4 model and a YOLOv4-tiny model were trained with the combined dataset of vehicle images of Bangladesh to detect and classify the vehicles from the video feed of traffic cameras. Another YOLOv4-tiny model was trained with the second dataset to detect helmets from the motorbike riders that are detected by the previous model. YOLOv4-tiny was chosen to detect helmets on bike riders for its speed, this sustained the speed and efficiency of the whole system while using two object detection models at the same time.

All the models were trained using open source Neural Network framework Darknet [14] in Google Colab. Parameters modified for training YOLOv4, and YOLOv4-tiny object detection models on our dataset are shown in Table II. Batches refer to the number of images loaded for each iteration during training and split into the number of subdivisions to process in the GPU. Max batches are the number of iterations during training for each model. It is calculated as classes multiplied by 2000. There are 17 classes of vehicles in our combined dataset. The number of iterations for YOLOv4 and YOLOv4-tiny models are 17*2000 or 34000, respectively. For the helmet detection model, we have set the number of iterations to 6000 as the minimum number of iterations in YOLOv4 models is required to be more or equal to 6000. Steps are the number of iterations at which learning rates are changed. It is set to 80% and 90% of the max batches for each model. Network size refers to the resized input image for the model. The number of filters is calculated as follows-

$$\text{filters} = (\text{classes} + 5) \times 3 \qquad (1)$$

For our vehicle classification models, the number of filters is (17+5)*3 or 66. For the helmet detection model, it is (2+5)*3 or 21.

TABLE II. MODIFIED PARAMETERS FOR TRAINING YOLOV4 AND YOLOV4-TINY OBJECT DETECTION MODELS

Parameters	**YOLOv4-tiny Vehicle Detection Model**	**YOLOv4 Vehicle Detection Model**	**YOLOv4-tiny Helmet Detection Model**
classes	17	17	2
batch	64	64	64
subdivisions	16	32	16
max_batches	34000	34000	6000
steps	27200, 30600	27200, 30600	4800, 5400
Network size (width, height)	416, 416	608, 608	608, 608
filters	66	66	21

4) Vehicle Tracking: Vehicles that are detected by YOLOv4 object detection models are tracked throughout the video frames. When a vehicle enters into the video frame for the first time, it is given a unique id and tracked between frames using DeepSORT [15]. Each vehicle's id and position in each frame are stored temporarily and analyzed for possible traffic offenses. We have implemented the DeepSORT algorithm in our system following the procedure of [16].

B. Traffic Offense Detection

1) Overspeed Detection from Speed Estimation: The speed of each detected vehicle is estimated as shown in Fig. 3 by the change in their position.

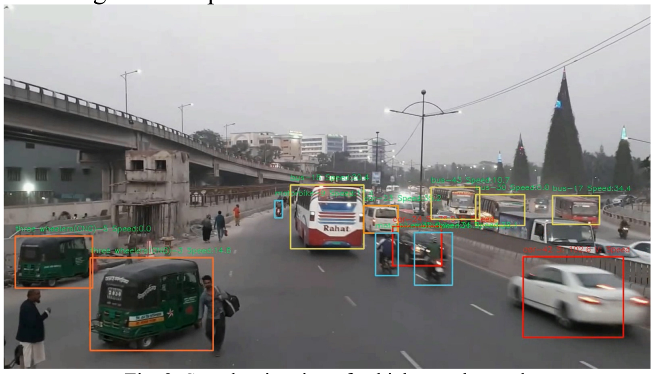


Fig. 3. Speed estimation of vehicles on the road.

Speed is estimated when a vehicle stays at least five frames in the video. The change of vehicle's position in the x-axis and y-axis is considered for five consecutive frames, and speed is estimated in terms of pixel per second. When the total area covered by the traffic camera is available to the system, it can also calculate the estimated speed in terms of meter per second. When a vehicle's estimated speed goes over the allowed speed, it is visually marked on the video frame and logged for overspeeding.

