Journal of Real-Time Image Processing (2023) 20:93
https://doi.org/10.1007/s11554-023-01344-1

RESEARCH

Faster RCNN based robust vehicle detection algorithm for identifying and classifying vehicles

Md Khorshed Alam[1] · Asif Ahmed[2] · Rania Salih[3] · Abdullah Faiz Saeed Al Asmari[4] · Mohammad Arsalan Khan[5,6] · Noman Mustafa[7] · Mohammad Mursaleen[8] · Saiful Islam[4]

Received: 12 April 2023 / Accepted: 7 July 2023 / Published online: 30 July 2023

Abstract

Deep convolutional neural networks (CNNs) have shown tremendous success in the detection of objects and vehicles in recent years. However, when using CNNs to identify real-time vehicle detection in a moving context remains difficult. Many obscured and truncated cars, as well as huge vehicle scale fluctuations in traffic photos, provide these issues. To improve the performance of detection findings, we used multiscale feature maps from CNN or input pictures with numerous resolutions to adapt the base network to match different scales. This research presents an enhanced framework depending on Faster R-CNN for rapid vehicle recognition which presents better accuracy and fast processing time. Research results on our custom dataset indicate that our recommended methodology performed better in terms of detection efficiency and processing time, especially in comparison to the earlier age of Faster R-CNN models.

Keywords Classification · Deep learning · Modified vgg16 · Vehicle detection

1 Introduction

Technologies of vehicle detection have existed under development in industry and academics in recent years. Many state-of-art image identification algorithms have not been able to compete in the field of vehicle detection standards. The primary obstacles in automobile identification include big differences in object sizes, substantial occlusion, and considerable fluctuations in light. Sensor-based algorithms may be used to solve some surveillance tasks in urban traffic systems, such as vehicle counting [1], license plate identification [2], incident detection [3], driver facial emotion identification [4–7],and internet of things (IOT) source location and identification [8]. On the other hand, vision-based approaches can fully use the abundance of visual patterns to distinguish target objects in a human-like manner. For example, radar sensor-based methods can only identify cars in a relatively small area, but vision-based systems may use a camera to discover all the vehicles in a vast viewable region and describe additional aspects of each detected vehicle at the same time. As a result, numerous computer vision and machine learning

✉ Rania Salih
rania.salih2@rsu.edu.sd

✉ Mohammad Arsalan Khan
arsalan.khan@ifg.uni-kiel.de

✉ Mohammad Mursaleen
mursaleenm@gmail.com

[1] School of Automotive Studies, Tongji University, Shanghai, China

[2] Department of Geotechnical Engineering, College of Civil Engineering, Tongji University, 1239 Siping Road, Shanghai 200092, China

[3] Department of Civil Engineering, Red Sea University, Port-Sudan, Sudan

[4] Civil Engineering Department, College of Engineering, King Khalid University, Abha 61421, Saudi Arabia

[5] Primary Affiliation: Geomechanics and Geotechnics Group, Kiel University, 24118 Kiel, Germany

[6] Department of Civil Engineering, Z. H. College of Engineering and Technology, Aligarh Muslim University, Uttar Pradesh, Aligarh, India

[7] School of Electronic Information and Electrical Engineering, Shanghai Jiao Tong University, 800 Dong Chuan Road, Shanghai 200240, China

[8] China Medical University Hospital, China Medical University (Taiwan), Taichung 40402, Taiwan

models are thoroughly investigated in this study in order to solve a range of fascinating challenges in intelligent transportation systems. Researchers have suggested various classic vehicle detection algorithms from the earliest stages of the field to current days [9–15]. The performance of techniques is determined by handcrafted characteristics. The most often utilized features are the Haar-like [16] and Histogram of Oriented Gradient (HOG) [17]. The cascaded detector [18], exhibiting a commendable level of precision, stands out as one of the pioneering real-time detection systems. Two well-known methods of the part-based design method are Support Vector Machines (SVM) [19] and deformable part-based models (DPM) [20]. The researchers focus on three major practical issues in vehicle detection such as huge variations in light, heavy occlusion, and big variations in sizes. To solve the issue of high variance in light, Saini [21] presented a strong CNN model for traffic management light recognition for autonomous cars. As input, the framework uses raw picture data, identifies candidate regions, and later detects and recognizes traffic lights. Heavy occlusion makes distinguishing occluded vehicles hard to detect. Phan [22] presented a strategy for dealing with thick occlusion caused by surveillance cameras that are fixed. The approach includes background removal, occlusion detection, and automobile detection, which extracts occluded cars separately based on exterior attributes. For big variations of size, Lu [23] has suggested a scale-aware Region Proposal Network (RPN) to handle the challenge of identifying vehicles of various sizes. The scale-aware RPN is composed consisting of two particular sub-networks: one that detects big proposals and the other detects little proposals, which are then fed into two different XGBoost [24] classifiers to create the final prediction.

