I^2UTS: An IoT based Intelligent Urban Traffic System

Vejey Pradeep Suresh Achari, Zeba Khanam, Amit Kumar Singh,
Anish Jindal, Alok Prakash and Neeraj Kumar

***Abstract*—Growing population and migration to cities have given birth to multiple urban issues. Traffic congestion is one of the most prominent ones with severe side effects like fuel wastage, loss of lives, and slow productivity. The traditional traffic control system deploys programming logic control (PLC) which uses round-robin scheduling algorithm. However, few recent works have proposed IoT-based framework which requires the deployment of a series of sensors. In this paper, we propose an IoT-based framework that uses the existing network of CCTV cameras at the junction. An edge device is used to estimate the traffic density and detect emergency vehicles using YOLO v3 -Efficient Net. These two parameters are used as an input to a novel traffic control algorithm. The performance of the proposed framework has been evaluated by analyzing its properties using the UA-DETRAC dataset. The proposed framework achieves 68.10% vehicle detection accuracy.**

***Index Terms*—Vehicle Detection, Deep Neural Network, Traffic Control, Edge Computing**

I. Introduction

TRAFFIC congestion is one of the major urban issues which most of cities worldwide are reeling under. The growing number of vehicles and limited land resources have broken the backbone of the traffic infrastructure. The other issues which stem from traffic congestion are diverse but not limited to environmental damage due to extra fuel consumption, productivity slow down and loss of lives due to delay in emergency vehicle. One of the most common technique adopted by most of the countries to address this pressing urban issue is by upgrading the infrastructure with construction of new roads and addition of new lanes to the existing roads. However, the land resource constraint in most of the urban cities especially in developing countries has proved to be a major bottleneck in improving the road infrastructure.

This has drawn attention of researchers to look for the solution of traffic congestion using Internet of Things (IoT) frameworks for traffic management [1]. The traditional traffic control system is based on programming logic controllers (PLC) that set a fixed duration for each lane to have a green signal, i.e., allow vehicles to pass from a lane [2]. A round robin based scheduling algorithm is used to allow the traffic to pass through each lane. This has proven to be extremely inefficient and major cause of traffic congestion.

To overcome aforementioned problems, recent works have designed intelligent traffic systems using IoT frameworks [3], [4]. Most of works have focused on using recent technologies like infrared cameras and GPS [5], RFID [6], Bluetooth and Zigbee [7]. However, these technologies require deployment of array of sensors at the traffic signal and emergency vehicle. Apart from the requirement of large investment to revamping the entire traffic infrastructure, the accuracy of detection of emergency vehicle and traffic density estimation are not substantially high due to interference from environmental noise [8].

Few handful of works have leveraged existing CCTV camera network for designing the IoT based framework for traffic systems. This has proven to be more economical and efficient. A series of works [9], [10] have proposed an IoT framework for urban traffic management which uses CCTV videos to determine traffic speed and density. CCTV footage has also been used for predicting accidents [11], monitoring spatio-temporal behaviour of pedestrians [12], and detection of firearms and knives [13].

Vejey Pradeep Suresh Achari and Neeraj Kumar are with the Department of Computer Science Engineering, Thapar Institute of Engineering and Technology, Deemed to be University, Patiala, Punjab, India e-mail: vejey.pradeep@gmail.com neeraj.kumar@thapar.edu

Zeba Khanam, Amit Kumar Singh and Anish Jindal are with the School of Computer Science and Electronics Engineering, University of Essex, U.K e-mail: zeba.khanam,a.k.singh,a.jindal@essex.ac.uk

Alok Prakash is with the School of Computer Science and Engineering, Nanyang Technological University, Singapore e-mail: alok@ntu.edu.sg

