Real-Time Vehicle Type Recognition Using Deep Learning Techniques

AMRNVB Pethiyagoda[1#], TL Weerawardane[2], MWP Maduranga[1] and DMR Kulasekara[1]

[1]*Department of Computer Engineering, Faculty of Computing, General Sir John Kotelawala Defence University, Ratmalana, Sri Lanka*

[2]*Department of Electrical, Electronic & Telecommunication Engineering, Faculty of Engineering, General Sir John Kotelawala Defence University, Ratmalana, Sri Lanka*

[#]nadinpethiyagoda4@gmail.com

Abstract*: Modern intelligent transportation systems heavily rely on vehicle type classification technology. Deep learning-based vehicle type classification technology has sparked growing concern as Image Processing, Pattern recognition, and Deep Learning have all advanced. Convolutional neural work, particularly You Only Look Once (YOLO), has demonstrated significant benefits in image classification and object detection during the past few years. Due to its ability to forecast objects in real-time, this algorithm increases detection speed. High accuracy: The YOLO prediction method yields precise results with few background mistakes. Additionally, YOLO is aware of generalized object representation. This method, which ranks among the best for object detection, performs significantly better than R-CNN techniques. In this paper, YOLOv5 is used to demonstrate vehicle type detection; YOLOv5 m model was chosen since it suits mobile deployments, The model was trained with a dataset of 9200 images, where 2300 images were allocated for each class with a variety of vehicles. Experimental results for 100 epochs with a batch size of 16 show mAP@.5 at 78.1% and mAP@.5:.95 at 71.7% trained and tested on four vehicle classes.*

Keywords: *You Only Look Once (YOLO), Deep Learning, Convolutional Neural Networks (CNN), Single Shot Detector (SSD) Vehicle Recognition*

1. Introduction

Various advancements in the field of machine vision have fundamentally transformed the world. Technology has had an impact on various industries, including transportation. Because of population increase and human requirements, the use of vehicles has risen dramatically. As a result of the increased difficulties in controlling these vehicles, Intelligent Traffic Systems were developed, Vehicle Type Detection systems are critical components of intelligent traffic systems, and they have a wide range of applications [1], including highway toll collection, traffic flow statistics, and urban traffic monitoring. The development of autonomous driving technology has given people a new knowledge of high-level computer vision, and intelligent transportation and driverless driving technologies have drawn an increasing amount of interest. Vehicle Type Detection is a relatively significant technology in intelligent transportation and autonomous driving. There are already numerous methods for categorizing different types of vehicles thanks to the quick growth of large-scale data, computer hardware, and deep learning technologies. CNN, Faster RCNN, YOLO [2,18], and SSD have mostly used approaches that can be applied. This work uses the most recent real-time object detection technique of YOLO to address the drawbacks of existing object detection systems.

YOLO is a state-of-the-art, real-time object detection system introduced in 2015 by [13]. YOLO proposes using an end-to-end neural network that provides predictions of bounding boxes and class probabilities all at once as opposed to the strategy used by object detection algorithms before it, which repurpose classifiers to do detection. R-CNNs are a type of two-stage detector and one of the early deep learning-based object detectors. The major issue with the R-CNN family of networks is their speed. However, though they frequently produce very accurate results, they were incredibly slow, averaging barely 5 FPS on a GPU. YOLO employs a one-stage detector technique to aid in accelerating deep learning-based object detectors. YOLO has the natural advantage of speed, better Intersection over Union in bounding boxes, and improved prediction accuracy compared to real-time object detectors. YOLO runs at up to 45 FPS, making it a far faster algorithm than its competitors. The GoogleNet architecture inspired YOLO's architecture, YOLO's architecture has a total of 24 convolutional layers with 2 fully connected layers at the end. The main problems with YOLO, the identification of small objects in groups and the localization accuracy—were supposed to be addressed by YOLOv2 [14]. By implementing batch normalization, YOLOv2 raises the network's mean Average Precision. The addition of anchor boxes, as suggested by YOLOv2, was a considerably more significant improvement to the YOLO algorithm. As is well known, YOLO predicts one object for every grid cell. Although this simplifies the constructed model, it causes

problems when a single cell contains several objects because YOLO can only assign one class to the cell.

