JSR Publications

Available Online

J. Sci. Res. **13** (3), 809-820 (2021)

JOURNAL OF SCIENTIFIC RESEARCH

www.banglajol.info/index.php/JSR

Comparative Analysis on Deep Learning Approaches for Heavy-Vehicle Detection based on Data Augmentation and Transfer-Learning techniques

V. Sowmya[1*], R. Radha[2]

[1]Research Department of Computer Science, SDNBV College for Women, University of Madras, Chennai, India

[2]Research Department of Computer Science, SDNBV College for Women, Chennai, India

Received 7 March 2021, accepted in final revised form 2 June 2021

Abstract

Vehicle detection and recognition require demanding advanced computational intelligence and resources in a real-time traffic surveillance system for effective traffic management of all possible contingencies. One of the focus areas of deep intelligent systems is to facilitate vehicle detection and recognition techniques for robust traffic management of heavy vehicles. The following are such sophisticated mechanisms: Support Vector Machine (SVM), Convolutional Neural Networks (CNN), Regional Convolutional Neural Networks (R-CNN), You Only Look Once (YOLO) model, etcetera. Accordingly, it is pivotal to choose the precise algorithm for vehicle detection and recognition, which also addresses the real-time environment. In this study, a comparison of deep learning algorithms, such as the Faster R-CNN, YOLOv2, YOLOv3, and YOLOv4, are focused on diverse aspects of the features. Two entities for transport heavy vehicles, the buses and trucks, constitute detection and recognition elements in this proposed work. The mechanics of data augmentation and transfer-learning is implemented in the model; to build, execute, train, and test for detection and recognition to avoid over-fitting and improve speed and accuracy. Extensive empirical evaluation is conducted on two standard datasets such as COCO and PASCAL VOC 2007. Finally, comparative results and analyses are presented based on real-time.

Keywords: Deep-learning; Data augmentation; Transfer-learning; Vehicle detection.

doi: http://dx.doi.org/10.3329/jsr.v13i3.52332 J. Sci. Res. **13** (3), 809-820 (2021)

1. Introduction

The modern transportation infrastructure requires advanced technology to enable a smooth traffic management ecosystem to reduce traffic congestion caused by various classes of vehicles. One of the major contributors to this challenge is heavy vehicles. The heavy vehicles use existing regular roads that are more prone to space constraints and congestion of the roads leading to several challenges. The building of a road system does incur a high budget, and there are a lot of challenges like space allocation, geographic and climatic constraints. During rush hours, much congestion is just caused by heavy vehicles. A heavy

[*] *Corresponding author*: v.sowmy81@yahoo.in

vehicle detection system is the solution for efficient traffic management. Our proposed system is solution-oriented, enabling heavy vehicle detection and recognition to address the challenges above.

The fine-tuned deep learning system of heavy vehicle detection is implemented in the transportation ecosystem to source attributes of vehicles like logo, model, production year, vehicle make, max speed, and so on. By dynamically populating and building a database with this information, we monitor the entire real-time transportation of a particular high-density geographic location. As a result, these systems facilitate us to analyze the movement of heavy vehicles on the roads that will further contribute to smart transportation systems by establishing new traffic rules.

The principal objective of the proposed system is to identify and detect heavy vehicles in a video frame. The process of identification and detection is very resource-intense requires maximum computational power. Consequently, advanced techniques such as data augmentation and transfer-learning are applied to deep learning detection algorithms, namely Faster R-CNN, YOLOv2, YOLOv3, YOLOv4 to achieve better accuracy and performance than other state-of-approaches.

The rest of the paper is organized as follows: section 2 outlines related works based on vehicle detection. Section 3 provides a framework and methodology used in the proposed system. Experimental results will be provided in section 4, followed by a conclusion in section 5.

2. *Related Works*

Suhail *et al.* [1] propose a study on CNN-enabled object detection. The three approaches, namely, objectness, category-specific, and salient, are applied in CNN for object detection. Object detection leverages distinct objects that are highlighted and detected. The differentiation of the images is based on category-specific detection, which is one of the core principles segmented into classes using a pre-defined function. An advanced model-RCNN, Faster R-CNN that is enhanced CNN framework are used in the system. Kumar *et al.* [2] distinguish several models of deep learning object detection. The proposed system applies YOLO, RCNN, Fast RCNN, and Faster R-CNN models. Latency, frames per second, and Mean Average Precision (mAP) are defined as the evaluation metrics. Besides, the model's method-adoption function decides whether a real-time or static detection approach must be utilized is predicted, and it contributes to the effectiveness of the YOLO model in real-time detection. Chate *et al.* [3] proposes a model for heavy-vehicle detection applying CNN. The model uses the CIFAR10 dataset. This system uses transfer-learning methods and fine-tuned attributes for detecting heavy vehicles and work to achieve good accuracy.

