

# Vehicle detection and classification using three variations of you only look once algorithm

Gehad Saleh Ahmed Mohammed, Norizan Mat Diah, Zaidah Ibrahim, Nursuriati Jamil

School of Computing Sciences, College of Computing, Informatics and Media, Universiti Teknologi MARA, Shah Alam, Malaysia

---

## Article Info

### Article history:

Received Oct 6, 2022

Revised Feb 18, 2023

Accepted Apr 19, 2023

---

### Keywords:

COCO dataset

Deep learning

Object detection

Vehicle classification

You only look once

---

## ABSTRACT

Vehicle detection and classification are essential for advanced driver assistance systems (ADAS) and even traffic camera surveillance. Yet, it is challenging due to complex backgrounds, varying illumination intensities, occlusions, vehicle size, and type variations. This paper aims to apply you only look once (YOLO) since it has been proven to produce high object detection and classification accuracy. There are various versions of YOLO, and their performances differ. An investigation on the detection and classification performance of YOLOv3, YOLOv4, and YOLOv5 has been conducted. The training images were from common objects in context (COCO) and open image, two publicly available datasets. The testing input images were captured on a few highways in two main cities in Malaysia, namely Shah Alam and Kuala Lumpur. These images were captured using a mobile phone camera with different backgrounds during the day and night, representing different illuminations and varying types and sizes of vehicles. The accuracy and speed of detecting and classifying cars, trucks, buses, motorcycles, and bicycles have been evaluated. The experimental results show that YOLOv5 detects vehicles more accurately but slower than its predecessors, namely YOLOv4 and YOLOv3. Future work includes experimenting with newer versions of YOLO.

*This is an open access article under the [CC BY-SA](https://creativecommons.org/licenses/by-sa/4.0/) license.*



---

## Corresponding Author:

Norizan Mat Diah

School of Computing Sciences, College of Computing, Informatics and Media

Universiti Teknologi MARA

40450 Shah Alam, Selangor, Malaysia

Email: norizan289@uitm.edu.my

---

## 1. INTRODUCTION

Perceiving their surroundings is one of the fundamental needs for autonomous vehicles with advanced driver assistance systems (ADAS). These vehicles use sensors including cameras, lidar, radars, and ultrasonic technology to gather environmental data. Cameras provide the greatest resolution and most detailed textural data on the environment of the vehicles when compared to other sensor types. However, deciphering camera-captured visual data is still a difficult task for computers to complete, particularly when real-time performance is needed [1]. This task is made more challenging by the fluctuating weather and illumination, complicated backgrounds, and object occlusions [1].

Proper solutions are already in place to address the automotive industry's traffic control issues. Toyota, for example, makes use of its technology [2]. Mobileye [3], NVidia [4], and StradVision [5] are just a few of the companies working on developing algorithms that can find objects using cutting edge of technology. Various studies have been conducted to detect and classify objects for autonomous vehicles, and supervised machine learning (SML) is one of the most popular techniques. By classifying input photos for SML, classifiers must extract features from the images. The histogram of oriented gradients, scale-invariant feature transform

(SIFT), and sped-up robust feature (SURF) are three of the most used features [6], [7]. Support vector machine (SVM), k-nearest neighbors (kNN), and artificial neural network (ANN) are three popular classifiers [8]. Although rapid, precise, and adaptive, these methods do not perform well for many features or occlusion [9].

Deep learning (DL) approaches are rapidly being used by researchers for automatic vehicle recognition and classification. ANN underlies the architecture of DL with layers that can learn and make intelligent selections on their own [10], [11]. Popular DL approaches for object recognition and classification include region-based convolutional neural network (R-CNN) [12], [13], Fast R-convolutional neural network (Fast R-CNN) [14], [15], faster R-convolutional neural network (faster R-CNN) [16], and you only look once (YOLO) [17]. These methods are adaptable, dependable, and precise [18]. Detecting diverse shapes, colours, and vehicle types, on the other hand, remains a significant challenge.

Building a robust object detection and classification algorithm is still a challenge today. Many of us have misunderstood the magnitude of this issue because it is easy for our eyes to detect and classify objects like vehicles coming in different shapes, colours, and types such as cars, lorries, and buses. Nonetheless, there are numerous challenges in designing a machine to design and mimic human eyes. One object detection and classification problem are the disparity in vehicle heights and shapes [19]. Vehicle sizes can vary in height and shape, not just for one type of car. When many vehicle entities have the same height and shape, it is hard for a computer system to find and sort them by figuring out how their shapes are different [20].

Another complex problem in the detection and classification process is occlusion. The amount of information in an image is limited by occlusion [21]. Occlusions can occur in real-world scenarios between vehicle types, such as cars, trucks, buses, and other objects, such as humans, animals, or buildings. These are all common types of occlusions. These occlusions can reduce the accuracy of detection and classification algorithms, particularly in a crowded environment [19]. Another challenging task is that different lighting conditions can affect the visibility and even the appearance of a vehicle. Figure 1 shows some occlusion samples impacting vehicle detection and classification performance. In this figure, the parked vehicles are occluded by other vehicles in a crowded car parking area. When combined with bad lighting caused by fogs, occlusion can be a major problem for accurate object detection and classification.