Recent advances in computer vision has revolutionized the field of video analytic allowing end-to-end learning on large datasets using Deep Neural Networks (DDNs). Fan et. al. [14] had explored use of Faster R-CNN neural network architecture [15] to track vehicles in CCTV footage. Peppa et al. [16] evaluated the performances of multiple DNNs like two versions of Faster R-CNN trained on two famous datasets COCO [17] and KITTI [18], the Single Shot MultiBox Detector MobileNet neural network on CCTV footage with different weather conditions. Acharya et al. [19] used combination of Faster R-CNN neural network architecture and kalman filter for pedestrian detection and tracking. Roy et. al [20] used a combination of YOLO v3 [21] and VGG-16 network to detect emergency vehicle in heavy traffic. The aforementioned works prove that use of DNNs to estimate traffic density and detect emergency vehicle using CCTV footage is a promising approach. However, most of these works in literature performs offline analysis of recorded CCTV footage on the cloud server without considering computational resource constraints. Another major issues which the researcher face when using CCTV footage on the cloud server is the privacy concern.

As suggested by many works [22], [23], edge computing paradigm offers a solution to address the privacy and computational resource constraint issues. Edge computing has been explored in a pilot project of building urban traffic system for Australian city of Liverpool (NSW) [9]. The project aims to be detect the pedestrians and vehicles using YOLO v3 [21] and Kalman filter deployed on Nvidia Jetson TX2. However, the project limits itself to the tracking of vehicles and pedestrians and does not use this information for traffic congestion. Another work [24] has used embedded GPU Jetson Nano NVIDIA to run multiple DNNs to detect vehicles and pedestrians in complex rural scenarios. Chauhan et. al. [10] have trained YOLO network on GPU and then transfer the weights of DNNs to the embedded platforms co-located with the CCTV cameras. However, the count of the vehicles are sent to the cloud for further analysis and not controlling the traffic.

To the best of our knowledge, there exists no work in literature which has onsite traffic density and emergency vehicles to control the traffic signal scheduling in a privacy preserving resource-constrained device. In this work, we estimate the traffic density by detecting vehicles using YOLO v3 trained on the backbone of Efficient Net from the images captured by CCTV. The trained network weights are transferred to the edge device, i.e. Raspberry Pi connected to the CCTV camera at the traffic junction to detect vehicles in the road lanes and emergency vehicles. Our proposed novel traffic control algorithm uses traffic density and emergency vehicles at junctions as an input.

The remainder of the paper is organised as follows. Section II defines the proposed IoT framework. The experimental evaluation and results of the proposed system is shown in Section III. Finally, the conclusion and future work are presented in Section IV.

II. Proposed IoT framework

The proposed IoT based urban traffic management system is described next. The previous works have used NVIDIA Jetson TX2 which is a powerful edge device available with the dual support of CPU and GPU. However, the cost of the device is 24 times more than average cost of the edge devices. Considering the economic feasibility of the system, we deploy Raspberry PI 4 as the edge platform with moderate memory of 4 GB and powerful Quad core cortex-A72 (Arm-8) 1.5GHz processor. A higher level overview of the system is as follows.

1) Train the YOLO v3 network with Efficient net as a backbone on cloud to detect the vehicles
2) Transfer the trained weights to the YOLO v3-Efficient network deployed on the edge device Raspberry Pi for testing.
3) Estimate the traffic density of each road lane and emergency vehicles.
4) Use the parameters generated in previous steps as an input to the proposed traffic control algorithm.

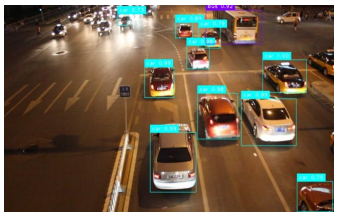
A. Dataset

Most of the previous works have either used KITTI and COCO datasets for training YOLO v3 network. However, a recent dataset UA-DETRAC [25] created by the University of Albany for detection and tracking is more profound as a real-world multi-object detection and tracking benchmark. It consists of traffic camera videos at 25 fps for 10 hours with a resolution of 960x540 pixels. The recordings are from 24 different locations within Beijing and Tianjin in China. There are around 1.21 million labelled bounding boxes of 8250 vehicles consisting of 4 classes, namely car, van, bus, others. The emergency vehicle falls in the category of the van. Some of the CCTV images are illustrated in Figure 1. Another advantage of using this dataset is that CCTV footage captures multi-class weather variance and day and night illumination variation. In this paper, the training, validation and testing data is derived by dividing the dataset into 70:15:15 ratio.