The first appropriate technique, known as region-based convolutional neural network (R-CNN) [25], performed well in vehicle identification. The region-based convolution neural network has a region proposal network with the CNN to outperform HOG [17] features with an SVM classifier. Raw picture data is fed into a region-based convolutional neural network, which generates region recommendations. The region suggestions are then put into the CNN to extract the features map, and the support vector machine [26] is used to forecast. In the Pascal VOC 2010 competition, the basic RCNN obtained a mean average accuracy of 53% and spatial Pyramid Pooling (SPP) [27] employs a convolution layer on the whole picture and extracts the features map using SPP-net, avoiding the high cost of computation of R-CNN.

In 2015, He et al. suggested a faster R-CNN [28] for object recognition. Faster R-CNN was the primary to use the Region Proposal Network (RPN) as a candidate generator for Regions of Interest (Roi). For the COCO [29] and Pascal VOC [30] two-dimensional object detection standards, the faster R-CNN performs well. One recent work has performed well, with a mean average accuracy of 83.92 percent on the KITTI [31] automobile detection standard. Another recent work the use of a faster R-CNN with domain adaption in road vehicles detection research shows about accuracy 83.1% and 0.56 s testing time on COCO dataset [32].

Faster R-CNN's competitive performance on the KITTI vehicle identification benchmark may be explained by one primary factor that’s the wide range of vehicle scales. The RPN is fed convolutional feature maps and produces possible ROI. RPN ignores tiny objects and vehicle overlooking due to the wide range of vehicle scales. However, we think there is a scope to further improve the faster R-CNN performance, therefore we propose a model to solve the issue of wide-scale variance in vehicle detection. Not only for adequate transportation management or administration but also for efficient damages detection in insurance solutions, accurate classification of automobiles into distinct kinds is critical. Therefore, a work towards automatic damage assessment procedure on vehicles is essential to prevent work accidents that may be caused by individuals while assessing the damage.

2 Custom dataset creation

2.1 Preparing dataset

This study has created a unique and customized dataset as well as KITTI Vision Benchmark Suite for training and evaluation purposes. Every subcategory has at least 66,000 illustrations in our collection. Figure 1 show some samples of our custom dataset. Our dataset was gathered from the different roadside and upper sides of a different roads. We installed the camera at a specific place and recorded videos at 60 frames per second at a different place for three days. The images were then retrieved from videos and duplicate pictures were eliminated. We categorized data following gathering it according to its classifications. The five categories in our database are car, bus, truck, motorcycle, and cycle. We offered images of every category from several perspectives, including panoramic, front base, and lateral views. There are 4000 samples capture in this study and more than 2000 were taken from the KITTI dataset. altogether we got 6000 images and for our training purpose, we used 4000 images and for validation purposes 2000 images were used. The dataset is graded on three difficulty levels: easy, moderate, and hard. The easy type objects are made up of anchor boxes with the least height of 40 pixels and higher, the moderate type objects are made up of anchor boxes with the height of 25 pixels to 40 pixels, and the hard type images are made up of anchor boxes with a Smaller than 25 pixels considered as hard.