Figure 1. Examples of different occlusions that can affect vehicle detection and classification performance

Fast R-CNN has a few misclassification errors, such as misclassifying background patches in an image as vehicles because it cannot detect the larger context [14]. Faster R-CNN has been used for vehicle detection and tracking; however, the training and testing images were only captured during the day and not at night [22], even though detecting and classifying vehicles at night is just as important as during the day. YOLO is faster than faster R-CNN, but it cannot detect small objects such as vehicles farther away from the camera [23], [24]. YOLOv2 was created to address this limitation and can also detect and recognise objects with significant differences between classes, such as humans and cars, but not different types of vehicles [25]. Based on aerial images, a performance comparison of YOLOv3 and YOLOv4 for small object detection was conducted, and the results show that YOLOv4 has better accuracy but not speed [26], [27]. YOLOv5 has demonstrated exemplary performance in detecting vehicle front but not rear parts [28]. Regarding accuracy in detecting faulty unmanned aerial vehicles (UAVs), YOLOv5 outperforms YOLOv4 and YOLOv3 but has a slightly slower inference speed [27]. Thus, this study compares the performance of YOLOv3, YOLOv4, and YOLOv5 in terms of speed and detection accuracy based on videos captured using a mobile phone mounted on a car's dashboard in various views and different types of vehicles, during the day and at night.

## 2. THEORETICAL OVERVIEW

Due to its powerful learning capabilities and advantages in coping with occlusion, scale transformation, and background switches, DL-based object detection has been a prevalent study issue in recent years. DL-based object identification systems deal with sub-problems like occlusion, clutter, and low resolution [29]. Existing DL-based detectors are classified into two types. The first type of detector is a CNN-based two-stage detector, which has the advantage of accuracy, but not speed. The first step generates the proposed region, while the second stage classes and regresses the region, as in faster R-CNN. The second type of detector, YOLO, does classification and regression on dense anchor boxes without generating a sparse region of interest (RoI).

Previous convolutional neural network (CNN) models required a fixed input size; for example, AlexNet supports images up to  $224 \times 224$  pixels. The development of a spatial pyramid pooling (SPP) layer, which allows a CNN to produce a fixed-length representation without rescaling, regardless of the size of the image/region of interest, is the main feature of SPP-Net [30]. When using SPP-Net for object detection, the feature maps can only be created once from the entire image. Then, instead of computing completely convolutional features repeatedly, fixed-length representations of arbitrary regions can be used to train detectors. Without losing identification sensitivity (VOC07 mAP=59.2%), SPP-Net is 20 times faster than R-CNN. SPP-Net has successfully increased detection speed, although there are still certain limitations. For starters, training continues to be multi-staged. Second, SPP-Net fine-tunes the already connected layers while ignoring the ones that came before [31].

Lin *et al.* [32] proposed faster R-CNN-based feature pyramid networks (FPN). Most DL detectors only detect the top layer of a network before FPN. While CNN's deeper layers are excellent for recognising categories, finding objects is difficult. FPN does this by constructing high-level semantics at all sizes using a top-down architecture with lateral relations. Because CNN's forward propagation forms a function pyramid by default, FPN has demonstrated substantial performance in detecting objects of various scales. The Microsoft common objects in context (MSCOCO) dataset produces state-of-the-art single-model detection using FPN in a simple faster R-CNN method. FPN is now a fundamental component of several modern detectors [31]. Girshick *et al.* [12] suggested a multi-stage classification method based on the region's model, which was realised using the R-CNN. The national proposal element, the CNN vector extractor function, and the SVM classifier are all part of the framework.

As a result, CNN is more effective at removing attribute vectors from the past for both positive and negative soil-reality regions while reducing file size for the training phase. Then, in SVM classification processing, such functional vectors are used. The external region recommendation section uses a selective search to construct 2,000 category-dependent, fixed-size sections containing objects. Subsequently, the CNN extractor converts all feasible regions into vectors, which the SVM uses to define the domain. A linear regression model and an intersection over union (IoU) overlay with a superior score field optimise the bounding box and eliminate duplicate detections.

In comparison to other DL techniques, R-CNN achieves higher detection precision. However, R-dynamic CNN's multi-stage pipeline has significant drawbacks. CNN contains sensitive information. Adding specific regions with slow detection and identification times slows down the overall system and depends on region prediction. However, automating the multiple training methods for each part is difficult. In other words, the CNN element cannot be changed while the SVM classifier is being trained [33]. R-CNN and SPP-Net expansion can be considered. Fast R-CNN [14] was created to perform complete training and testing, while Fast R-CNN, including SPP-Net, switches the zone extraction feature and runs CNN for measurement sharing. Later, the final pooling is also tested to process images of any scale. Fast R-CNN, on the other hand, differs from SPP-multi-scale Net pooling in that it employs only one RoI pooling layer.

Because loss problems propagate back to updated convolutional layers and RoI bundling layers, this approach is advantageous for training. On each labelled RoI, Fast R-CNN also incorporates class score loss, boundary box loss, and multi-task lowering. As a result, the entire network can be trained. These innovations simplify training and evaluation while enhancing identification accuracy, according to the results. Accelerated network time exposes slow geographic notions as a bottleneck [34].