By enabling the prediction of numerous bounding boxes from a single cell, YOLOv2 eliminates this restriction. The network is instructed to anticipate five bounding boxes for each cell to do this. YOLO9000 [14] was presented as a technique to discover more classes than COCO as an object detection dataset could have made possible, using a similar network design to YOLOv2. Although YOLO9000 has a lower mean Average Precision than YOLOv2, it is still a powerful algorithm because it can identify over 9000 classes. YOLOv3 [15] was proposed to enhance YOLO with modern CNNs that utilize residual networks and skip connections. YOLOv2 employs the DarkNet-19 as the model architecture, but YOLOv3 uses the significantly more intricate DarkNet-53, a 106-layer neural network with residual blocks and up sampling networks, as the model backbone. With the feature maps being extracted at layers 82, 94, and 106 for these predictions, YOLOv3's architectural innovation allows it to forecast at three different sizes.

YOLOv4 [16] is built using CSPDarknet53 as the backbone, SPP (Spatial pyramid pooling), and PAN (Path Aggregation Network) for what is known as "the Neck," and YOLOv3 for "the Head" following recent research findings.

This system uses the latest algorithm, YOLOv5, which uses the PyTorch [20] framework possessing many advantages such as smaller size, higher performance, and better integration than YOLOv4.

2. Related Works

In the works of [2][6], the authors have presented experimental results that that YOLOv4 had better performance, F1score, precision, recall, and mAP values compared to other models in [2] and YOLOv3 has demonstrated better results in performance and accuracy than R-CNN and Fast R-CNN [6]. Yanhong Yang [3] uses the SSD algorithm to achieve vehicle classification and positioning, from the picture collection, picture calibration, model training, and model detection, several aspects of the detailed introduction of the vehicle classification process. PASCAL VOC dataset was used, and TensorFlow framework and SSD model with VGG16 model were used for model training. In [2-9] [11][12] Common vehicle categories are bus, car, truck, bus, and motorbikes. In [6-7] limitations were how to effectively detect vehicles in complex environments. Due to the limitations of hardware and time, in-depth research can be conducted in the future on the aspects of improving accuracy, improving detection accuracy, and improving calibration methods. A combination of YOLOv4 and DeepSORT has been used in [7] for vehicle detection and real-time object tracking, respectively.

In [8] proposes a CNN model for vehicle classification with low-resolution images from a frontal perspective. The model was trained as a multinomial logistic regression where the cross-entropy of the ground truth labels, and the model's prediction estimates the error. Data augmentation was performed to prevent overfitting. A leaky rectifier activation function (LReLU) instead of (ReLU) was set up for the convolution output. However, [10] proposed a CNN architecture for vehicle type classification. The system requires only one input, a vehicle image. The model consists of two convolution layers, 1st, and 2nd layer. Two pooling layers, four activation functions (ReLU) The 3rd, 4th, and 5th layers are fully connected. In [12] proposed the network developed has a total of 13 layers, 1 convolutional input layer, 11 intermediate layers including a combination of Rectified Linear Unit (ReLU) activation, convolutional, dropout, max pooling, flatten, and densely connected layers, and 1 SoftMax output layer. In the works [6][11] the gathered datasets from public sources such as COCO, OpenImage, PASCAL VOC, and some works their traffic data collected from camera sources. Dataset split was 80:20 80% for training and 20% for testing [6][9].

In the works [17][12] The test data gave it had produced better accuracies with pictures with high definition while for the pictures with low definition, the recognition accuracy decreases. It is also observed that the probability of identifying small cars as medium-sized vehicles is only 8.69%, and the probability of identifying large cars is lower, 2.14% only in [17] and. Further improvements in prediction accuracy include training on more quality images to allow it to extract more features from the data and further dividing into more classes [12]. In [5][10] the authors wish to aim for better accuracies and stability by searching for suitable hyperparameters. Research gaps in [5-7] show the need to cover more variations of vehicles, Cars Image datasets need more data to classify, train, and real-time data analysis of the traffic and also more complex environmental conditions such as night-time and heavy rain. In [8-9][11] images that could be produced were replicated using data augmentation to improve precision and [4] stated image processing techniques were used to improve prediction accuracy. In [11] the authors have used Faster R-CNN, for shareable convolutional layers of RPN and detection network, the improved ZF net is applied on the PASCAL VOC2012 as the backbone network.