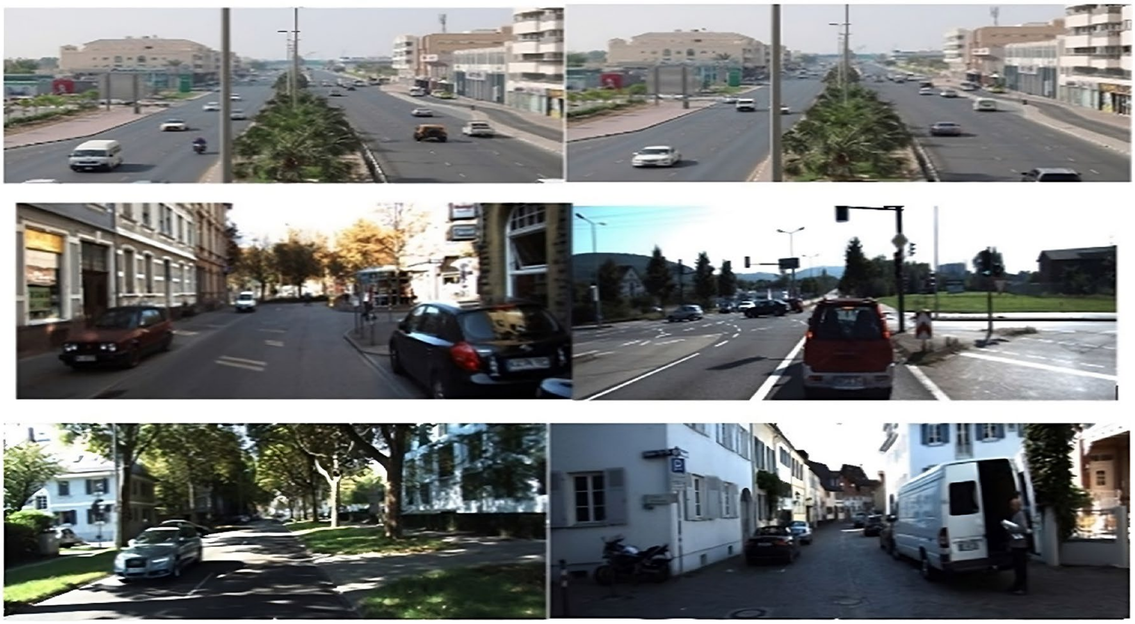

Fig. 1 Few examples of our custom dataset

2.2 Annotating dataset

The key difficulties in machine learning involve object detection and categorization. The detection and classification methods help to identify numerous items on the street, including automobiles, humans, and fixed things like traffic lights, street signs, and lamp posts. Real-time training datasets are required for the creation of identification and classification methods. However, we created our own dataset with in different streets with a different expression. The images in these datasets are typically manually labeled by us. we build bounding boundaries around the recognized items and save the features of such objects through an annotation stage like most of the researchers use some open-source software to do the annotation process manually[33].

Utilizing the tools, we generate shapes for image segmentation, build anchor boxes in object detection and recognition, and add captions to the selected regions.

The annotation information is saved in a variety of forms, including text, JSON, YOLO, XML, ILSVRC, and others. In our case, we saved the annotation information as XML. The hand annotation process is very costly and also time-consuming. For example, the YOLO object detection database requires approximately 35 s to build an anchor box over an object [28]. Experts use two alternative ways to make the operation of bounding box annotations affordable and efficient. There are three types of annotation processes: manual, semi-automated, and completely automatic that we can use. A manual annotation tool had been employed for manually labeling the images. This tool allows us to label using a wide variety of labeling techniques. To date, the tool has been used to annotate over 6000 photos that are only in the training dataset which has over 66,000 automobile illustrations. The labeling method that we used is the bounding box method. The below Figure shows some samples from our annotation image and annotation XML file information. Our annotation XML files contain the information as different vehicle types like the car, bus/truck, cycle/motorbike, and the difficulty level are easy, moderate, and hard.

3 Methodology

3.1 *The implemented architecture*

At the beginning of our method, we take our custom annotated dataset as import Then the imported images go through the base network. In the base network we used different kinds of architecture, those are our modified Vgg16, ResNet50, ResNet101, and MobileNetV3.The output from the base network (base network feature map) fed to RPN, soft NMS, as a result, we got our proposal layer (with anchor box). Roi pooling layer takes the input from the base network and proposal to perform max polling with transposed convolution and gives the output as the refined proposal. Finally, the output of the refined proposal goes through the classification and regression layer to show the final detection results with the regression box and classification box. The suggested approach's general architecture in our research is illustrated in Fig. 2.

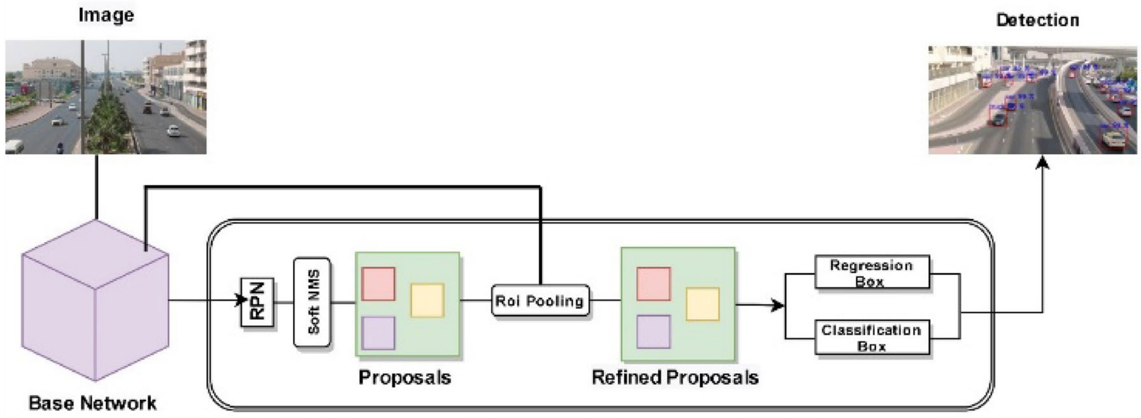

Fig. 2 Proposed conventional faster-RCNN model

3.2 Modified VGG16

As the backbone framework, we have used the modified VGG 16. It has the large-sized Kernel Filtering, with 11 in the first Convolutional Layer, each with 3×3 numerous kernel-sized filters. It is preferable to transmit 224×224×3 to Conv1 layers, after which the images go through numerous convolutional layers with very narrow filtering of size 3×3. A 1×1 convolutional filter is also employed in some circumstances. After every block data goes to the dropout layer and then batch normalization. Very small size kernels are layered together around the perception to retain the spatially numerously small shaped kernels layered together on the receptive field since it assists to understand complicated features at low cost as numerously nonlinear-layers boost the depth of the network. The padding is set appropriately (for the 3×3 Convolutional layer, padding is set to 1×1 pixels), however, the strides are set to 1. The spatial Pooling is performed by the five Max Pooling stages; this comes after a few convolutional layers but is not complete of them. Max Pooling makes use of the 2×2 Pixels Kernel or the window with the 2 strides. At the last instant of max pooling, we added a Global Average Pooling (GAP) layer before fully connecting FC-12 and the SoftMax layer.