2) Unauthorized Vehicles Detection: Unauthorized vehicles are detected in the first stage when the video feed goes through the YOLOv4 model. Any detected vehicle is classified as rickshaw, bicycle, van, auto-rickshaw, wheelbarrow, or horse cart; it is immediately marked and logged as an unauthorized vehicle that is not permitted to be on the main roads. Fig. 4 shows a bicycle and several rickshaws being marked on the video frame as unauthorized on the main street of Dhaka, Bangladesh.

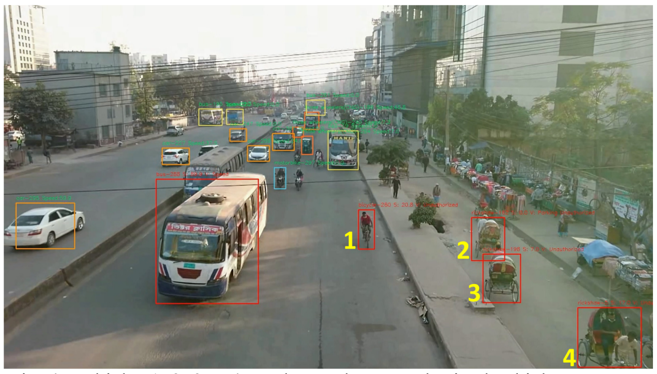


Fig. 4. Vehicles 1, 2, 3 & 4 are detected as unauthorized vehicles on street.

3) Traffic Signal Violation: The traffic signal is synchronized with our system, and it monitors the road intersection points for a possible signal violation when the traffic light indicates red. A visual line on the intersection point of the road works as a reference for our system, and when any vehicle crosses the line while the traffic light indicates red, it is marked and logged for traffic signal violation. The vehicle remains marked even if it reverses back after violating the traffic signal. Fig. 5 shows an example of a traffic light violation detected by the system on a video collected from YouTube.

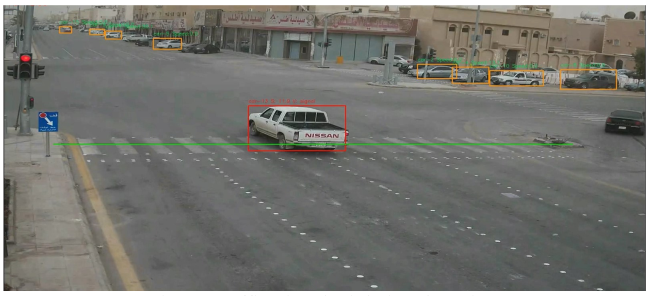


Fig. 5. Traffic Signal Violation detection.

4) Unauthorized Parking: Vehicles that stand still on the main road and cause roadblocks are marked and logged for unauthorized parking. All the vehicle's IDs and positions from the previous frames are stored in the system. The current position of each detected vehicle is compared with the positions they were on previous frames to detect how long the vehicles stood still in the same places. Depending on the time the vehicle is standing idle on the street, the vehicle is marked for unauthorized parking. Figure 6 shows a bus standing idle on the main street is being marked on the video frame for unauthorized parking.

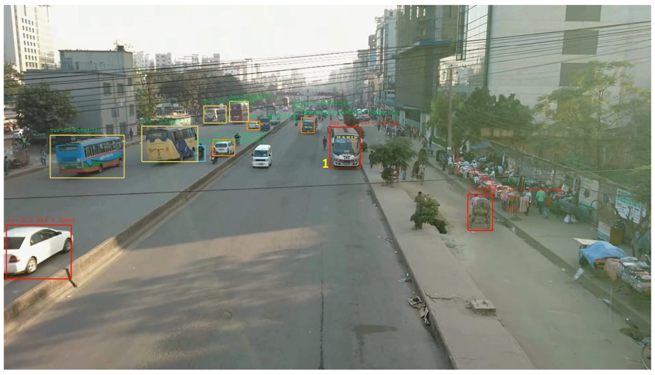


Fig. 6. Vehicle 1 is detected for unauthorized parking on the street.