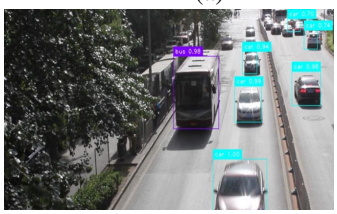
(a)

(b)

Fig. 1: Example of CCTV images from UA-DETRAC [25]



(a)



(b)

Fig. 2: Examples of vehicle detection by YOLO v3-Efficient Net object detector on CCTV images from UA-DETRAC [25]

B. DNN for Vehicle detection

For vehicle detection and classification into our four annotation classes: bus, car, vans and others, we use YOLO v3 CNN model [21] trained on the backbone of Efficient Net [26]. Unlike the traditional YOLO v3 model [21] which has used DarkNet-53 network as backbone, our network has a high vehicle detection accuracy at low inference latency.

1) YOLO v3: The single neural network, YOLO v3, is applied to CCTV images. The neural network divides the input image into the grid regions. For each grid region, bounding boxes are predicted with confidence score which is calculated by logistic regression. Each region has a probability associated with it which are used to threshold the detection using non-maximum suppression.

2) Efficient Net: Efficient Net [26] is used as the backbone of YOLO v3. The main principle of Efficient Net is strategical scaling of DNNs. In general, the performance of a DNN in terms of scaling is increased either by width, depth or resolution scaling. A novel compound scaling method is proposed which scales all the three dimensions (depth, width and resolution) based on fixed ratio. Few instances of vehicle detection are illustrated in Figure 2.

C. Traffic Control Algorithm

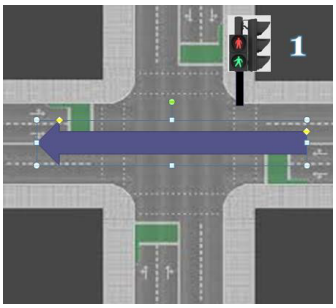
In this section, we propose a novel traffic control algorithm where the traffic density and emergency vehicle based priority is used as an input parameter. In order to elucidate our Algorithm 1, we consider a junction with two traffic lights which have red, amber and green signal respectively as shown in Figure 3. Each traffic light i has following variables associated with it:

1) TD_i : Traffic density of the road where traffic light i is installed. This is calculated as the number of vehicles detected on the corresponding road lane.
2) P_i : Priority of the traffic light.
3) $O_i = \{o_r, o_a, o_g\}$: A cluster of boolean variable indicating the state of the signal. If the red signal is on then o_r = 1, other wise o_r = 0. Similarly, o_a and o_g indicate the state of amber and green signal, respectively.

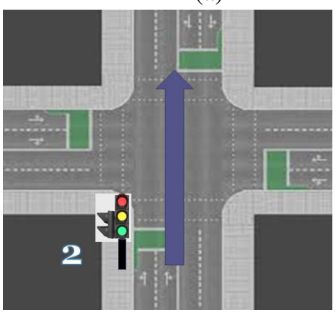
Algorithm 1 assign the signal priority according to the traffic density on the corresponding road, i.e. the signal with the highest traffic density is assigned the highest priority. For any given traffic light, green and amber signal are switched on for t_m and t_n seconds such that $t_m >> t_n$. However, if the emergency vehicle is detected the corresponding traffic light is set with highest priority to clear the lane of the emergency vehicle. This approach allows the designed traffic system to save as many lives as possible.