In [12] the authors have developed a CNN, to detect types of vehicles commonly found on the road for database collection purposes and improve the existing vehicle recognition for advanced applications. [10] To avoid overfitting Dropout method has been used, and the final layer is the predictor. TensorFlow was used to implement the CNN structures. The hyperparameters for the CNN model were also mentioned which can affect the performance of the CNN. The dataset mentioned was obtained from extracted frames of a video source. In works [11] RPN is trained by using Stochastic Gradient Descent (SGD). The method has better detection average precision for cars and trucks, while the average precision of minivans

and buses is lower. The result might be caused by a little training set of minivans and buses.

Some knowledge gaps were identified in the literature review conducted; many authors have included foreign datasets, resulting in fewer accuracies when tested over real-time data. Some researchers are trying to improve the performance and recognition accuracies by considering images of different lighting, weather conditions, noise reduction, etc. which had been a challenge. According to the works in [2], it is clear that YOLO had outperformed other models such as Faster-RCNN and SSD.

3. Design Framework and Methodology

This section follows the methodology applied to collecting and building the dataset needed for model train/test, exploring, and understanding the architecture behind the YOLOv5, and choosing the best approach for the desired output.

A. Dataset

In this experiment, the gathered dataset was from public sources such as Kaggle, Stanford Cars Dataset, vehicle images scraped from local automotive e-commerce websites in Sri Lanka, and traffic data collected from a video source with the following characteristics: Video duration: 300 seconds, resolution: 1080x2340 pixels, frame rate: 30 FPS. The dataset was prepared for six major types of vehicles, such as cars, buses, vans, trucks, motorbikes, and three wheels. These images are of different illumination, angle, and different vehicle models. The total dataset size is 9,200 images at a resolution of resized to 160x120 pixels, with 2,300 images allocated for each class. The gathered dataset was annotated in YOLO format using a free open source called Labelling to graphically label the images. For each image file in the same directory, a text file with the same name is created in the YOLO labelling format. Each text file provides the object class, object coordinates, height, and width for the accompanying image file. The dataset was split into train and test, which are 80% (7,360 images) and 20% (1,840 images) respectively.

Figure 1. Sample images from the training dataset

B. YOLOv5

The most cutting-edge object detection algorithm currently in use is the YOLOv5 [19], which Ultralytics introduced in June 2020. It is a novel convolutional neural network (CNN) that accurately detects objects in real-time. This method processes the entire image using a single neural network, then divides it into parts and forecasts bounding boxes and probabilities for each component. The predicted probability is used to weight these bounding boxes. In the sense that it only performs one forward propagation cycle through the neural network, the approach "only looks once" at the image before making predictions. After non-max suppression, it then provides discovered items. YOLOv5 consists of:

- *Backbone: New CSP-Darknet53*
- *Neck: SPPF, New CSP-PAN*
- *Head: YOLOv3 Head*

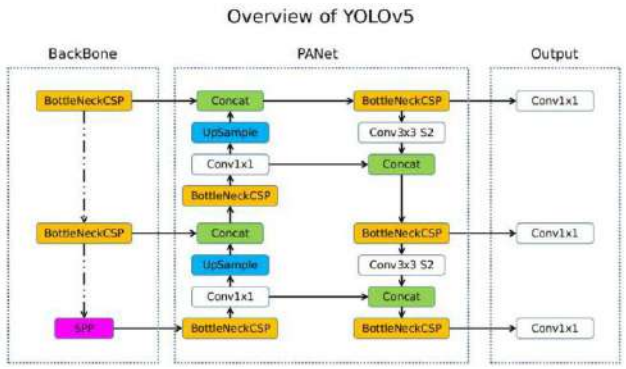

Figure 2. YOLOv5 architecture

The overview of the YOLOv5 architecture is shown in Figure 2. To understand the classes of objects in the data, YOLOv5 models need to be trained using labelled data. The custom dataset that was prepared was of the YOLO format with one text file per image. The text file specifications are:

- One row per object
- Each row is *class x_center y_center width height* format.
- Box coordinates must be in normalized xywh format (from 0 - 1). If your boxes are in pixels, divide *x_center* and *width* by image width, and *y_center* and *height* by image height.
- Class numbers are zero-indexed (start from 0).