Faster R-CNN [15] successfully solved a region selection problem. It has established a national suggestion network to support the proposed regional mission without employing external field advisory methods but instead shares image convolutional calculation with a detection network. A Fast R-CNN is a region proposal network (RPN) for class evaluation. Aside from the feature diagram, each region is represented by an  $m \times n$  matrix with nine different area ratios and sizes. The RPN will incorporate both geographic theories to predict the appearance and location of artefacts. Then, the second Fast R-CNN can be applied to the RPN's high-scoring regions for additional grouping and bounding box optimisation. Faster R-CNN is the basis of good follow-up detection systems, using multiple data sets to find new objects accurately.

Liu *et al.* [34] introduced the single shot multibox detector (SSD). It is the second one-stage detector in computer vision. The fundamental contribution of SSD is the introduction of multi-reference and multi-resolution detection techniques, which considerably increase the detection precision of a single-stage detector, especially for small objects. SSD offers benefits in terms of detection speed and accuracy. The fundamental distinction between SSD and previous detectors is that the former detects objects at various scales on different network layers. On the other hand, the latter identifies things solely on their top layers [31].

RetinaNet was proposed by Lin *et al.* [35]. The primary cause is a significant foreground-background mismatch observed during dense detector preparation. RetinaNet has introduced a novel loss feature known as "focused loss" by altering the conventional cross-entropy loss during preparation to concentrate more on rough, misclassified examples. Due to focus loss, one-stage detectors may retain extremely fast detection speeds while achieving the excellent performance of two-stage detectors [36].

The YOLO algorithm is a collection of detectors first developed in 2016 by Redmon *et al.* [17], and it divides the input image into an S grid. It investigates using a network for real-time object detection using the one-stage detector to detect and recognise objects simultaneously. YOLO sensing speed is roughly ten times faster than other cutting-edge systems to a single architecture. The YOLOv1 algorithm comprises 24 convolutional layers, followed by two fully connected layers. Convolutions of size 11 are used in some convolutional layers to reduce the depth dimension of feature maps. Fast YOLO employs only nine convolutional layers, but it affects accuracy.

Redmon and Farhadi [23] recommended the current YOLOv2 edition. A new Darknet-19 network configuration is generated by deleting the network's entire connection layers and performing batch normalisation on each layer. In the anchor boxes of the faster R-CNN anchor mechanism, K-means clustering is employed. The predicted boxes have now been retrained using direct prediction. YOLOv2 outperforms YOLOv1 in terms of precision and speed of target recognition; it is faster but less accurate, making it unsuitable for sensitive tasks, including security and autonomous vehicles. Redmon and Farhadi [23] pioneered object detection by merging feature extraction and object localisation into a single monolithic block.

The first difference between YOLOv3 and previous models is using a multi-label grouping rather than a mutually exclusive label. Instead of the general mean square error used in previous versions, the binary cross-entropy loss for each label is used for classification loss. It employs a logistic classifier to assess the likelihood of the versions, with the softmax feature producing the score probabilities. Using various bounding box predictions is the second improvement. Unlike others, it assigns the objectless score of 1 to the bounding box anchor overlapping the ground truth object. It excludes anchors overlapping the ground truth object by more than the implementation's chosen threshold of 0.7. As a result, YOLOv3 assigns one bounding box anchor to each ground truth object. The third enhancement is feature pyramid networks for prediction across sizes. YOLOv3 predicts boxes on three different scales before deriving the characteristics from those scales [37].

The number of convolutional layers is added to the base feature extractor. Using the Darknet-53 framework as the foundation for YOLOv3, the number of convolutional layers is increased to 53. Three bounding boxes are predicted for each scale, and each bounding box's predicted classes have multiclass classifications. The network is a YOLOv2/Darknet-19 hybrid with  $3 \times 3$  and  $1 \times 1$  shortcut filters. It provides better features earlier in the network, and predictions from later layers benefit from early computing [38]. Figure 2 shows the Darknet-53 network, and Table 1 shows how different backbones work.

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

Figure 2. Darknet53 network [17]

Table 1. Comparison of backbones for various DL models in terms of accuracy, billions of operations (Bn Ops), billions of floating-point operations per second (BFLOP/s), and frames per second (FPS) [23]

Backbone	Top-1	Top-5	Bn Ops	BFLOP/s	FPS
Darknet-19 [18]	74.1	91.8	7.29	1246	171
ResNet-101 [38]	77.1	93.7	19.7	1039	53
ResNet-152 [38]	77.6	93.8	29.4	1090	37
Darknet-53	77.2	93.8	18.7	1457	78

Bochkovskiy *et al.* [39] proposed a YOLOv4 algorithm with significant changes and improved accuracy compared to previous versions. YOLOv4 is a significant improvement over YOLOv3; by implementing modern backbone architecture and neck improvements [40], mean average precision (mAP) and FPS accuracy have increased by 10% and 12%, respectively. It is now possible to train this neural network on a single GPU [39], [41]. It employs the dense block to construct a deeper and more complex network to gain more precision. The dense blocks input charts are divided into two parts by cross stage partial (CSP) connections. The first part is immediately guided to the next transformation layer, while the second moves through the thick block. The computing requirements are reduced because only one component passes through the dense block. It provides the foundation for feature extraction with the CSPDarknet-53, using CSP connections with the Darknet-53 from the previous YOLOv3. In YOLOv3, path aggregation network (PANet) is used instead of FPN.