Kawakatsu *et al.* [4] introduce a bridge weigh-in-motion technique that uses a deep CNN for detecting heavy vehicles. The surveillance camera enables the CNN to adapt to the real traffic conditions and by utilizing a single-strain sensor, high accuracy is achieved. The test results consist of vehicle attributes such as speed, axle positions, and loads that are measured. Ucar *et al.* [5] propose a hybrid system that integrates SVM and

CNN for autonomous driving applications. Initially, the CNN framework is employed on each image and then it segments them into local regions and forms multiple features. The PCA facilitates final features. Finally, the SVM classifier is applied to augment the efficiency and generalization of the proposed system. The proposed framework is tested with the CalTech pedestrian benchmark dataset. The accuracy of 92.80 @0.50 for 30 images is accomplished.

To surmount the above inefficiencies in the core aspects of speed, accuracy, and overfitting problems in regular and real-time mode in this work, a heavy-vehicle detection system coupled with different deep learning approaches is proposed with real-time capabilities. The submitted paper describes the deep learning techniques by using various algorithms such as Faster R-CNN, YOLOv2, YOLOv3, YOLOv4 for detecting the heavy-vehicle. Besides, the proposed systems further integrate data augmentation and transfer learning techniques for heavy-vehicle detection. These features' implementation enhances the effectiveness of the system in speed, accuracy and avoids overfitting during the training phase. The fine-tuning of the algorithmic mechanism of the framework is achieved by adopting the transfer learning technique and applying the freeze and unfreeze methods on certain CNN layers for optimization of the deep learning vehicle detection algorithm. Besides, fine-tuning is applied to all the above algorithms to adjust the hyper-parameters like learning rate, method of parameter initialization, etc. Finally, an OpenCV algorithm is used for testing the proposed model in a real-time environment.

3. Proposed Framework and Methodology

The architecture of the proposed framework is shown in Fig. 1. In this proposed system, we have captured a video of one-minute duration at different places in real-time, including different commuting vehicles. Then data pre-processing is performed to extract images from video frames and create a custom dataset of 3500 images that consists of bus and truck (heavy-vehicles). For each image, create an annotation file further integrate with the data augmentation technique. Then, Annotated and Augmented custom dataset classifies further into the training and testing dataset with 2450 and 1050 images (7:3), respectively. In the training dataset, the deep learning techniques using various algorithms such as Faster R-CNN, YOLOv2, YOLOv3, and YOLOv4 are applied to detect heavy vehicles. Train the parameters with the transfer learning technique and fine-tune the network parameters to avoid overfitting and increase optimal speed and accuracy. Also, non-maximum suppression (NMS) is applied for deleting the overlapping boxes and helps to predict the heavy vehicle. Finally, test the algorithms with the benchmark dataset and then compare the outcome with proposed models.