We mainly have done modifications on VGG-16 and named it as modified VGG-16. After every max-pooling stage of Traditional VGG16, we add a dropout and a batch normalization layer. The goal of Batch Normalization is to achieve a stable distribution of activation values throughout training, and in our experiments, we apply it before the activation layer. BN layer performs scaling operation on the outputs of the layer before it. This process brings stability to the weights updating during the training of the model. This has the effect of stabilizing and speeding-up the training process of deep neural networks. In result, the network weights optimization becomes simplified and the network achieves loss convergence in lesser time. At the place of last max pooling, we added a Global Average Pooling (GAP) layer before fully connecting FC-12 and the SoftMax layer. Traditional VGG16 doesn't have a dropout and batch normalization layer in it. With our modified VGG-16, the weight that has been pre-trained will be benefited. After adding dropout, batch normalization layers and using global average pooling, the modified VGG16 networks exhibit excellent classification performance and shorter testing time. On the other hand, the other three base network (MobileNetV3, ResNet-101and. ResNet-50) that we have also tested, those were unchanged and fine-tuned for our model which didn't perform well like modified Vgg16.

From Fig. 3, we can see that the RPN creates a collection of anchor boxes from the base network's convolution feature map. Those anchors frequently overlap, and proposals frequently overlap over the identical object. To solve the problem of overwriting proposals, the soft non-maximum suppression (SNMS) algorithm is used. The NMS algorithm is often used to erase duplicate proposals for many state-of-the-art object recognition techniques, including Faster R-CNN. Classical NMS eliminates any other proposal that overlaps a winning proposal by over a predetermined threshold. The classical NMS algorithm could remove beneficial proposals surprisingly caused by heavy automotive occlusion in traffic (Fig. 4).

This study used a soft-NMS algorithm to solve the NMS problem with overlapped vehicles. The neighboring proposals of successful proposals really aren't completely effectively suppressed with soft-NMS. Rather, those are suppressed relying on the neighboring proposals' updated objectiveness scores, which are calculated depending on the level of overlap between the neighboring proposals or the winning proposal.