YOLOv4 now contains more tools for constructing robust and reliable object detection models. It allows anyone to train a quick and reliable object detector utilising a 1080 Ti or 2080 Ti GPU. YOLOv4's architecture features a training mechanism dubbed Freebie's Bag, improving accuracy without requiring hardware, and it helps to increase the results while incurring no additional processing costs. The effect of cutting-edge bag-of-freebies and bag-of-specials object detection methods, such as cross-iteration batch normalisation (CBN), PANet, and spatial attention module (SAM), is also examined during detector training. It changes and increases their performances and applicability for single-GPU training [39].

There are additional improvements to the single GPU robust training implementation. They include adding new data increase methods (CutMix, mosaic, and others), selecting hyperparameters while implementing general algorithms, and adjusting existing methods to make their design best suited for detection and training. Concerning the YOLOv4 architecture, the CSP Darknet-53 backbone, additional SPP module, PANet neck, non-maximum suppression (Greedy NMS), and YOLOv3 (anchor-based) headers have been chosen [39].

ResNet, DenseNet, and VGG backbone models were used as feature extractors. They are fine-tuned on the detection dataset after being trained on datasets like ImageNet [42]. As the network becomes deeper (more layers), these networks generating different feature levels with higher semantics are useful for later sections of the object detection network [35], [43].

Between the backbone and the head, there are additional layers. Neck networks include FPN, PANet, and Bi-FPN, to name a few. At various backbone levels, they are used to extract various feature maps. For example, YOLOv3 uses FPN to remove features from the backbones of various scales [39], [43]. Ahead is a network detecting bounding boxes (classification and regression). Objected anchor-based detectors, such as YOLOv4, anchor-based, one-stage detectors, SSD, and RetinaNet [39], [43]. add the head network to each anchor box. A single output may look like four values representing the predicted bounding box (x, y, h, w) and the probability of k classes +1. (one extra, for background).

Bag of freebies (BoF) and bag of specials (BoS) are two strategies utilised in YOLOv4 to improve the object detector's precision (BoS). BoF enables the object detector to receive improved accuracy without increasing inference costs; it only alters the training plan or increases the training expenditures. BoF is used to augment data, improving the generalisability of the YOLOv4 model. Photometric distortions, such as adjusting the brightness, saturation, contrast, or noise, are elements of BoF, as are picture geometric distortions, such as rotating, cropping, and others [39], [43]. BoF approaches enhance detector accuracy.

BoS modestly raises the inference cost while significantly improving object detection accuracy. Adding attention mechanisms (squeeze-and-excite and spatial attention module), increasing the model's receptive field, and improving feature incorporation are examples of module/method categories [39], [43]. Figure 3 depicts the YOLOv4 architecture. Figure 4 illustrates the YOLOv5 architecture. YOLOv5 uses CSPDarknet53 as the backbone and PANet as the neck to boost the information flow. The head in YOLOv5 is the same as YOLOv4 and YOLOv3.

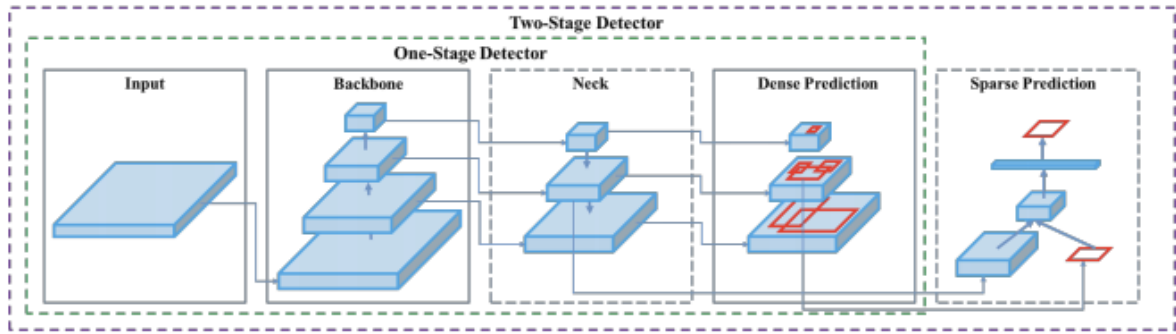


Figure 3. YOLOv4 architecture [32]

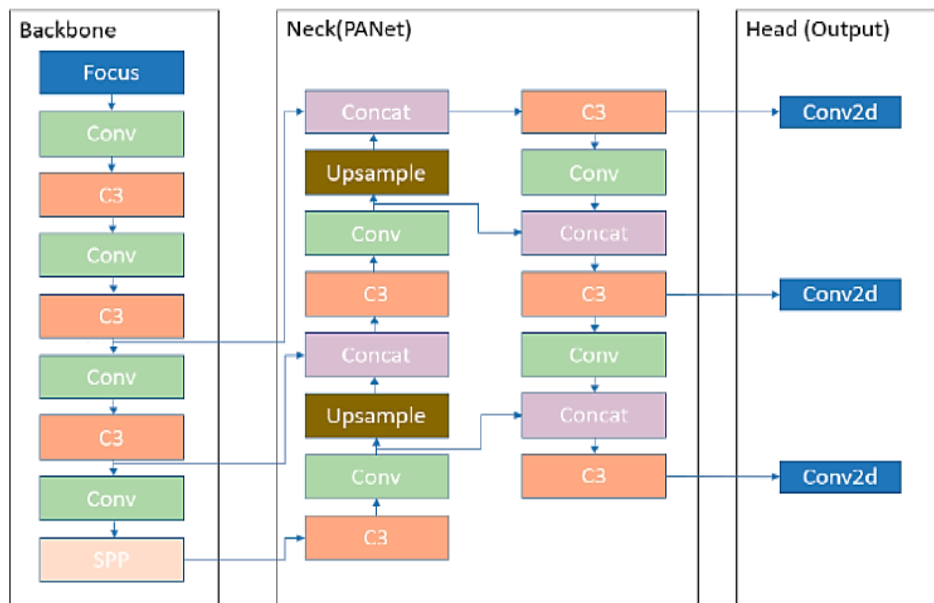


Figure 4. YOLOv5 architecture [21]