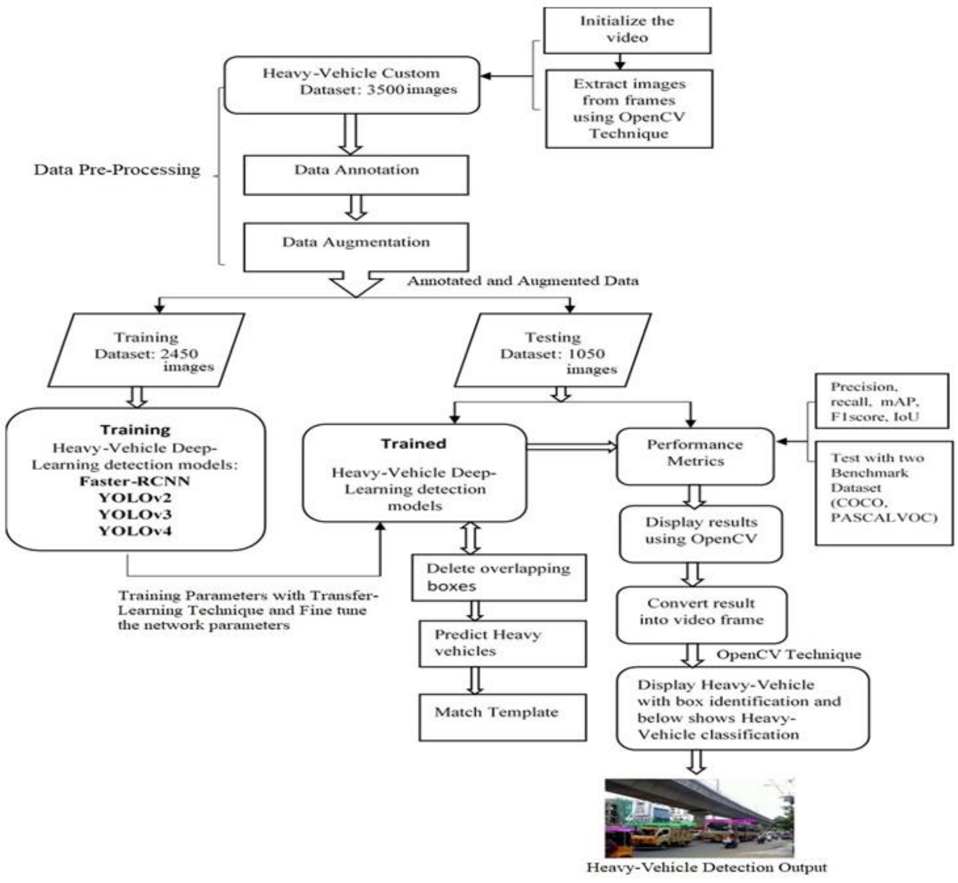

Initialize the video
Extract images from frames using OpenCV Technique
Heavy-Vehicle Custom Dataset: 3500images
Data Pre-Processing
Data Annotation
Data Augmentation
Annotated and Augmented Data
Training Dataset: 2450 images
Testing Dataset: 1050 images
Training Heavy-Vehicle Deep-Learning detection models: Faster-RCNN YOLOv2 YOLOv3 YOLOv4
Trained Heavy-Vehicle Deep-Learning detection models
Performance Metrics
Precision, recall, mAP, F1score, IoU
Test with two Benchmark Dataset (COCO, PASCALVOC)
Display results using OpenCV
Convert result into video frame
OpenCV Technique
Display Heavy-Vehicle with box identification and below shows Heavy-Vehicle classification
Delete overlapping boxes
Predict Heavy vehicles
Match Template
Training Parameters with Transfer-Learning Technique and Fine tune the network parameters
Heavy-Vehicle Detection Output

Fig. 1. Heavy-vehicle detection framework using various deep-learning models.

3.1. *Transfer learning and data-augmentation*

In the proposed system, the core functional aspect of transfer learning is its mechanism with features to form a new problem; these features are learned and transferred from a pretrained network. Transfer learning attributes to the process of simulating a pre-compressed CNN, which substitutes certain layers in the dataset that comprises the last convolutional layers and the formation of such layers. The transfer learning technique is applied to freeze and unfreeze certain CNN layers for optimization of the deep learning vehicle detection algorithm. Using freezing techniques, image features like edges are extracted by deploying deep CNN and the fully connected layers utilize this information to classify data pertinent to the problem. Also, fine-tuning is applied to all the above algorithms to adjust the hyper-parameters like learning rate, parameter initialization, etc.

Data augmentation is an approach that significantly contributes to the multifariousness of existing data for training and testing prototype frameworks without necessarily collating new sets of data. In this proposed model, data augmentation techniques such as horizontal flipping, padding, and cropping are generally applied to train large-scale neural networks.

3.2. ***Faster R-CNN***

The Faster R-CNN was introduced by Girshick *et al.* [6] in the year 2016. Girshick's RCNN system employs Regional Proposal Network (RPN) structures to detect objects. In our proposed system, to achieve a high detection ratio and precision-oriented, we have applied data augmentation, machine learning, and bottleneck features on the based model. Initially, RPN and the base network are trained with identified parameters for an effective training process. The rapid selection process is achieved by deploying a sub-convolutional network by generating regions of interest. Next, integrated with RPN, this model utilizes the anchor boxes to handle the scale of objects and the multiple aspect ratios. Then, rpn_min_overlap=0.5 and rpn_max_overlap=0.8 form the range of proposals as determined by the RPN to distinguish 'positive', 'negative' for each of the anchor. The output of the final region feature map is 7×7×512. Then, it is processed for training with the classifier, including two fully connected layers and with the necessary dropout. The final execution is the implementation of a softmax classifier for classification for detection of the vehicle and linear activation for regression to generate a filtered bounding box including the weights. The Adam algorithm further enhances the model by optimizing the loss functions. The initial learning rate is 0.001, with a decay of 0.0005 per batch. Later, the network is set to an iterative training process for 100 epochs.