5) Wrong Way Driving: Vehicles that drive in the wrong direction of the road are detected by our system from their movement. Fig. 7 demonstrates such a situation where the system treats all the lanes as one-directional and all the vehicles moving into opposite directions are marked for wrong-way driving. The direction of driving on the road is predefined in our system. It can be changed depending on the road that is being monitored through the system. Our system checks the change of position of every vehicle on the x-axis and y-axis. Change of positions along the x-axis of the video frame indicates in which direction the vehicles are taking turns. And the changes in position along the y-axis of the video frame indicates in which direction the vehicles are moving forward. From the change of position along both axis, our system detects when vehicles move in the wrong direction.

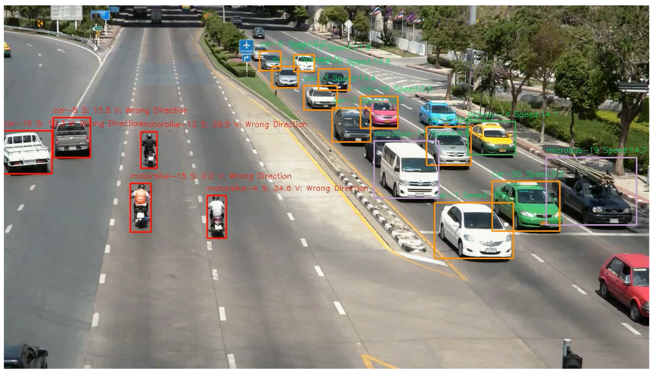


Fig. 7. Wrong way driving detection.

6) Motorbike Riders without Helmet: Our system detects motorbike riders without a helmet in four steps, as illustrated in Fig. 8.

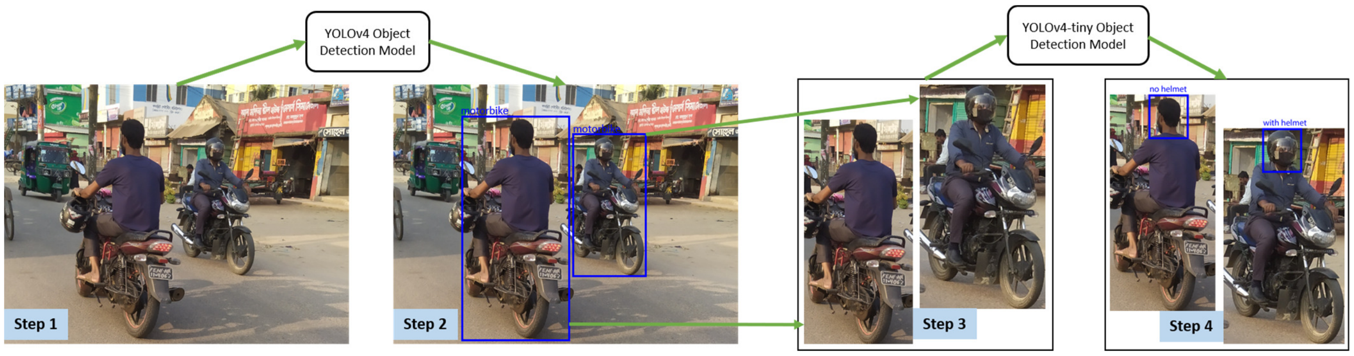

YOLOv4 Object Detection Model
YOLOv4-tiny Object Detection Model
motorbike
motorbike
no helmet
with helmet
Step 1
Step 2
Step 3
Step 4

Fig. 8. Steps of detecting helmet on motorbike riders.