YOLOv5 provides pre-trained models:

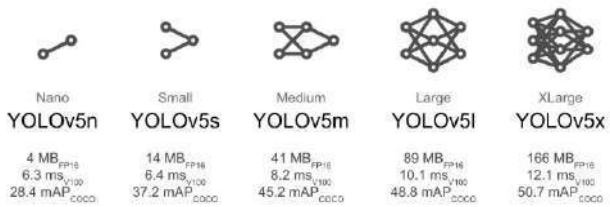

Figure 3. YOLOv5 models

Larger models, such as YOLOv5x and YOLOv5x6, will almost always yield better results, but they contain more parameters, need more CUDA memory to train, and run more slowly.

The YOLOv5 loss consists of three parts:

- Classes loss (BCE loss)
- objectless loss (BCE loss)
- Location loss (CIoU loss)

$$L = \lambda_1 L_{cls} + \lambda_2 L_{obj} + \lambda_3 L_{box} \quad (1)$$

The objectness losses of the three prediction layers (P3, P4, P5) are weighted differently. The balance weights are [4.0, 1.0, 0.4] respectively.

$$L_{obj} = 4.0 \cdot L^{obj}_{P_3} + 1.0 \cdot L^{obj}_{P_4} + 0.4 \cdot L^{obj}_{P_5} \quad (2)$$

L^{obj}

YOLOv5 uses the following formula to calculate the predicted target information:

$$b_x = (2 \cdot \sigma(t_x) - 0.5) + C_x \quad (3)$$

$$b_y = (2 \cdot \sigma(t_y) - 0.5) + C_y \quad (4)$$

$$b_w = p_w \cdot (2 \cdot \sigma(t_w))^2 \quad (5)$$

$$b_h = p_h \cdot (2 \cdot \sigma(t_h))^2 \quad (6)$$

The build targets to match positive samples: Calculate the aspect ratio of GT and Anchor Templates

$$r_w = w_g / w_{at} \quad (7)$$

$$r_h = h_g / h_{at} \quad (8)$$

$$j^{wh} = \max(r, \frac{1}{r_w}) \quad (9)$$

$$j^{wh} = \max(r, \frac{1}{r_h}) \quad (10)$$

$$j^{wh} = \max(r, h^{wh}) \quad (11)$$

(j^{wh}

$$j^{wh} = wh_t \quad (12)$$

The final metric, the mAP across test data, is generated by averaging all mAP values for each class in order to z

4. Model Training and Results

The model was trained on a system equipped with Ubuntu 20.04.4 LTS, CUDA 10.2, 32 GB RAM, NVIDIA GeForce RTX 3090, Python 3.8, PyTorch 1.8.0. The yolov5m model was used for the training and test purpose, and a YAML file was defined to configure the paths for the dataset and the number of classes to train (Bus, Car, Motorbike, Van, Truck, Three-wheel).

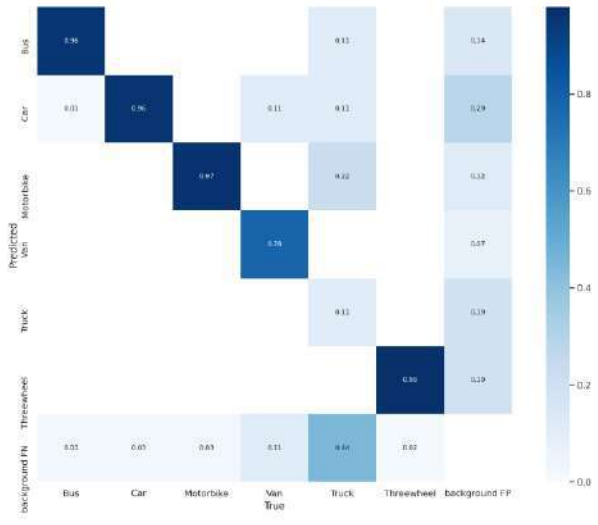

Figure 4. Confusion Matrix

However, it was trained only for 4 classes (Bus, Car, Motorbike, Three-wheel) despite having a few images of trucks and vans included. The model was trained for 100 epochs with a batch size of 16, to visualize and track data in real-time wandb (Weights and Bias) Platform was used.

during training, which allows for determining the epoch where the model starts to overfit. Figure 4 shows the confusion matrix, the only images that were incorrectly classified when the trained model was tested using the validation set were those in which a truck was misinterpreted for a bus and vice versa. This is because, when viewed from the front, a bus, and a truck both have characteristics in common.