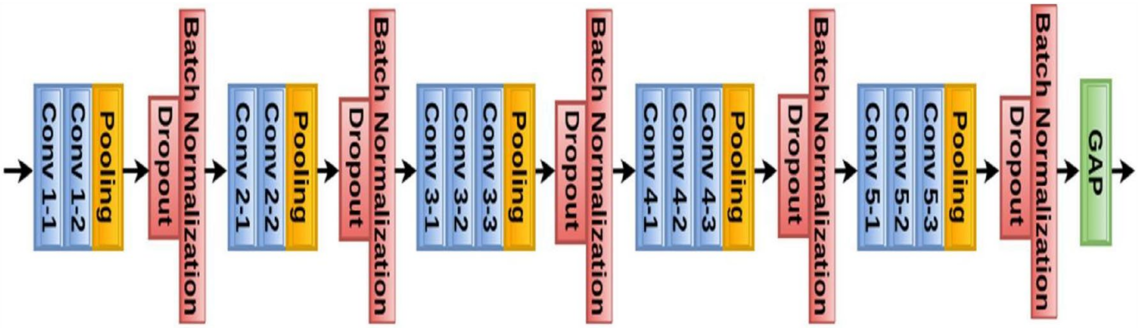

Fig. 3 Our the proposed VGG16 illustrate

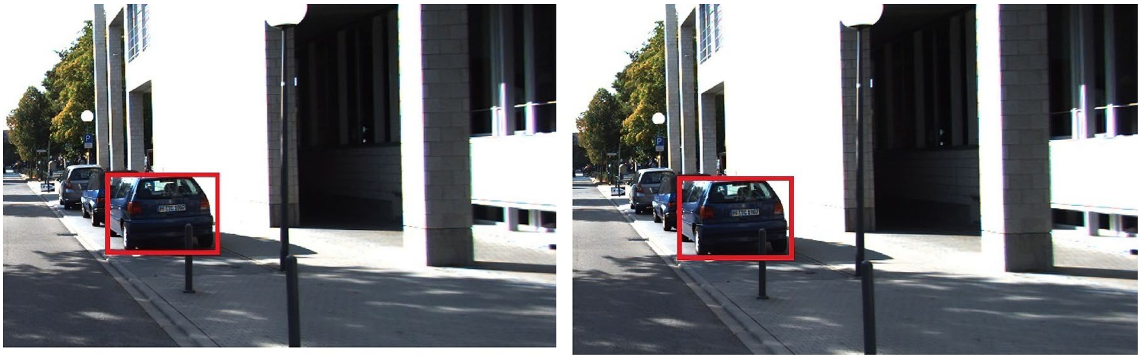

Fig. 4 Illustration result of SNMS

3.3 Refined proposal

The Roi pooling layer is often used in various two-stage object recognition methods, including Faster R-CNN, and Fast R-CNN [23], to regulate reducing the size of proposals to a certain size. The Roi pooling layer employs the max pooling. that turn the features within each acceptable region of interest from the proposal layer into a compact feature map with such a specific geographic area H × W. The h × w Roi proposal is divided into an H × W matrix of sub-windows of similar sizes of (h/H) × (w/W), and the elements in every sub-window are max-pooled into the appropriate output bounding box.

If a proposal size is smaller compared to H × W, that will be extended to accommodate the extra space by adding repeated values. Because Roi pooling avoids processing the convolutional layers again, it may drastically reduce both training and testing time. Adding repeated values to tiny proposals, on the other hand, is not acceptable, particularly with little vehicles, since it may ruin the actual shapes of the small automobiles.

Furthermore, adding duplicated values for minor proposals would result in erroneous forward propagating an error buildup in backward propagation throughout the training phase. The detection of tiny cars will be decreased. To reduce the size of suggestions to a fixed size while preserving the original features of tiny cars and improving the efficiency of the suggested technique for identifying small vehicles by the refined proposal.

Figure 5 depicts the refined proposal idea. When the size of the proposal is bigger than the specified feature size map output in the refined proposal process, max-pooling is employed to decrease the proposal's size to a specific size.

When the proposal size is lower than the output feature map's predetermined size, Transposed Convolution is used to extend the proposal size to the fixed size. The proportion between the refined proposal size and the input proposal size determines the kernel size. Moreover, while the wideness of a proposal is greater than the stable output feature map's size and the proposal height is lower than the specified height of the outputs feature, transposed convolution is used to increase the proposal height while max-pooling used to decrease the width of that proposal. The proposal size has been regulated to a fixed size with improved proposal.

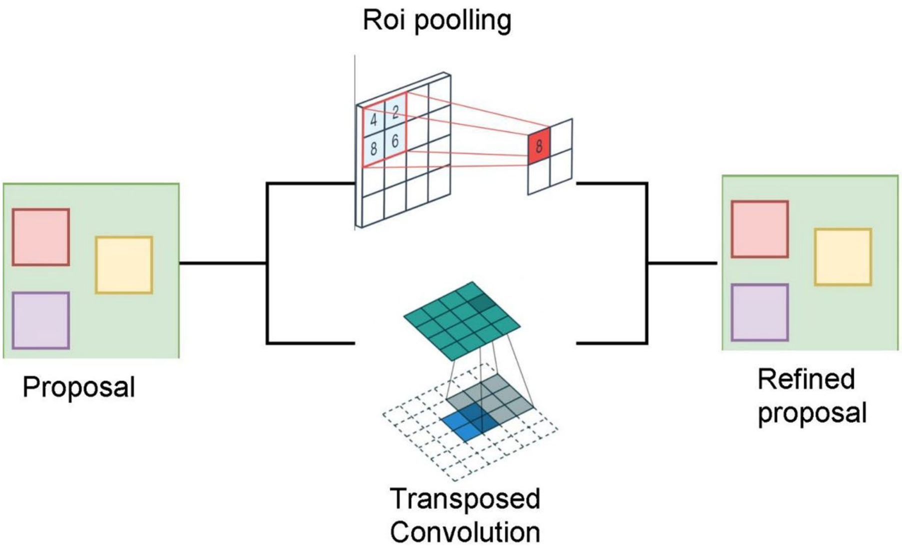

Fig. 5 Refined proposal

4 Result and analysis

In this research, we primarily apply the modified Vgg-16, MobileNetV3, ResNet-101, and ResNet-50 model to our custom-made dataset, which is then fine-tuned on the KITTI dataset for the base network. The training environment utilized for our experiments involved the utilization of the Nvidia RTX3080 GPU. This high-performance GPU from Nvidia played a crucial role in accelerating the training process and enabling efficient model optimization. Its advanced capabilities provided the necessary computational power to handle the complex training tasks and achieve optimal results. Each batch normalization layer's weights and dropout in the pre-trained model were increased to speed up training and reduce overfitting. Turn after turn, the classifier and the RPN are trained. A mini-batch is used to train the RPN initially, with the base network and RPN variables changed just once. The RPN's negative and positive proposals are then used to update and train the classification. The classifier parameters are adjusted once, then the characteristics of the basic convolutional layers are adjusted once more. RPN and the modified Vgg-16, MobileNetV3, ResNet-101, and ResNet-50-based classifier both use the same underlying convolutional layers. The loss function for bounding box regression and coordinate parameterization are similar to the traditional Faster R-CNN work. In the loss function, the balance parameter is set to 1. The loss functions are optimized using the SGD with momentum. With the learning rate per mini-batch set at 0.0001, the RPN and the classifier's starting learning rates are predetermined to 0.0001 and we used 200 epochs for training.