In the first two steps, every motorbike is detected and classified along with the riders as a motorbike. In step 3, the portion of the video frame that contains the detected area of motorbikes are separated as individual images, and they go through the YOLOv4-tiny model in step 4. The YOLOv4-tiny model detects helmets on the riders of the motorbikes images passed to it. If no helmet is detected, our system labels it as a bike rider without helmet and marks it in the video frame for riding a motorbike without helmet. Running two object detection models, one for detecting the motorbike and another one to detect the helmet, increases time complexity. To tackle this complexity, YOLOv4-tiny is used for helmet detection. YOLOv4-tiny model is faster than a regular YOLOv4 model, but the trade-off is accuracy.

IV. Result and Discussion

Maintaining reasonable accuracy and speed was crucial while implementing the system as computer vision-based techniques are computationally expensive. Our system can run on both CPU and GPU. But the GPU is required to get minimum speed to monitor the traffic in real-time. We have implemented and tested the system with OpenCV[17] DNN (Deep Neural Network) module on an Nvidia 1050Ti 4GB GPU. The OpenCV library does not support GPU by default, and it was compiled from source code to enable GPU support.

The first stage of our system is crucial for detecting traffic offenses. The accuracy in this stage in detecting and classifying vehicles affects how accurately the vehicles' positions and classes are being logged and analyzed for the possible traffic offenses. We have trained and tested one YOLOv4-tiny model and one YOLOv4 model to detect and classify vehicles in the first stage of our system. The YOLOv4-tiny model produced 24~30 FPS on our setup and showed 72.02% mean average precision. While using the YOLOv4 model in the first stage of the system, our system was able to process the video feed from the traffic camera at 15~20 FPS speed with our setup. It was slower than the YOLOv4-tiny model but showed better accuracy. The YOLOv4 object detection model showed 86% average precision in detecting and classifying the vehicles. Table III shows the comparison of class-wise average precision of the YOLOv4-tiny model and YOLOv4 model in the detection and classification of the vehicles.

The accuracy in classifying vehicles differed with the number of samples of each class present in the dataset. The lower number of samples of police cars and ambulances in the dataset reduced the average precision of the YOLOv4-tiny and YOLOv4 models in detecting and classifying them. Table III shows that though the YOLOv4 model produces less FPS, it outperforms YOLOv4-tiny in average precision in most vehicle classes. So, we have used the YOLOv4 model in the first stage of our system for its better accuracy.

TABLE III. Class-wise average precision of YOLOv4-tiny model and YOLOv4 model.

Class Labels	YOLOv4-tiny Model's Average Precision	YOLOv4 Model's Average Precision
bus	75.82%	95.40%
rickshaw	69.17%	92.66%
motorbike	69.13%	92.61%
car	74.48%	94.11%
three wheelers(CNG)	79.52%	96.38%
pickup	61.27%	91.27%
microbus	57.28%	89.48%
van	88.95%	96.97%
truck	84.62%	96.55%
bicycle	77.73%	94.32%
police car	21.30%	17.83%
ambulance	21.17%	51.75%
human hauler (leguna)	89.55%	97.58%
wheelbarrow	88.80%	94.50%
auto rickshaw	86.67%	97.47%
army vehicle	67.46%	75.27%
horsecart	92.35%	97.04%

Our second object detection model used in the system is a YOLOv4-tiny model for detecting helmets from motorbikes that are detected in the first stage of the system by the previous model. Fig. 9 shows the mean average precision and validation loss over iteration during training YOLOv4-tiny model.

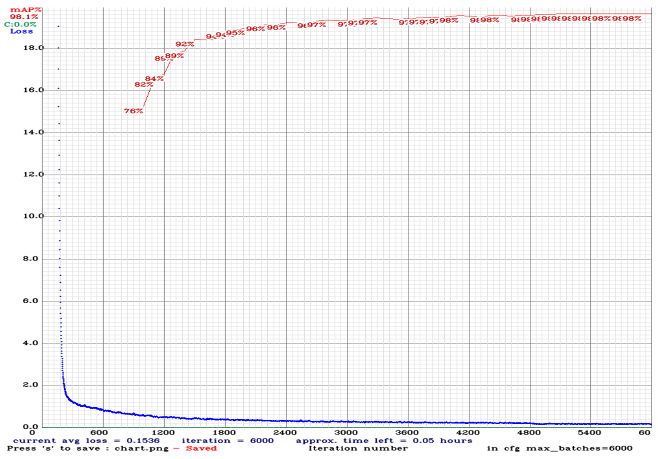


Fig. 9. mAP vs loss during training YOLOv4-tiny model for helmet detection.