### 3. EXPERIMENTAL RESULT AND ANALYSIS

The open image [44] dataset is one of the most extensive datasets of images annotated in many ways to train computer vision tasks for deep CNN. The open image dataset integrates 9M images with annotated 36M labels, 15.8M bounding boxes, 2.8M instance divisions, and 391K visual relationships with the launch of version V6. Open images V6 represents critical qualitative and quantitative steps toward enhancing coherent image recognition, object identification, visual relationship detection, and instance segmentation. Images are modified in Roboflow [45]. It organises, prepares, and enhances the training data on photos and annotations and creates a higher computer vision model dataset. Roboflow supports object classification and detection models [45]. In this research, YOLOv3, YOLOv4, and YOLOv5 models were trained with 5,000 images of the COCO dataset in the system's default setting. In addition, 30 images captured within a few areas in Shah Alam and Kuala Lumpur (Figure 5) were also used. These images were converted into joint photographic experts group (JPEG) format before being fed into YOLOv3, YOLOv4, and YOLOv5. All additional images are captured using a mobile phone camera under natural illumination settings. Most images were captured under direct, bright sunlight and some were acquired under bridges or highways creating non-uniform illumination settings. While some vehicles are fully visible in the image, others included occluded vehicles. The different variations are to create image data collections for robust training datasets.

Figure 6 displays three examples of the test images. Figure 6(a) shows two fully visible vehicles captured from a bird's eye view. A motorcycle was also present partially occluded by the red car. Figure 6(b) with four cars is captured under low-intensity lighting and at night time. Figure 6(c), there are four motorcycles and two cars. One motorcycle is partly hidden behind the blue car and only a small part of another car is visible parked next to the blue car.





Figure 5. Sample images captured around Shah Alam and Kuala Lumpur, Malaysia



Figure 6. Samples of testing images captured in Kuala Lumpur where (a) shows two fully visible vehicles captured from a bird's eye view, (b) captured under low-intensity lighting and at night time, and (c) third test image, there are four motorcycles and two cars during day

Table 2 shows a few examples of object detection and classification using YOLOv3, YOLOv4, and YOLOv5. For the first test image named as image 1, YOLOv4 outperformed YOLOv3 in terms of classification and classification speed. YOLOv3 detected three objects and correctly classified only one object as a car. Meanwhile, YOLOv4 detected five objects, and both cars are correctly classified. YOLOv4 also is more efficient than YOLOv3 in which the detection and classification were done faster at 33.42 ms compared to YOLOv3 at 41.56 ms. YOLOv5 took the longest time at 154 ms, and six vehicles were detected with four misclassifications. Furthermore, the confidence scores for car classification were 99% and 65% compared to 99% confidence scores of YOLOv4 for both cars. For image 1, YOLOv4 seemed to outdo YOLOv3 and YOLOv5 both for detection, classification, and speed.

As for image 2, YOLOv5 performed the worst compared to YOLOv4 and YOLOv3. YOLOv5 required 190 ms to conduct the detection and classification as compared to YOLOv3 at 41.53 ms and YOLOv4 at 54.72. YOLOv3 detected five cars, YOLOv4 detected eight objects of which six were classified as cars and two as traffic lights. However, YOLOv5 only detected three objects of which two were classified as cars and one potted plant. YOLOv3 seemed to perform better than YOLOv4 and at a faster rate for image 2. It is also interesting to note that YOLOv4 classified two objects as traffic lights. Upon closer inspection, we believed that the two streetlights in image 2 were misclassified as two traffic lights.

YOLOv4 clearly performed the best for image 3. All six objects were detected, and all four motorcycles and two cars were classified correctly. The detection and classification were also done in 54.72 ms. YOLOv3 was also able to classify the four motorcycles and two cars successfully. However, a false negative object was also detected and classified as a person. YOLOv3 took longer than YOLOv4 with a speed of 91.30 ms. YOLOv5, however, performed poorly at with the slowest speed of 180 s.