We can see in Fig. 6 that at some time, the training and validation losses both diminish and stabilize. This demonstrates that our model's ideal fit does not underfeed or overfeed the data.

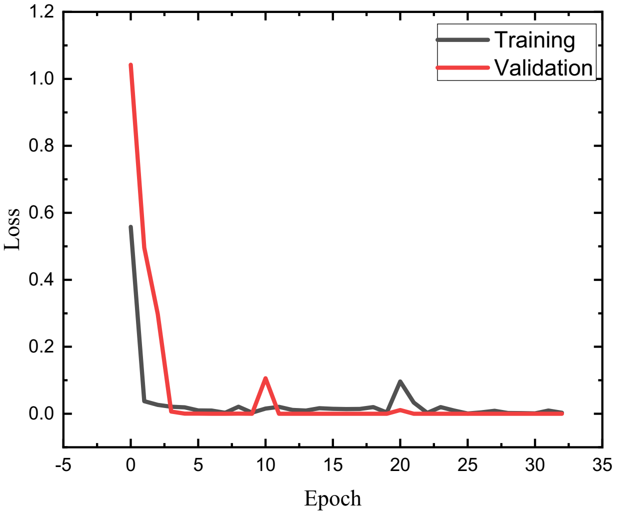

Fig. 6 Training and validation losses

In Fig. 7 The training accuracy and validation accuracy both stabilize at a specific point which means we got a well-trained network with modified VGG16 as a base network.

Table 1 shows some of the outcomes of our modified model. We also discovered that the traditional Faster R-CNN fails to detect little objects (less than 64 pixels). As a result, we proposed Modified VGG16 with soft NMS and a refined proposal to accommodate tiny objects. Table 1 reports the analysis of the traditional Faster R-CNN and our improved model comparison.

Table 1 displays the results that our proposed model gives better MAP and processing time performance than the older version of Faster R-CNN. Research results indicate that our recommended methodology performed better in terms of detection efficiency and processing time, especially in comparison to the traditional Faster R-CNN models.

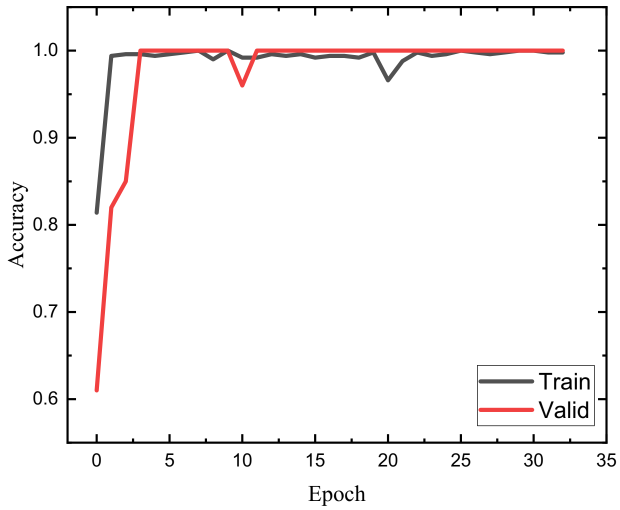

Fig. 7 Training and validation accuracy

As we can see from Table 1, our modified Vgg16 and MobileNetV3 provide a better MAP accuracy and our model can detect different categories of vehicle (car as medium size vehicle, bus/truck big size vehicle, and cycle Motorbike as small size vehicle). Therefore, we decided to choose modified Vgg16 as a base network for our final comparison with those recent publication techniques using the KITTI dataset. We can see the difference of improvement in Table 2.

As we can see from Table 2 our modified Vgg16 and MobileNetV3 give a better map and our model can detect different categories of vehicle (car as medium size vehicle, bus/truck big size vehicle, and cycle/ Motorbike as small size vehicle) So we decided to choose modified Vgg16 as a base network for our final comparison with those recent publication techniques using the KITTI dataset. Later, Fig. 8 has shown the results of Precision-Recall for three different categories (car, bus/truck, and cycle/ Motorbike) average precision (AP) measures given by our model with modified VGG16 in terms of easy, medium, and hard level.

Table 2 Results of the different base networks on our proposed model

Base Network	Learning Rate	mAP	Car	Truck/ Bus	Motor-bike/ Cyclist
Vgg16	0.0001	88.35	90.25	87.43	87.37
Modified Vgg16	0.0001	91.78	93.67	89.15	92.52
MobileNetV3	0.0001	89.52	92.41	90.66	85.49
ResNet50	0.0001	72.61	75.42	72.25	70.14
ResNet101	0.0001	75.99	78.14	75.39	74.45

On our custom dataset, the suggested model with modified VGG16 has 93.67% for cars, 89.15% for truck/ buses, and 92.52% for motorbikes/cycle Global Accuracy, indicating that it performs well on different hardness level by pixel of the bounding box.