3.3. ***YOLOv2***

The YOLOv2 was introduced by Redmon *et al.* [7] in the year 2016. In our proposed system, the YOLOv2 uses anchor boxes, batch normalization, fine-grained features, multi-level classifiers, high-resolution classifiers, and Darknet19. All these features provide a framework for an enhanced v2 model with significant benefits than v1. For the classification of objects, the Darknet19 feature extractor uses its softmax layer, 5 max-pooling layers, and 19 convolutional in an identified target image. Further, YOLOv2 applies k-means clustering algorithms, which contribute to great IOU scores.

The network is capable of multiscale classification, which is achieved by changing the input image dimension while training. Consequently, the input size equates to $416 \times 416 \times 3$, which produces an odd number of spatial dimensions. The odd number grid cell spatial projections on the object are very targeted where it fits. It eliminates one pooling layer to modify spatial network output to $13 \times 13 \times 40$. To start fine-tuning, the YOLOv2 model is tuned with corresponding weights and a .cfg file. Next, by applying the transfer learning technique, train a model with freezing and unfreezing the convolutional layers with the learning rate 0.0001 and 0.001. Finally, the best weights are saved for object detection. Eventually, the number of epochs amounts to 1080 times.

3.4. *YOLOv3*

The YOLOv3 utilizes a feature extractor called Darknet53, which comprises 53 convolutional layers, each enabled with Leaky ReLU activation and batch normalization layer. Void pooling techniques, a convolutional layer of the configuration of stride two is employed to downsample feature maps. Annotation files for each of the images and label files are generated and are bind together. To enhance the configuration, the three YOLO layers are trained applying yolo.cfg file. During the training process, each object goes through iterations of at least 2000 times. For optimum training speed, the parameter values of batch and subdivisions are set to 64, and 8 respectively. The width and height values were set at 416×416 each for maximum speed and greater accuracy of detection. The no. of filters used in the convolution layer is tuned to 21 and the algorithm applies a pre-trained network Darknet53. conv.74. Finally, tuned the YOLOv3 algorithm's network hyper-parameters. The batch normalization layer is frozen during the training process. To start fine-tuning, the YOLOv3 model is tuned with corresponding weights and a .cfg file. Next, by applying the transfer learning technique, train a model with freezing and unfreezing the convolutional layers with the learning rate 0.0001 and 0.001. Finally, the best weights are saved for object detection. Initially, the iterations are set to 2000 times with a learning rate of 0.0001 and also to evaluate the performance metrics.

3.5. *YOLOv4*

Bochkovskiy *et al*. [8], in the year April 2020 introduced a YOLOv4 model that integrates CSPdarknet53 as a backbone network that constitutes darknet53 and CSPNet. CSPdarknet53 is implemented on a dense net and is applied as a feature extractor. The dense net is programmed to communicate the layers with CNN for the benefit of reduction in the number of parameters, solving gradient problems, and also network features are reused. Each convolutional layer contains batch normalization, convolution, and Leaky Relu. Spatial Pyramid Pooling (SPP) is employed in RCNN (Regional Convolution Neural Network), YOLOv4 selects PAN replacing FPN (Feature Pyramid Network) that is used in YOLOv3. The 3 YOLO CNN layers are trained to utilize yolo.cfg file for optimum configurations. The dataset goes through 1000 iterations. For effective training speed, the parameters of batch and subdivisions are equated to 64 and 16 in the system, respectively. The width and height parameters are set to 416×416 each for the best speed and higher accuracy of detection. The number of filters is set to 21 in the convolution layer. The algorithm applies pre-trained network yolov4. conv.137. To start fine-tuning, the YOLOv4 model is tuned with corresponding weights and a .cfg file. Next, by applying the transfer learning technique, train a model with freezing and unfreezing the convolutional layers with the learning rate 0.0001 and 0.001. Finally, the best weights are saved for object detection.