III. Experimentation and Results

In this section, we evaluate the performance of our proposed IoT framework. The weights of the YOLO v3-Effecient Net are trained on a cloud server with Intel i7-9th generation processor custom fitted with Nvidia 1660Ti GPU on Linux 18.04 LTS operating system. CUDA 10.1 with Cudnn 7 libraries were used to perform computation on the GPU. The edge device used for testing is Raspberry Pi 4 equipped with Quad core cortex-A72 (Arm-8) 1.5GHz processor, 4GB RAM and OpenGL ES 3.0 graphics. It host the Raspbian Buster as the operating system.



(a)



(b)

Fig. 3: Example of traffic junction with two red lights (a) Direction Flow of traffic when traffic light 1 is either green and amber (b) Direction Flow of traffic when traffic light 2 is either green or amber

1) Evaluation Metric: The principal metric used for evaluation of the proposed IoT framework is Mean Average Precision (mAP). mAP is indicative of the accuracy of object detectors. In order to compute mAP, we first need to compute precision and recall. Precision is a measure of how often the model detects correctly and recall is a measure of how good the model has found all the positives. The precision and recall are calculated using following equations.

$$Precision = \frac{TP}{TP + FP} \tag{1}$$

$$Recall = \frac{TP}{TP + FN} \tag{2}$$

where, TP, TN, FP and FN are true positive, true negative, false positive and false negative, respectively.

Algorithm 1: Traffic Control Algorithm

Input: 1. TD_i: Traffic Density of Traffic light i
2. P_i: Priority of Traffic light i
3. n: Number of traffic lights
Output: $O_i = \{o_r, o_a, o_g\}$
begin
  Initialization:
    $O_i = \{1, 0, 0\}$ ∀i = 1,...,n ▷ Set all the traffic lights red
  while *True* **do**
    Set $P_i = 0$ ∀i = 1,...,n; ▷ Set Priority of all the traffic junction 0
    Calculate traffic density TD_i for each traffic light i;
    if *emergency vehicle detected at junction* i **then**
      $P_i = 1$;
      Sort $T = \{j \mid j = 1, \ldots, n \;\&\&\; j \neq i\}$ in descending order according traffic density TD;
      Set Priority $P_{T_k} = 1 + k$ ∀k = 1,...,n-1;
    else
      Sort $T = \{j \mid j = 1, \ldots, n\}$ in descending order according traffic density TD;
      Set Priority $P_{T_k} = k$ ∀k = 1,...,n;
    i = 1;
    for $i \neq n$ **do**
      Set $O_{P_k == i} = \{0, 0, 1\}$;
      Wait for a time period t_m;
      Set $O_{P_k == i} = \{0, 1, 0\}$;
      Wait for a time period t_n;
      Set $O_{P_k == i} == \{1, 0, 0\}$;