Table 2. Detection and classification results for YOLOv3, YOLOv4, and YOLOv5

Model	Object detection	Confidence score	Speed (ms)
YOLOv3	 Image 1	Boat: 95% Boat: 39% Car: 89%	41.56
YOLOv4	 Image 1	Potted plant: 30% Potted plant: 89% Car: 99% Motorcycle: 50% Car: 99%	33.42
YOLOv5	 Image 1	Car: 99% Car: 65% Motorcycle: 99% Motorcycle: 96% Motorcycle: 99% Motorcycle: 97%	154
YOLOv3	 Image 2	Car: 56% Car: 58% Car: 99% Car: 75% Car: 92%	41.53
YOLOv4	 Image 2	Car: 73% Car: 58% Car: 95% Traffic light: 47% Car: 87% Car: 91% Traffic light: 61% Car: 36%	54.72
YOLOv5	 Image 2	Potted plant: 84% Car: 84% Car: 82%	190
YOLOv3	 Image 3	Motorcycle: 99% Motorcycle: 95% Person: 37% Motorcycle: 92% Motorcycle: 93% Car: 100% Car: 80%	91.30
YOLOv4	 Image 3	Motorcycle: 99% Motorcycle: 98% Motorcycle: 98% Motorcycle: 95% Car: 99% Car: 59%	54.72
YOLOv5	 Image 3	Car: 48% Car: 50%	180



#### 4. DISCUSSIONS AND FUTURE WORK

This paper discusses vehicle detection and classification achievements using DL techniques. It shows preliminary results from three DL techniques: YOLOv3, YOLOv4, and YOLOv5. Based on the experimental results, the three versions of YOLO performed differently depending on the size of objects in the image and the illumination setting. In an earlier study Jeong *et al.* [18] on aerial images, YOLOv4 has better accuracy but slower speed compared to YOLOv3. However, the same conclusion cannot be drawn based on our test images. While YOLOv3 performed faster in good illumination settings, it deteriorates when non-uniform lighting such as in image 2 and image 3. YOLOv5 has been shown to be slower in [21] and the same finding is also derived in this research. YOLOv5 also has shown unpredictable results with misclassifications and has a lower detection rate compared to YOLOv3 and YOLOv4, especially in non-uniform lighting and occluded objects. Figure 6 also shows that YOLOv4 can detect more objects than YOLOv3 and YOLOv5. Future work will improve YOLOv5's efficiency and detection performance for occlusion and apply it to vehicle counting applications.

#### 5. CONCLUSION

This study presents the results of vehicle detection and classification using DL approaches. It shows preliminary findings from three DL techniques: YOLOv3, YOLOv4, and YOLOv5. For the training, two separate datasets were used. The first dataset was COCO, which was trained with the classes' default values of 80, and the second dataset was Open Image, which was trained with only four classes. The experimental findings with thirty photographs of different types of cars obtained in Kuala Lumpur and Shah Alam, Selangor, during the day and at night show that YOLOv5 recognises vehicles quite successfully but at a slower rate than its predecessors. However, recognising vehicles with occlusion for YOLOv5 still has potential for improvement. Future work will focus on improving the efficiency of YOLOv5 and its detection performance for occlusion and applying it to vehicle counting applications.

#### ACKNOWLEDGEMENTS



The College of Computing, Informatics and Media, Universiti Teknologi MARA, Shah Alam, Selangor, Malaysia is acknowledged for supporting this work.

#### REFERENCES




- [1] J. White, T. Kameneva, and C. McCarthy, "Vision processing for assistive vision: a deep reinforcement learning approach," *IEEE Transactions on Human-Machine Systems*, vol. 52, no. 1, pp. 123–133, Feb. 2022, doi: 10.1109/THMS.2021.3121661.
- [2] H. Saribas, H. Cevikalp, and S. Kahvecioglu, "Car detection in images taken from unmanned aerial vehicles," in *26th IEEE Signal Processing and Communications Applications Conference, SIU 2018*, May 2018, pp. 1–4, doi: 10.1109/SIU.2018.8404201.
- [3] "Driving the autonomous vehicle evolution," Mobileye, 2022, Accessed: Feb. 12, 2022. [Online]. Available: <https://www.mobileye.com/>.
- [4] "In the Nvidia Studio," Nvidia, 2022 Accessed: Feb. 13, 2022. [Online]. Available: [www.nvidia.com](http://www.nvidia.com).
- [5] "AI assisted driving for everyone," Stradvision, 2022, Accessed: Mar. 01, 2022. [Online]. Available: <https://www.appengine.ai/company/stradvision>.
- [6] V. L. Surekha and M. Pradeep, "A new advanced approach for content based image retrieval using texton pattern," *International Journal of Research Science & Management*, vol. 4, no. 6, pp. 129–135, 2017, doi: 10.5281/zenodo.820831.
- [7] D. Srinivas and K. Hanumaji, "Analysis of various image feature extraction methods against noisy image: SIFT, SURF and HOG," *Journal of Engineering Sciences*, vol. 10, no. 2, pp. 32–36, 2019, doi: 10.15433/JES.2019.V10I2.43P.7.
- [8] E. Y. Boateng, J. Otoo, and D. A. Abaye, "Basic tenets of classification algorithms K-nearest-neighbor, support vector machine, random forest and neural network: A review," *Journal of Data Analysis and Information Processing*, vol. 08, no. 04, pp. 341–357, 2020, doi: 10.4236/jdaip.2020.84020.
- [9] S. Tang, M. Andriluka, and B. Schiele, "Detection and tracking of occluded people," *International Journal of Computer Vision*, vol. 110, no. 1, pp. 58–69, 2014, doi: 10.1007/s11263-013-0664-6.
- [10] D. Neupane and J. Seok, "A review on deep learning-based approaches for automatic sonar target recognition," *Electronics (Switzerland)*, vol. 9, no. 11, pp. 1–30, Nov. 2020, doi: 10.3390/electronics9111972.
- [11] I. H. Sarker, "Deep learning: a comprehensive overview on techniques, taxonomy, applications and research directions," *SN Computer Science*, vol. 2, no. 6, p. 420, Nov. 2021, doi: 10.1007/s42979-021-00815-1.
- [12] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Jun. 2014, pp. 580–587, doi: 10.1109/CVPR.2014.81.
- [13] S. M. Alkentar, B. Alsahwa, A. Assalem, and D. Karakolla, "Practical comparison of the accuracy and speed of YOLO, SSD and Faster RCNN for drone detection," *Journal of Engineering*, vol. 27, no. 8, pp. 19–31, Aug. 2021, doi: 10.31026/j.eng.2021.08.02.
- [14] R. Girshick, "Fast R-CNN," in *Proceedings of the IEEE International Conference on Computer Vision*, Dec. 2015, vol. 2015 Inter, pp. 1440–1448, doi: 10.1109/ICCV.2015.169.
- [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, Jun. 2017, doi: 10.1109/TPAMI.2016.2577031.