3.6. *Dataset*

The custom vehicle dataset contains 3500 images of buses and trucks, downloaded from various sources such as icrawlers, the internet, and real-time video (converted into images). The dataset consists of two different types of image entity which equates to a ratio – 7:3 for training and testing. COCO and PASCAL VOC 2007 are the two standard benchmark datasets are used to test the proposed model

4. Experimental Results

The proposed system uses transfer learning and data augmentation techniques in deep learning vehicle detection algorithms such as Faster R-CNN, YOLOv2, YOLOv3, and YOLOv4. The mAP, IoU, FPS, precision, recall, and F1score are the metrics used for improving the performance of the proposed model.

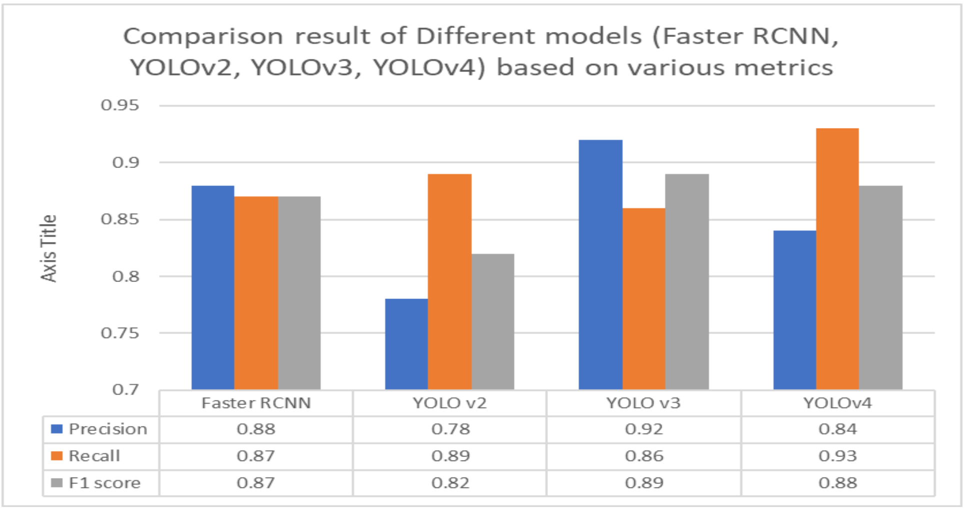
Precision is defined as the ratio of accurately predicted positive experimental output data to the total predicted positive experimental output data. The recall is defined as the ratio of accurately predicted positive experimental output data to all experimental output data in the actual class. F1 score is defined as the (weighted) average of recall and precision. Intersection over Union (IoU) is an indicative value that defines how near the predicted image is corresponding to the original picture [9]. The average mean Average Precision (mAP) is defined as the recovery detection bounding boxes and product accuracy. It is a precision measurement of the sensitivity of the network to the objects and its efficiency in avoiding false signals. The effectiveness of the precision of the network is based on the score of the mAP reaching a higher value, and the execution speed is cost-bound. The overall processing speed of the system is based on the processed frames per second (FPS) [10]. Table 1 shows the comparison of different models concerning latency, mean Average Precision (mAP), Frames Per Second (FPS), IoU, inference time, and AP of bus and truck. The YOL0v4 model achieves better results on the following attributes; low IoU with a score of 72.41, AP scores of 98 % for bus and 93 % for truck, and higher mAP with a score of 96.54 %. The Faster R-CNN shows good results for Avg_FPS with a score of 17 and inference time with a score of 0.42, but it is not well-suited with the real-time environment in comparison with all the real-time YOLO models.

Table 1. Comparison results of different models (Faster R-CNN, YOLOv2, YOLOv3, YOLOv4).

Models	mAP	IoU	Inference time	Avg_FPS	AP	
					Bus	Truck
Faster R-CNN	87.65	65.48	0.42s	**17**	88	81
YOLOv2	80.15	62.14	1s	45	86	83
YOLOv3	95.14	**81.84**	1s	35	**99**	91
YOLOv4	**96.54**	72.41	1s	34.2	**98**	**93**

In Fig. 2, the comparison of various metrics such as precision, recall, F1 Score, mAP implementing the Faster R-CNN, YOLOv2, YOLOv3, and YOLOv4 models are shown.