This model achieved 98.1% mean average precision in detecting helmets with 6000 iterations in training. The YOLOv4-tiny model ran faster than the YOLOv4 model and maintained the speed of the system.

We have tested the system with several video footage we captured from the streets of Dhaka and collected from YouTube. In our observations, the efficiency of the system depended on the accuracy of the object detection models. Table IV shows performance metrics for all the three models we have tested in our system. Precision shows the accuracy of the models, recall refers to the actual positives identified by the models, f1-score refers to the balance between precision and recall, and the last metric, mAP@0.50 refers to the mean average precision with a threshold of 0.50 intersection-over-union. Replacing the YOLOv4-tiny model with the YOLOv4 model for detecting and classifying vehicles increased the detection efficiency of the traffic offense. With the YOLOv4 model for detecting vehicles and the second YOLOv4-tiny model for detecting helmets, the system analyzed the information from the object detection models and detected most of the traffic offenses it was intended to. An increased number of vehicles reduced the speed of the system and resulted in an FPS drop, which might be improved by running the system on better hardware.

TABLE IV. PERFORMANCE METRICS OF THE MODELS

Class Labels	**Precision**	**Recall**	**F-1 score**	**mAP @0.50**
YOLOv4-tiny for Vehicle Detection	69%	72%	71%	72.07%
YOLOv4 for Vehicle Detection	86%	92%	89%	87.19%
YOLOv4-tiny for Helmet Detection	92%	98%	95%	98.1%

V. Conclusion

In this paper, a computer vision-based road safety system has been presented. We performed real-time monitoring to get information of vehicles along with their various traffic offenses (Over speeding, unauthorized vehicles, signal violation, unauthorized parking, wrong-way driving, riders without helmets) if any. We investigated the vehicle detection, classification, and tracking methodologies as a guiding point of a road safety system. Existing traffic CCTV cameras can be used to capture live footages and then fed into a central server for further analysis with our system. In future work, better hardware and models with higher accuracy level can be integrated with the proposed system. It can be helpful for both traffic monitoring and automatic traffic offense detection.

References

[1] Global status report on road safety 2018. Geneva: World Health Organization; 2018. Licence: CC BY-NC-SA 3.0 IGO.
Retrieved from https://www.who.int/publications/i/item/9789241565684

[2] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, “YOLOv4: Optimal speed and accuracy of object detection,” 2020. [Online]. Available: arXiv:2004.10934

[3] G. Ou, Y. Gao and Y. Liu, "Real-Time Vehicular Traffic Violation Detection in Traffic Monitoring Stream," 2012 IEEE/WIC/ACM International Conferences on Web Intelligence and Intelligent Agent Technology, 2012, Macau, China, December 4-7, 2012. pp. 15-19, doi: 10.1109/WI-IAT.2012.91.
Available: https://ieeexplore.ieee.org/document/6511640

[4] D. Beymer, P. McLauchlan, B. Coifman and J. Malik, "A real-time computer vision system for measuring traffic parameters," Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 1997, San Juan, PR, USA, June 17-19, 1997. pp. 495-501, doi: 10.1109/CVPR.1997.609371. Available: https://ieeexplore.ieee.org/document/609371