- [16] Q. Ren, J. Geng, and J. Li, "Slighter Faster R-CNN for real-time detection of steel strip surface defects," in *Proceedings 2018 Chinese Automation Congress, CAC 2018*, Nov. 2019, pp. 2173–2178, doi: 10.1109/CAC.2018.8623407.
- [17] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: unified, real-time object detection," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Jun. 2016, vol. 2016-Decem, pp. 779–788, doi: 10.1109/CVPR.2016.91.
- [18] H. J. Jeong, K. S. Park, and Y. G. Ha, "Image preprocessing for efficient training of YOLO deep learning networks," in *Proceedings - 2018 IEEE International Conference on Big Data and Smart Computing, BigComp 2018*, Jan. 2018, pp. 635–637, doi: 10.1109/BigComp.2018.00113.
- [19] P. Chong and Y. H. Tay, "A novel pedestrian detection and tracking with boosted HOG classifiers and Kalman filter," in *Proceedings - 14th IEEE Student Conference on Research and Development: Advancing Technology for Humanity, SCORED 2016*, Dec. 2017, pp. 1–5, doi: 10.1109/SCORED.2016.7810052.
- [20] Y. Song, J. Yao, Y. Ju, Y. Jiang, and K. Du, "Automatic detection and classification of road, car, and pedestrian using binocular cameras in traffic scenes with a common framework," *Complexity*, vol. 2020, pp. 1–17, May 2020, doi: 10.1155/2020/2435793.
- [21] H. Chandel and S. Vatta, "Occlusion detection and handling: a review," *International Journal of Computer Applications*, vol. 120, no. 10, pp. 33–38, Jun. 2015, doi: 10.5120/21264-3857.
- [22] Y. Zhang, J. Wang, and X. Yang, "Real-time vehicle detection and tracking in video based on faster R-CNN," *Journal of Physics: Conference Series*, vol. 887, no. 1, p. 012068, Aug. 2017, doi: 10.1088/1742-6596/887/1/012068.
- [23] J. Redmon and A. Farhadi, "YOLO9000: better, faster, stronger," in *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, Jul. 2017, vol. 2017-Janua, pp. 6517–6525, doi: 10.1109/CVPR.2017.690.
- [24] Y. Zhang, Z. Guo, J. Wu, Y. Tian, H. Tang, and X. Guo, "Real-time vehicle detection based on improved YOLO v5," *Sustainability (Switzerland)*, vol. 14, no. 19, p. 12274, Sep. 2022, doi: 10.3390/su141912274.
- [25] J. Sang *et al.*, "An improved YOLOv2 for vehicle detection," *Sensors (Switzerland)*, vol. 18, no. 12, p. 4272, Dec. 2018, doi: 10.3390/s18124272.
- [26] H. Xu, Y. Cao, Q. Lu, and Q. Yang, "Performance comparison of small object detection algorithms of UAV based aerial images," in *Proceedings - 2020 19th Distributed Computing and Applications for Business Engineering and Science, DCABES 2020*, Oct. 2020, pp. 16–19, doi: 10.1109/DCABES50732.2020.00014.
- [27] U. Nepal and H. Eslamiat, "Comparing YOLOv3, YOLOv4 and YOLOv5 for autonomous landing spot detection in faulty UAVs," *Sensors*, vol. 22, no. 2, p. 464, Jan. 2022, doi: 10.3390/s22020464.
- [28] M. Kasper-Eulaers, N. Hahn, P. E. Kummervold, S. Berger, T. Sebulonsen, and Ø. Myrland, "Short communication: detecting heavy goods vehicles in rest areas in winter conditions using YOLOv5," *Algorithms*, vol. 14, no. 4, p. 114, Mar. 2021, doi: 10.3390/a14040114.
- [29] Z. Q. Zhao, P. Zheng, S. T. Xu, and X. Wu, "Object detection with deep learning: a review," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 30, no. 11, pp. 3212–3232, Nov. 2019, doi: 10.1109/TNNLS.2018.2876865.
- [30] K. He, X. Zhang, S. Ren, and J. Sun, "Spatial pyramid pooling in deep convolutional networks for visual recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, no. 9, pp. 1904–1916, Sep. 2015, doi: 10.1109/TPAMI.2015.2389824.
- [31] D. Zhou *et al.*, "IoU loss for 2D/3D object detection," in *Proceedings - 2019 International Conference on 3D Vision, 3DV 2019*, Sep. 2019, pp. 85–94, doi: 10.1109/3DV.2019.00019.
- [32] T. Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, Jul. 2017, vol. 2017-Janua, pp. 936–944, doi: 10.1109/CVPR.2017.106.
- [33] J. R. R. Uijlings, K. E. A. Van De Sande, T. Gevers, and A. W. M. Smeulders, "Selective search for object recognition," *International Journal of Computer Vision*, vol. 104, no. 2, pp. 154–171, Sep. 2013, doi: 10.1007/s11263-013-0620-5.
- [34] W. Liu *et al.*, "SSD: Single shot multibox detector," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 9905 LNCS, 2016, pp. 21–37, doi: 10.1007/978-3-319-46448-0\_2.
- [35] T. Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollar, "Focal loss for dense object detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 42, no. 2, pp. 318–327, Feb. 2020, doi: 10.1109/TPAMI.2018.2858826.
- [36] M. Haris and A. Glowacz, "Road object detection: A comparative study of deep learning-based algorithms," *Electronics (Switzerland)*, vol. 10, no. 16, p. 1932, Aug. 2021, doi: 10.3390/electronics10161932.
- [37] B. Benjdira, T. Khursheed, A. Koubaa, A. Ammar, and K. Ouni, "Car detection using unmanned aerial vehicles: comparison between faster R-CNN and YOLOv3," in *2019 1st International Conference on Unmanned Vehicle Systems-Oman, UVS 2019*, Feb. 2019, pp. 1–6, doi: 10.1109/UVS.2019.8658300.
- [38] P. Mahto, P. Garg, P. Seth, and J. Panda, "Refining Yolov4 for vehicle detection," *International Journal of Advanced Research in Engineering and Technology*, vol. 11, no. 5, pp. 409–419, 2020, doi: 10.34218/IJARET.11.5.2020.043.
- [39] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "YOLOv4: optimal speed and accuracy of object detection," *arXiv:2004.10934*, Apr. 2020.
- [40] C. Zhang, F. Kang, and Y. Wang, "An improved apple object detection method based on lightweight YOLOv4 in complex backgrounds," *Remote Sensing*, vol. 14, no. 17, 2022, doi: 10.3390/rs14174150.
- [41] P. Rugery, "Explanation of YOLO V4 a one stage detector," *Becominghuman*, 2020, Accessed: Apr. 12, 2022. [Online]. Available: <https://becominghuman.ai/explaining-yolov4-a-one-stage-detector-cdac0826cbd7>.
- [42] B. Xu *et al.*, "Evaluation of deep learning for automatic multi-view face detection in cattle," *Agriculture (Switzerland)*, vol. 11, no. 11, 2021, doi: 10.3390/agriculture11111062.
- [43] A. Anka, "YOLO v4: Optimal speed and accuracy for object detection," *Medium*, 2020. Accessed: Apr. 20, 2022. [Online]. Available: <https://towardsdatascience.com/yolo-v4-optimal-speed-accuracy-for-object-detection-79896ed47b50>.
- [44] I. Krasin *et al.*, "Open images dataset V6+extensions," 2017. Accessed: Apr. 05, 2022. [Online]. Available: <http://storage.googleapis.com/openimages/web/index.html>.
- [45] "Label faster. Train smarter. Deploy anywhere," *Roboflow*. Accessed: Mar. 05, 2022. [Online]. Available: <https://app.roboflow.ai>.