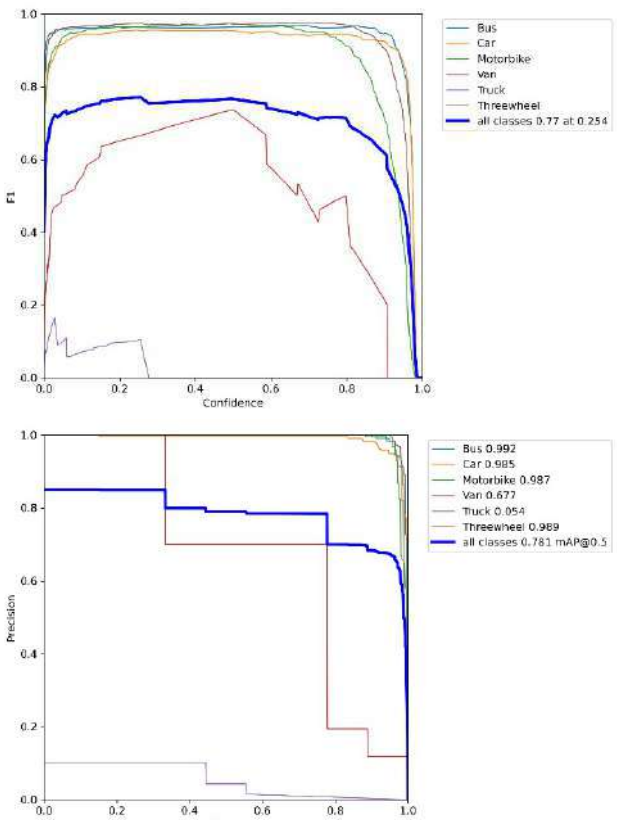

Figure 5. F1 Curve

The weighted harmonic mean of a classifier's precision (P) and recall (R), considering =1, is known as the F-measure (F1 score). According to the greatest F1 value in Figure 5, the confidence value that maximizes the precision and recall is 0.254. (0.77). Figure 6 presents the precision of each class obtained after the completion of training, it is the proportion of positive identifications which are actually correct, and Figure 7 presents the recall of each class, it is the proportion of actual positives which are identified correctly, the classes Bus, Car, Motorbike and Threewheel displayed satisfactory results although the classes Van and Truck had low results due to lack of images in the training dataset. Figure 8 shows the precision/recall curve generated from the validation set after training completes.

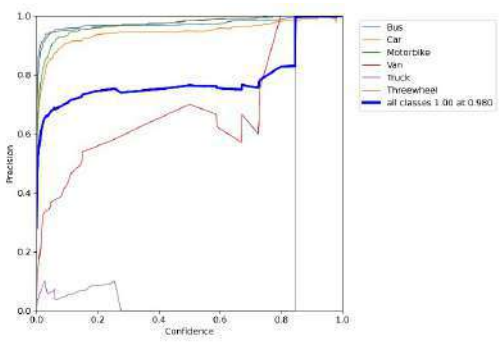

Figure 6. Precision Curve

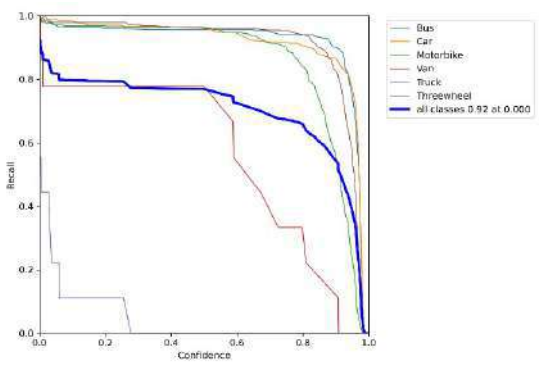

Figure 7. Recall Curve

Figure 8. Precision/Recall Curve

Class			mAP	
	Precision	Recall	mAP@.5	mAP@.5:.95
Bus	96.8	95.9	99.2	96.9
Car	94.2	96.8	98.5	93.2
Motorbike	96.5	96.4	98.7	90.0
Thre e-	96.7	98.0	98.9	95.0
All	75.4	79.3	78.1	71.7

Table 1. Model results of each class

The mAP@.5 was 78.1% and mAP@.5:.95 was 71.7%. The training results under several epochs 30, 60, and 90 are provided as follows. At 30 epochs mAP@.5 was observed of 71.43%, mAP@.5:.95 of 62.16%, At 60 epochs mAP@.5 of 76.7%, mAP@.5:.95 of 68.62% and at 90 mAP@.5 of 77.96% , mAP@.5:.95 at 71.2%. The best results were found at epoch 93 with a mAP@.5 at 78.066% and mAP@.5:.95 at 71.726%. The following experiment has shown satisfactory results, considering the knowledge gaps identified in the existing research.