The model YOLOv3 shows better results for precision with a score of 0.92, the model. YOLOv4 shows an enhanced score for recall score of 0.93, and the F1 score is equally good on YOLOv2 and YOLOv3.



	Faster RCNN	YOLO v2	YOLO v3	YOLOv4
■ Precision	0.88	0.78	0.92	0.84
■ Recall	0.87	0.89	0.86	0.93
■ F1 score	0.87	0.82	0.89	0.88

Fig. 2. Comparison of various metrics (precision, recall, F1 score) implementing the Faster R-CNN, YOLOv2, YOLOv3, and YOLOv4 models.

In Table 2, using the COCO test dataset, the YOLOv4 system accomplishes a high score of 95.07 % in mAP than the other models. The AP value of the bus is 98.11%, which is less than YOLOv3 that scores 99.01 %, but YOLOv4 is efficient enough for evaluating the objects, and the value of the truck is 92.08 % which is good.

Table 2. Results on the test set COCO dataset using the Faster R-CNN, YOLOv2, YOLOv3, YOLOv4 method.

Methods	mAP	Bus	Truck
Faster R-CNN	85.75	87.11	80.54
YOLOv2	79.01	80.64	81.64
YOLOv3	94.73	**99.01**	90.45
YOLOv4	**95.07**	**98.11**	**92.08**

In Table 3, adopting the PASCAL VOC2007 test set, the YOLOv4 model scores 94.33 % in mAP with an AP value for bus 97.33 %, which is less than YOLOv3 98.86 %, and the value for truck is 92.08 %. From Fig. 3, YOLOv4 achieves higher accuracy of 96.54 % compared to other models.

Table 3. Results on the test set PASCAL VOC 2007 dataset using the Faster R-CNN, YOLOv2, YOLOv3, YOLOv4 method.

Methods	mAP	Bus	Truck
Faster R-CNN	84.68	86.54	80.14
YOLOv2	78.54	76.14	80.24
YOLOv3	93.82	**98.86**	88.77
YOLOv4	**94.33**	**97.75**	**90.91**

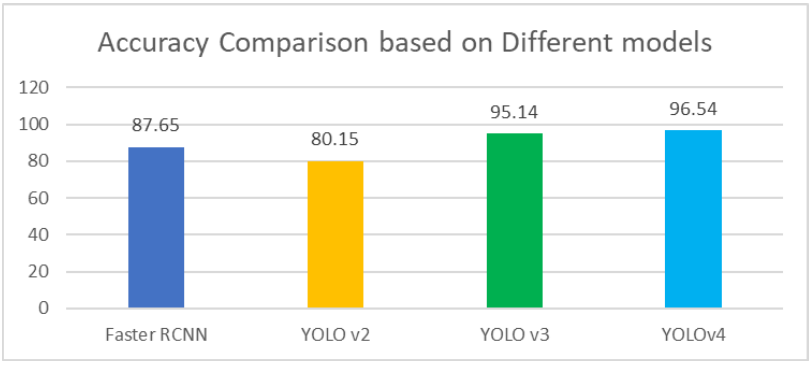


Fig. 3. Comparison of accuracy Faster R-CNN, YOLOv2, YOLOv3, and YOLOv4 models.

Table 4 compares the mAP with other state-of-approaches. Our proposed model achieves a Faster-RCNN of 87.65 %, YOLOv2 of 80.15 %, YOLOv3 of 95.14 %, and YOLOv4 of 96.54 %. The proposed system YOLOv4 algorithm is comparatively superior to other vehicle detection models.

Table 4. Comparisons results on vehicle detection models with others.

Models	Overall (mAP) (%)
YOLO [11]	70.58
YOLOv3 [12]	84.96
Refined YOLOv4 [13]	67.7
CNN [14]	75.7
DP-SSD [15]	75.43
Our proposed model:	
YOLOv4	96.54
YOLOv3	95.14
YOLOv2	80.15
Faster R-CNN	87.65

The proposed algorithm was tested on various live traffic densities in Chennai, as shown in Fig. 4. The pictures depict heavy vehicle detection using various proposed models; (a)(b) represents Faster R-CNN, (c) (d) shows YOLOv2, (e)(f) pictures YOLOv3, (g)(h) detailsYOLOv4 (day-time), and (i)(j) YOLOv4 renders (night-time). The YOLOv4 is most efficient in night-time detection than other models.