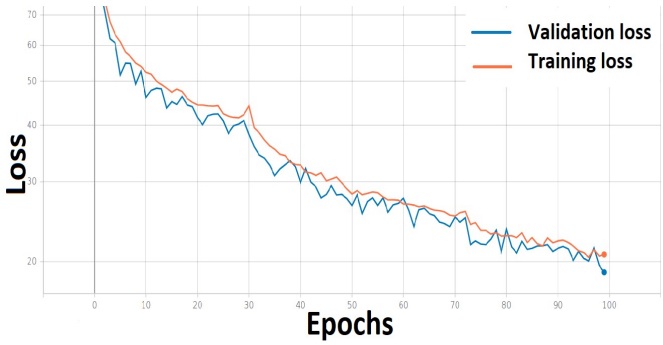


Fig. 4: Number of Epoch with respect to Loss

A. Results

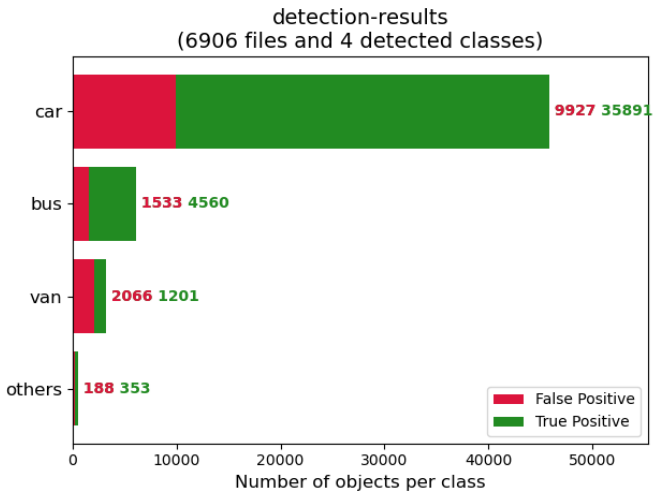
While training the DNN for vehicle detection and fine-tuning the hyper-parameters one must ensure that the model neither overfits nor underfits. A model overfits when the validation loss is greater than training loss. However, if reverse happens, the model's validation loss is lower than training loss, model underfits. To ensure both the phenomenons do not occur, we varied our number of epoch with respect to validation loss. As shown in Figure 4, the training and validation loss value's difference is lowest around 100^{th} epoch.

Further, trained kernel weights are then transferred to the inference model on the edge device, Raspberry Pi 4. It was observed the inference time of the detector on the Raspberry Pi varied between 1.45 – 1.57 sec per frame.

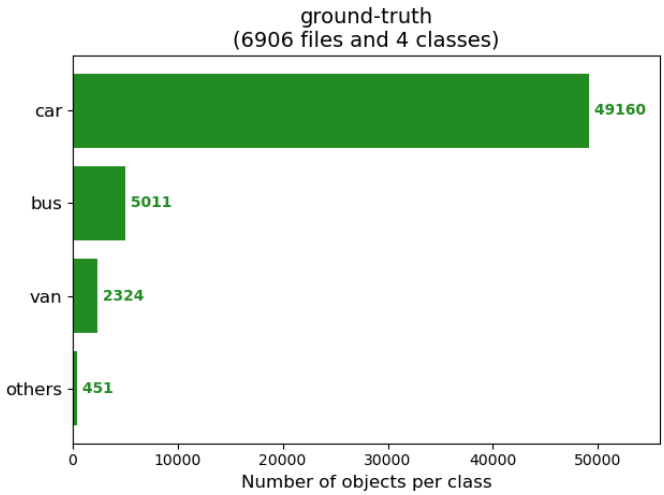
TABLE I: Power consumption of IoT device on different loads

Parameters	Current (Amps)	Voltage (Volts)	Power (Watts)
IoT device not connected to monitor	0.61	5.08	3.1
IoT device connected to monitor	0.68	5.08	3.45
IoT device running detector and monitor	1.4	5.13	7.182
IoT device running only detector	1.34	5.13	6.87

The power consumption of the IoT device (Raspberry Pi 4) per second on different loads is tabulated in Table I. The input to the IoT device was DC 5.1V and 3A. The highest power consumption, 7.18 W, is achieved when running our detector along side the monitor. However, it is worth noting that it is only half of the input power supplied. When running the detector connected to a traffic camera, the power consumption is 6.87 W. The power is measured using USB power connector. Figure 5 illustrates the detection of FP and TP for each class on the test set. These parameters are used as an input to calculate our principal evaluation metric, mAP, as evident from Equations 1 and 2. Figure 6 illustrates the mAP of each class. Average precision is obtained by calculating the area under a precision-recall curve. Precision for each class, i.e. bus, car, van and others are 0.8816, 0.7083, 0.3749 and 0.7592, respectively. An overall mAP indicates that the detector can identify objects and classify them with a success rate of 68.10%. The reason for high false positive for the car and van is because of the lack of annotated data in ground truth and occlusion scenarios.