Our model obtains 91.78% AP on a different degree of difficulty with a duration of 0.11 s per picture by utilizing a GPU with 11 GB of RAM.

Figure 9 can show some prediction accuracy in dataset detection results. Each layer is divided into proposed areas by our modified VGG16, which also predicts the locations of many anchor boxes of various scales and sizes for each object. The predicted box placement is adjusted using a global optimization, and we can see that our model predicted accurately although some images have poor lighting condition and vehicle is located in the shadow area of the road.

We also discovered that the traditional Faster R-CNN fails to detect little objects (less than 64 pixels). As a result, we proposed Modified VGG16 with soft NMS and a refined proposal to accommodate tiny objects.

We tried four different kinds of base networks as feature extractors (Modified Vgg16, MobilenetV3, ResNet50, and ResNet101). Using our modified model, we were able to recognize the automobile category in our custom detection dataset. In traditional faster R-CNN they used the classical VGG16 as a feature extractor but in our model we used modified VGG16 which give better accuracy and faster testing time. the soft-NMS method replaces the NMS (non-maximum suppression) method after the RPN (region proposal network) in the traditional Faster R-CNN to tackle the issue of duplicated proposals and it also slightly improve the AP performance. The proposals are then adjusted to the appropriate size using a refined

Table 1 Our proposed model result improvement after implementing different steps

Method	Easy Object mAP	Moderate Object mAP	hard Object mAP	Process-ing time (s/ Image)
Faster R-CNN [1]	86.71	81.84	71.12	2
Soft NMS	88.43	86.78	76.31	2
Refined proposal	89.59	91.39	81.24	2
Modified Vgg16	87.27	81.84	73.21	0.11
MobilenetV3	85.42	78.04	69.64	0.13

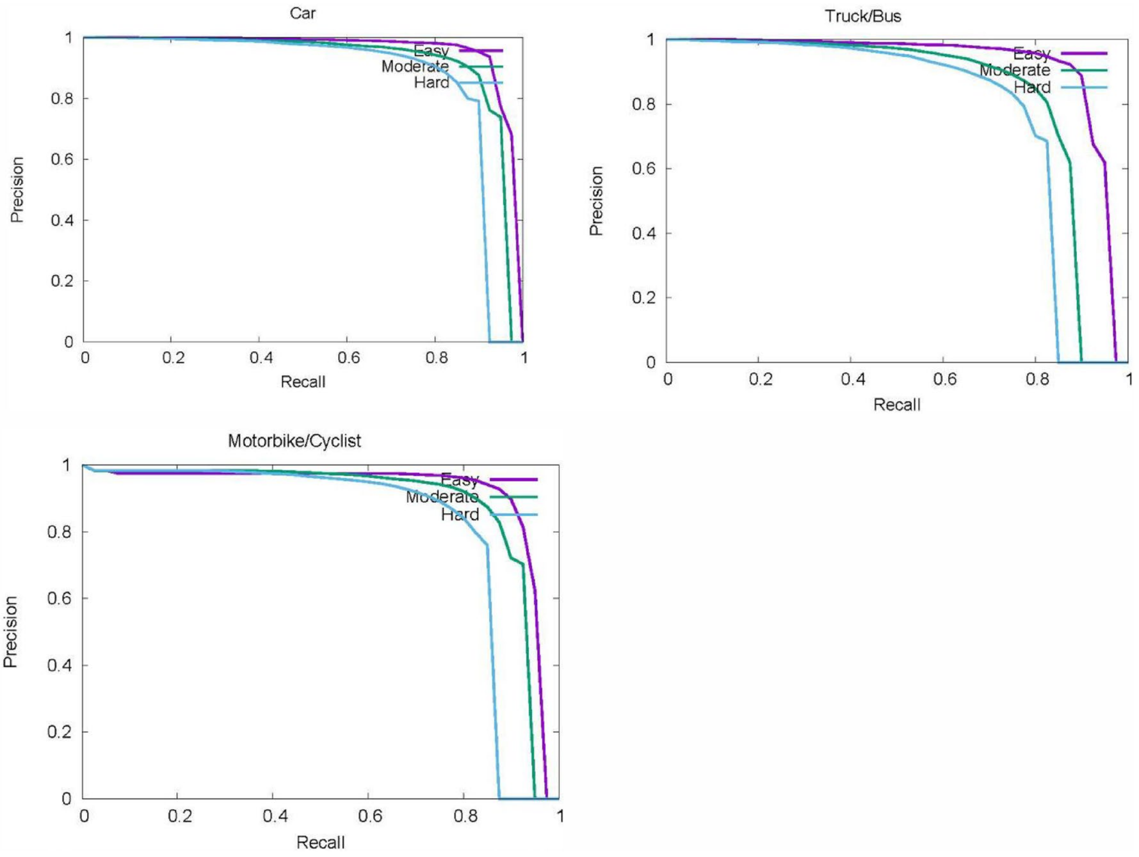

Fig. 8 Precision-recall categories with modified VGG16 model on easy, medium and hard levels

proposal layer without compromising vital contextual information which gives better performance to detect tiny sized vehicle than the traditional faster R-CNN.