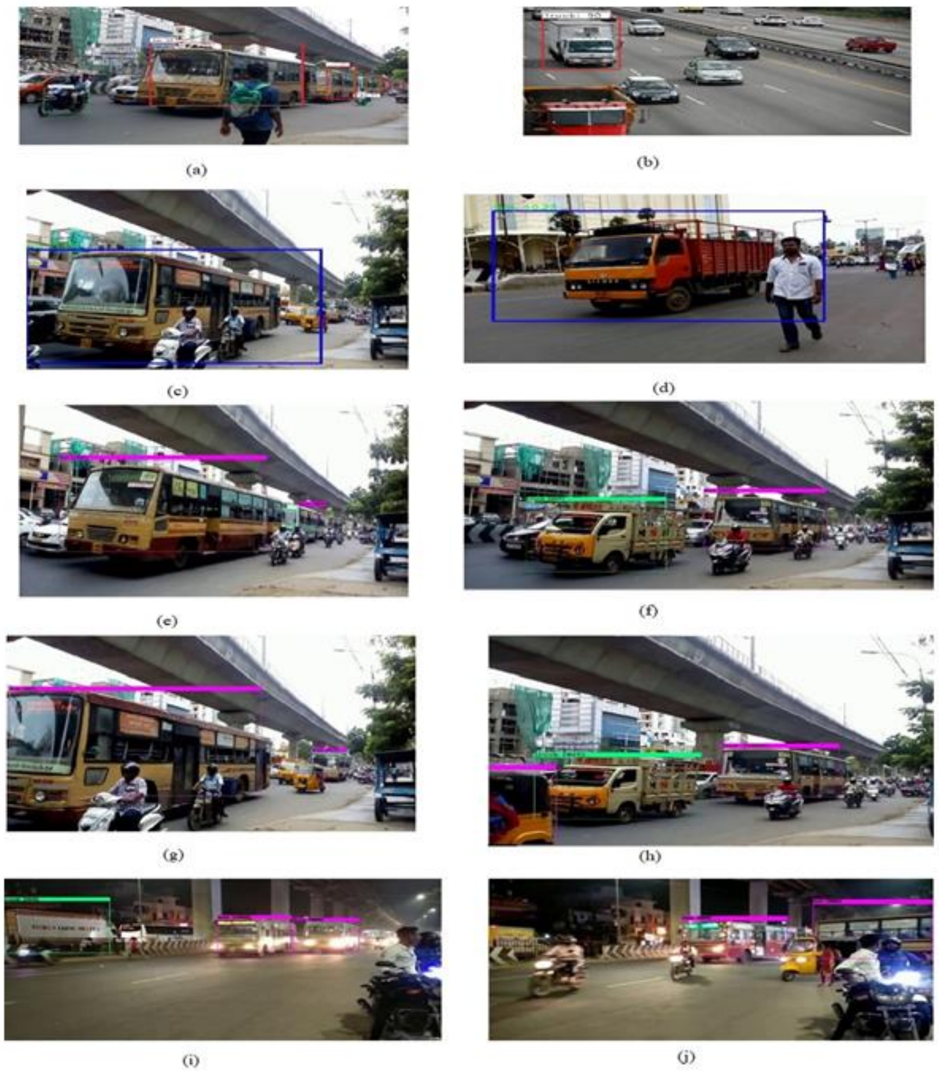


Fig. 4. Heavy vehicle detection using proposed model (a) (b) Faster R-CNN (c)(d) YOLOv2 (e)(f) YOLOv3 (g)(h) YOLOv4 (day-time) (i)(j) YOLOv4 (night-time).

The above tabulation details that YOLOv4 is better than the other models with low and higher mAP. To achieve detection speed, the system relies on the ratio of mAP and precision. Faster R-CNN is not suitable to be used in real-time applications and autonomous vehicle environments. The limitation of the YOLOv2 is that it is less efficient in the detection of small vehicles and YOLOv3 performs poorly in the detection of occluded vehicles. YOLOv4 has satisfactory mAP along with the appropriate setting of high FPS and low loU is suitable for real-time applications, thereby evident that it is the best model of its class.