5. Conclusion

This paper proposes a model for categorizing vehicle types utilizing the most recent state-of-the-art object detection model, YOLOv5. With an overall mAP@.5 of 78.1 %, the model's promising results. Although the dataset contains 9,200 images with 2,300 each, Bus, Car, Motorbike, and Three-wheel classes were actually trained, Van and Truck classes had very few data since some images had Vans and Trucks captured in the above 4 classes that were trained with, therefore they couldn't be omitted out. However, this result of mean average precision obtained relatively low compared to the higher precisions obtained for classes Bus, Car, Motorbike, and Three wheel due to the classes Van and Truck. However, this can be avoided in the future by annotating those classes which will yield better results. The

model will, nevertheless, be very successful at recognizing the type of car on the road, as shown by the results. Any future transportation system that must accurately identify the type of vehicle can easily incorporate it. While the classes can be further broken down into a more detailed manner, dividing classes of vehicles as SUVs, sedans, crossovers, jeeps, etc., Following the experiment, it was clear, that the model produced better results with respect to the precision and recall, and it is determined that the model could achieve best results, greater quality photos must be utilized to train the model to allow it to extract more features from the data. Hyperparameter tuning, a better dataset with high-resolution images, increasing the scope of Van and Truck training datasets to enhance precision.

References

[1] Chen, Y., Zhu, W., Yao, D., and Zhang, L. 2017. Vehicle type classification based on convolutional neural network. In 2017 Chinese Automation Congress (CAC) (pp. 1898-1901).

[2] Kim, J.a., Sung, J.Y., and Park, S.h. 2020. Comparison of Faster-RCNN, YOLO, and SSD for Real-Time Vehicle Type Recognition. In 2020 IEEE International Conference on Consumer Electronics - Asia (ICCE-Asia) (pp. 1-4).

[3] Yang, Y. 2020. Realization of Vehicle Classification System Based on Deep Learning. In 2020 IEEE International Conference on Power, Intelligent Computing and Systems (ICPICS) (pp. 308-311).

[4] Aishwarya, C., Mukherjee, R., and Mahato, D. 2018. Multilayer vehicle classification integrated with single frame optimized object detection framework using CNN based deep learning architecture. In 2018 IEEE International Conference on Electronics, Computing and Communication Technologies (CONECCT) (pp. 1-6).

[5] Htet, K., and Sein, M. 2020. Event Analysis for Vehicle Classification using Fast RCNN. In 2020 IEEE 9th Global Conference on Consumer Electronics (GCCE) (pp. 403-404).

[6] Shekade, A., Mahale, R., Shetage, R., Singh, A., and Gadakh, P. 2020. Vehicle Classification in Traffic Surveillance System using YOLOv3 Model. In 2020 International Conference on Electronics and Sustainable Communication Systems (ICESC) (pp. 1015-1019).

[7] Doan, T.N., and Truong, M.T. 2020. Real-time vehicle detection and counting based on YOLO and DeepSORT. In 2020 12th International Conference on Knowledge and Systems Engineering (KSE) (pp. 67-72).

[8] Roecker, M., Costa, Y., Almeida, J., and Matsushita, G. 2018. Automatic Vehicle type Classification with Convolutional Neural Networks. In 2018 25th International Conference on Systems, Signals and Image Processing (IWSSIP) (pp. 1-5).

[9] Harianto, R., Pranoto, Y., and Gunawan, T. 2021. Data Augmentation and Faster RCNN Improve Vehicle Detection and Recognition. In 2021 3rd East Indonesia Conference on Computer and Information Technology (EIConCIT) (pp. 128-133).

[10] Maungmai, W., and Nuthong, C. 2019. Vehicle Classification with Deep Learning. In 2019 IEEE 4th International Conference on Computer and Communication Systems (ICCCS) (pp. 294-298).