[5] Xiaoling Wang, Li-Min Meng, Biaobiao Zhang, Junjie Lu and K. -. Du, "A video-based traffic violation detection system," Proceedings 2013 International Conference on Mechatronic Sciences, Electric Engineering and Computer (MEC), 2013, Shenyang, China, December 20-22, 2013. pp. 1191-1194, doi: 10.1109/MEC.2013.6885246. Available: https://ieeexplore.ieee.org/document/6885246

[6] H. Zhu, C. Xu and F. Li, "The Traffic Volume Count Algorithm Based on Computer Vision," 2013 5th International Conference on Intelligent Human-Machine Systems and Cybernetics, 2013, Hangzhou, China, August 26-27, 2013. pp. 130-133, doi: 10.1109/IHMSC.2013.38. Available: https://ieeexplore.ieee.org/document/6643850

[7] L. Y. Deng, N. C. Tang, Dong-liang Lee, Chin Thin Wang and Ming Chih Lu, "Vision based adaptive traffic signal control system development," 19th International Conference on Advanced Information Networking and Applications (AINA'05) Volume 1 (AINA papers), 2005, Taipei, Taiwan, March 28-30, 2005. pp. 385-388 vol.2, doi: 10.1109/AINA.2005.342. Available: https://ieeexplore.ieee.org/document/1423717

[8] X. Wen-juan and L. Jian-feng, "Application of vision sensing technology in urban intelligent traffic control system," 2018 4th International Conference on Computer and Technology Applications (ICCTA), 2018, Istanbul, Turkey, May 3-5, 2018. pp. 74-77, doi: 10.1109/CATA.2018.8398659.
Available: https://ieeexplore.ieee.org/document/8398659

[9] J. -a. Kim, J. -Y. Sung and S. -h. Park, "Comparison of Faster-RCNN, YOLO, and SSD for Real-Time Vehicle Type Recognition," 2020 IEEE International Conference on Consumer Electronics - Asia (ICCE-Asia), 2020, Seoul, South Korea, November 1-3, 2020. pp. 1-4, doi: 10.1109/ICCE-Asia49877.2020.9277040.
Available: https://ieeexplore.ieee.org/document/9277040

[10] V. Zinchenko, G. Kondratenko, I. Sidenko and Y. Kondratenko, "Computer Vision in Control and Optimization of Road Traffic," 2020 IEEE Third International Conference on Data Stream Mining & Processing (DSMP), 2020, Lviv, Ukraine, August 21-25, 2020. pp. 249-254, doi: 10.1109/DSMP47368.2020.9204329. Available: https://ieeexplore.ieee.org/document/9204329

[11] S. Tabassum, S. Ullah, N. H. Al-Nur, and S. Shatabda, “Poribohon-BD. V2”, October 1, 2020, Distributed by Mendeley Data. Available: https://data.mendeley.com/datasets/pwyyg8zmk5/2

[12] ASM Shihavuddin, M. R. A. Rashid, “DhakaAI. V1”, September 19, 2020, Distributed by Harvard Dataverse. Available: https://dataverse.harvard.edu/dataset.xhtml?persistentId=doi:10.7910/DVN/POREXF.

[13] Larxel, “Helmet Detection. V1”, June 1, 2020, Distributed by Kaggle. Available: https://www.kaggle.com/andrewmvd/helmet-detection

[14] J. Redmon. “Darknet: Open Source Neural Networks in C.” pjreddie.com. https://pjreddie.com/darknet/ (accessed May 6, 2020).

[15] N. Wojke, A. Bewley and D. Paulus, "Simple online and realtime tracking with a deep association metric," 2017 IEEE International Conference on Image Processing (ICIP), 2017, Beijing, China, September 17-20, 2017. pp. 3645-3649, doi: 10.1109/ICIP.2017.8296962.
Available: https://ieeexplore.ieee.org/document/8296962

[16] OpenCV. 2020. opencv (Version 4.5.0) [Source code]. https://github.com/opencv/opencv

[17] The AI Guy. 2020. yolov4-deepsort [Source code]. https://github.com/theAIGuysCode/yolov4-deepsort