(a)



(b)

Fig. 5: Bar chart showing (a) false positive and true positives (b) ground truth for each class

IV. Conclusion and Future Work

This paper proposes a IoT framework for urban traffic management system using the CCTV camera. Unlike, the existing frameworks which require deployment of an array of sensors, this economical framework uses the existing CCTV camera network at traffic junction. This urban traffic management system aims to overcome the problem of traffic congestion by deploying YOLO v3-Efficient DNN to detect the traffic density and emergency vehicles. Unlike, the existing traffic light scheduling algorithm which deploy round robin scheduling, our novel traffic scheduling algorithm uses both these parameters to control the traffic lights at the junctions. Raspberry Pi 4 is used as an edge computing device to test the accuracy of detector. The inference YOLO v3-Efficient Net Model achieves 68.10% accuracy which can further be enhanced by training the network on multiple datasets with CCTV footage from different countries and scenarios. The inference model can further be enhanced by model pruning and using weights which have been learned on datasets which are more inclusive and descriptive in nature.

At present, UA-DETRAC dataset is used for training and testing the DNN model for vehicle detection. In future, we plan to evaluate the properties of our IoT framework by testing in real-time to measure the reduction in the average wait time of vehicle at the junction.

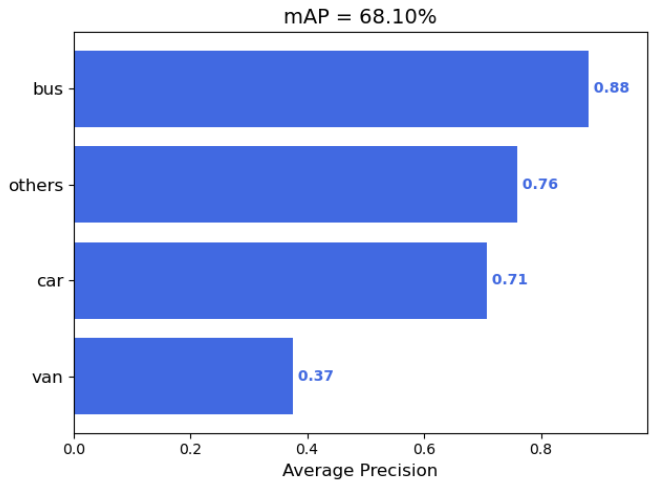


Fig. 6: Bar chart showing mAP for each class

ACKNOWLEDGMENT

The authors acknowledge the support of GCRC G011 project sponsored under the GCRF@Essex Engagement fund to carry out this research work.

REFERENCES

[1] K. Nellore and G. P. Hancke, "A survey on urban traffic management system using wireless sensor networks," *Sensors*, vol. 16, no. 2, p. 157, 2016.

[2] V. Paruchuri, S. Chellappan, and R. B. Lenin, "Arrival time based traffic signal optimization for intelligent transportation systems," in *2013 IEEE 27th International Conference on Advanced Information Networking and Applications (AINA)*. IEEE, 2013, pp. 703–709.

[3] M. Z. Talukder, S. S. Towqir, A. R. Remon, and H. U. Zaman, "An iot based automated traffic control system with real-time update capability," in *2017 8th International Conference on Computing, Communication and Networking Technologies (ICCCNT)*. IEEE, 2017, pp. 1–6.

[4] M. Saifuzzaman, N. N. Moon, and F. N. Nur, "Iot based street lighting and traffic management system," in *2017 IEEE region 10 humanitarian technology conference (R10-HTC)*. IEEE, 2017, pp. 121–124.

[5] S. Deshmukh and S. Vanjale, "Iot based traffic signal control for reducing time delay of an emergency vehicle using gps," in *2018 Fourth International Conference on Computing Communication Control and Automation (ICCUBEA)*. IEEE, 2018, pp. 1–3.

[6] Y.-L. Lai, Y.-H. Chou, and L.-C. Chang, "An intelligent iot emergency vehicle warning system using rfid and wi-fi technologies for emergency medical services," *Technology and health care*, vol. 26, no. 1, pp. 43–55, 2018.

[7] R. Sundar, S. Hebbar, and V. Golla, "Implementing intelligent traffic control system for congestion control, ambulance clearance, and stolen vehicle detection," *IEEE Sensors Journal*, vol. 15, no. 2, pp. 1109–1113, 2014.