[11] Xinchen Wang, Weiwei Zhang, Xuncheng Wu, Lingyun Xiao, Yubin Qian, and Zhi Fang 2017. Real-time vehicle type classification with deep convolutional neural networks. Journal of Real-Time Image Processing, 16(1), p.5–14.

[12] SAN, W., LIM, M., and CHUAH, J. 2018. Efficient Vehicle Recognition and Classification using Convolutional Neural Network. In 2018 IEEE International Conference on Automatic Control and Intelligent Systems (I2CACIS) (pp. 117-122).

[13] Redmon, J., Divvala, S., Girshick, R., and Farhadi, A.. (2015). You Only Look Once: Unified, Real-Time Object Detection.

[14] Redmon, J., and Farhadi, A. 2017. YOLO9000: Better, Faster, Stronger. In 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (pp. 6517-6525).

[15] Joseph Redmon, and Ali Farhadi 2018. YOLOv3: An Incremental Improvement. CoRR, abs/1804.02767.

[16] Alexey Bochkovskiy, Chien-Yao Wang, and Hong-Yuan Mark Liao 2020. YOLOv4: Optimal Speed and Accuracy of Object Detection. CoRR, abs/2004.10934.

[17] Lin, M., and Zhao, X. 2019. Application Research of Neural Network in Vehicle Target Recognition and Classification. In 2019 International Conference on Intelligent Transportation, Big Data & Smart City (ICITBS) (pp. 5-8).

[18] Armin, E., Bejo, A., and Hidayat, R. 2020. Vehicle Type Classification in Surveillance Image based on Deep Learning Method. In 2020 3rd International Conference on Information and Communications Technology (ICOIACT) (pp. 400-404).

[19] Jocher, G., 2020. GitHub - ultralytics/yolov5: YOLOv5 in PyTorch > ONNX > CoreML > TFLite. [online] GitHub. Available at: <https://github.com/ultralytics/yolov5> [Accessed 23 June 2022].

[20] Paszke, A. et al., 2019. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In Advances in Neural Information Processing Systems 32. Curran Associates, Inc., pp. 8024–8035. Available at: http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf.

Acknowledgment

I would like to acknowledge assistance provided by the Faculty of Computing of General Sir John Kotelawala Defence University. I would like to express my sincere gratitude to my supervisors for their kind cooperation and support in completing this research.

Author Biographies

AMRNVB Pethiyagoda is currently a BSc. Undergraduate in Computer Engineering at the Faculty of Computing, General Sir John Kotelawala Defence University. Current research is about developing a Real-Time Vehicle Type Recognition and License Plate Recognition using Deep Learning. His research interests are Deep Learning, and Computer Vision.

Prof. TL Weerawardane is a Professor at Faculty of Engineering, General Sir Kotelawala Defence University. He has obtained BSc (Hons) in Electrical Engineering, University of Moratuwa, Sri Lanka, MSc in Information & Communication Technology and PhD in Mobile Communication, University of Bremen, Germany. His research interests are 3G/4G/5G Mobile Communication, Wireless Communication, Telecommunication and Communication Networks, Industrial Internet of Things, Data Science and Big Data Analytics, Artificial Intelligence and Cyber Security and Stochastic Simulations and Statistical Analysis.

DMR Kulasekara is a Senior Lecturer Grade II and Head of Department of Computer Engineering at Faculty of Computing, General Sir Kotelawala Defence University.
DMR Kulasekara obtained B.Sc (Hons) Specialized Computational Physics from University of Colombo, BSc. Information Technology (Specialized Computer Systems and Networks) from Sri Lanka Institute of Information Technology and M.Phil in Image processing & graphics from University of Colombo School of Computing, Sri Lanka. His research interests are Image Processing and Computer vision, Computer graphics, Artificial Intelligence.

D. M.W. P Maduranga is a Lecturer and Coordinator of Industrial Training at Faculty of Computer Engineering, General Sir Kotelawala Defence University. He obtained his BSc.Eng (Hons) in Electronic Engineering degree from Asian Institute of Technology (AIT), Thailand and MSc.Eng in Electrical and Electronic Engineering from the University of Peradeniya, Sri Lanka. He received Engineering Charter in Electronics and Telecommunication Engineering from the Engineering Council, the UK in 2020. His current research interests include Indoor Localization, Internet of Things and Wireless Communications.