**BIOGRAPHIES OF AUTHORS**

**Gehad Saleh Ahmed Mohammed**    received his B.Eng. degree in Electrical and Electronic Engineering at Segi University Malaysia dual award with the University of Sunderland UK, and M.Sc. in Computer Science at Universiti Teknologi MARA (UiTM) Malaysia. He has working experiences as an Automation Engineer at Ideasparq Robotics Sdn. Bhd. And an instructor at Robotics and Coding at The Forest 163. His research lines are in image recognition and classification, robotics, and automation. He can be contacted at email: gehad1433@hotmail.com.






**Norizan Mat Diah**    is a Senior Lecturer at the School of Computing Sciences, College of Computing, Informatics and Media, Universiti Teknologi MARA, Shah Alam Selangor. She holds a Ph.D. in Information Sciences from the National University of Malaysia. Her research focuses are on gamification, game engines, real-time feedback (Algorithm), handwriting recognition (Algorithm) and game theory. She can be contacted at email: norizan289@uitm.edu.my.



**Zaidah Ibrahim**    is currently an Associate Professor at the School of Computing Sciences, College of Computing, Informatics and Media, Universiti Teknologi MARA, Shah Alam, Selangor, Malaysia. Her research areas are computer vision and deep learning. She can be contacted at email: zaida782@uitm.edu.my.



**Nursuriati Jamil**    is a professor from the School of Computing Sciences, College of Computing, Informatics and Media, Universiti Teknologi MARA, Shah Alam. She specializes in image and speech processing research and has been awarded international, industry, and national grants for fundamental and social research. She is a senior member of IEEE and has been involved in awarding student awards for IEEE Computer Society. She is a member of IEEE. She can be contacted at email: lizajamil@computer.org.