[8] D. D. Romero, A. S. Prabuwono, A. Hasniaty *et al.*, "A review of sensing techniques for real-time traffic surveillance," *Journal of applied sciences*, vol. 11, no. 1, pp. 192–198, 2011.

[9] J. Barthélemy, N. Verstaevel, H. Forehead, and P. Perez, "Edge-computing video analytics for real-time traffic monitoring in a smart city," *Sensors*, vol. 19, no. 9, p. 2048, 2019.

[10] M. S. Chauhan, A. Singh, M. Khemka, A. Prateek, and R. Sen, "Embedded cnn based vehicle classification and counting in non-laned road traffic," in *Proceedings of the Tenth International Conference on Information and Communication Technologies and Development*, 2019, pp. 1–11.

[11] A. P. Shah, J.-B. Lamare, T. Nguyen-Anh, and A. Hauptmann, "Cadp: A novel dataset for cctv traffic camera based accident analysis," in *2018 15th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*. IEEE, 2018, pp. 1–9.

[12] Y. Cai, L. Dai, H. Wang, L. Chen, Y. Li, M. A. Sotelo, and Z. Li, "Pedestrian motion trajectory prediction in intelligent driving from far shot first-person perspective video," *IEEE Transactions on Intelligent Transportation Systems*, 2021.

[13] M. Grega, A. Matiolański, P. Guzik, and M. Leszczuk, "Automated detection of firearms and knives in a cctv image," *Sensors*, vol. 16, no. 1, p. 47, 2016.

[14] Q. Fan, L. Brown, and J. Smith, "A closer look at faster r-cnn for vehicle detection," in *2016 IEEE intelligent vehicles symposium (IV)*. IEEE, 2016, pp. 124–129.

[15] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: towards real-time object detection with region proposal networks," *IEEE transactions on pattern analysis and machine intelligence*, vol. 39, no. 6, pp. 1137–1149, 2016.

[16] M. Peppa, D. Bell, T. Komar, and W. Xiao, "Urban traffic flow analysis based on deep learning car detection from cctv image series." *International Archives of the Photogrammetry, Remote Sensing & Spatial Information Sciences*, vol. 42, no. 4, 2018.

[17] D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part I*. Springer, 2014, vol. 8689.

[18] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *2012 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2012, pp. 3354–3361.

[19] D. Acharya, K. Khoshelham, and S. Winter, "Real-time detection and tracking of pedestrians in cctv images using a deep convolutional neural network," in *Proceedings of the 4th annual conference of research@ locate*, vol. 1913, 2017, pp. 31–36.

[20] S. Roy and M. S. Rahman, "Emergency vehicle detection on heavy traffic road from cctv footage using deep convolutional neural network," in *2019 International Conference on Electrical, Computer and Communication Engineering (ECCE)*. IEEE, 2019, pp. 1–6.

[21] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," *arXiv preprint arXiv:1804.02767*, 2018.

[22] M. Satyanarayanan, P. Simoens, Y. Xiao, P. Pillai, Z. Chen, K. Ha, W. Hu, and B. Amos, "Edge analytics in the internet of things," *IEEE Pervasive Computing*, vol. 14, no. 2, pp. 24–31, 2015.

[23] M. Satyanarayanan, "The emergence of edge computing," *Computer*, vol. 50, no. 1, pp. 30–39, 2017.

[24] L. Barba-Guaman, J. Eugenio Naranjo, and A. Ortiz, "Deep learning framework for vehicle and pedestrian detection in rural roads on an embedded gpu," *Electronics*, vol. 9, no. 4, p. 589, 2020.

[25] L. Wen, D. Du, Z. Cai, Z. Lei, M.-C. Chang, H. Qi, J. Lim, M.-H. Yang, and S. Lyu, "Ua-detrac: A new benchmark and protocol for multi-object detection and tracking," *Computer Vision and Image Understanding*, vol. 193, p. 102907, 2020.

[26] M. Tan and Q. Le, "Efficientnet: Rethinking model scaling for convolutional neural networks," in *International Conference on Machine Learning*. PMLR, 2019, pp. 6105–6114.