5. Conclusion

In this paper, a comparison of deep algorithms on various features is detailed, and the techniques include Faster R-CNN, YOLOv2, YOLOv3, and YOLOv4. This proposed system compares the advanced deep learning-based object detection algorithms adopting the COCO and Pascal VOC benchmark datasets for testing the models. The quick view of the comparative analysis reveals YOLOv4 is the fastest compared to other versions of YOLO - V2 and V3, and Faster R-CNN has the least sub-optimal outcomes. Even though the Faster R-CNN algorithm produces the most accurate results in terms of accuracy, the major drawback is the speed. The YOLOv2 and YOLOv3 systems find it difficult to detect heavy vehicles, especially at night-time, but the YOLOV4 algorithm exhibits exceptional capability in the detection of vehicles of heavy class. YOLOV4 provides a reasonable balance between both speed and accuracy. Besides, the YOLO framework's implementation improves recognition accuracy and reduces computational costs. By integrating transfer learning, the system achieves a significant improvement in accuracy rate. Considerable experiments have been conducted, achieving favorable outcomes. In the future, The YOLOv4 approach shall be studied, tested, and implemented for real-time vehicle detection in different climatic conditions.

References

1. A. Sohail, M. Jayabalan, and V. Thiruchelvam, J. Crit. Rev. **7**, 786 (2020).
2. P. Kumar, V. Garg, P. Somvanshi, and C. Pathanjali. Int. J. Eng. Res. Technol. **8**, 438 (2019).
3. M. Chate and V. Gohokar, Heavy Vehicle Detection Using Fine-Tuned Deep Learning, ed. D. Pandian et al. – *Proc. of the Int. Conf. on ISMAC in Computational Vision and Bio-Engineering 2018* (ISMAC-CVB), (2019). https://doi.org/10.1007/978-3-030-00665-5_175
4. T. Kawakatsu, K. Aihara, A. Takasu, and J. Adachi, Fully-Neural Approach to Heavy Vehicle Detection on Bridges Using a Single Strain Sensor - *2020 IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)* (2020) pp. 3047. https://doi.org/10.1109/ICASSP40776.2020.9053137
5. A. Uçar, Y. Demir, and C. Guzelis, Simulation **93**, 769 (2017). https://doi.org/10.1177/0037549717709932
6. R. Girshick, Fast R-CNN - *IEEE Int. Conf.* Computer Vision (2015) pp. 1440–1448. https://doi.org/10.1109/ICCV.2015.169
7. J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, You Only Look Once: Unified, Real-time Object Detection - *IEEE Conf. on Computer Vision and Pattern Recognition* (2016) pp. 779–788. https://doi.org/10.1109/CVPR.2016.91
8. Bochkovskiy, A. Wang, C. Y. Liao, and Hong-yuan, YOLOv4: Optimal Speed and Accuracy of Object Detection (2020). https://arxiv.org/pdf/2004.10934v1.pdf.
9. N. Tsoi, J. Gwak, I. Reid, and U. States, Generalized Intersection over Union: A Metric and A Loss for Bounding Box Regression (2019). N. Tsoi, J. Gwak, I. Reid, and U. States, Generalized Intersection over Union: A Metric and A Loss for Bounding Box Regression (2019). arXiv:1902.09630v1 [cs.CV] 25 Feb 2019 - IVSN
10. T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ra- mannan, P. Dolla'r, and C. L. Zitnick. Microsoft COCO: Common Objects in Context - *Eur. Conf. on Computer Vision*, (Springer 2014) pp. 740-755.
11. H. Wang, X. Lou, Y. Cai, Y. Li, and L. Chen, J. Sensors **2019**, ID 8473980 (2019). https://doi.org/10.1155/2019/8473980

12. K. Kim, P. Kim, Y. Chung, and D. Choi, Performance Enhancement of YOLOv3 by Adding Prediction Layers with Spatial Pyramid Pooling for Vehicle Detection – *15th IEEE Int. Conf. on Advanced Video and Signal Based Surveillance (AVSS)* (2018) pp. 1-6. https://doi.org/10.1109/AVSS.2018.8639438
13. P. Mahto, P. Garg, P. Seth, and J. Panda, Int. J. Adv. Res. Eng. Technol. **11**, 409 (2020).
14. L. Chen, F. Ye, Y. Ruan, H. Fan, and Q. Chen, EURASIP J. Image Video Processing **2018**, 109 (2018). https://doi.org/10.1186/s13640-018-0350-2
15. F. Zhang, C. Li, and F. Yang, Sensors **19**, 594 (2019). https://doi.org/10.3390/s19030594