We evaluated our model with the custom dataset for state-of-the-art detection on the KITTI testing dataset. We chose to select a slightly unique training dataset that we were using for assessment on our custom dataset since the KITTI testing dataset had comparable scenarios to the training set. Since we often encounter situations in the testing set where cars appear stranded on the street, we decided to include heavily occluded vehicles in our analysis. As a result, a dataset containing all occluded labels were utilized to train the network that was used to submit to the scoreboard. Aside from that, all of the training settings were identical. Table 3 presents the performance of our recommended approach on the custom dataset, as well as the results of other approaches on the KITTI test dataset. It provides a comprehensive overview of our position in the KITTI benchmark, showcasing the effectiveness and competitiveness of our proposed method compared to existing approaches.

Our proposed model gives better mAP and processing time performance than the older version of Faster R-CNN.

5 Conclusion and future work

The purpose of this research is also to use deep learning to get a better understanding of real-time road vehicles, including preparing our own dataset with image annotation and vehicle recognition. Tuning the number and density of the network's convolutional layers demonstrates the neural network and data flexibility. We chose our modified VGG16 as the core base network model for the feature extractor after assessing all of the evaluation indicators in general. In future work, the major component that needs to be focused on is a range of photograph collections, such as lighting settings and background surroundings. CNN models can readily notice patterns and output with a greater accuracy rate when given various input components. In addition, the volume of the dataset can play a role in learning algorithms.

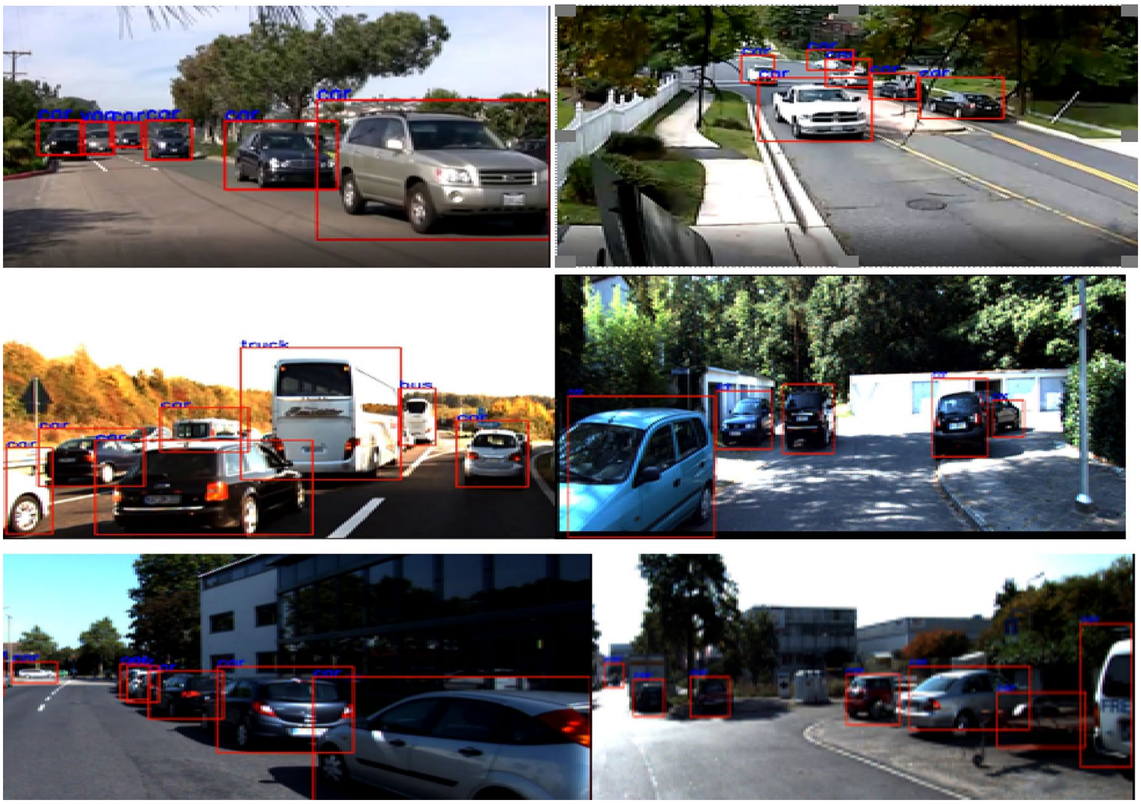

Fig. 9 Visual representation of detection results on test datasets samples

Table 3 Performance comparison on KITTI benchmark

Method	Easy Object mAP	Moderate Object mAP	hard Object mAP	Processing time (s/Image)
Faster R-CNN [10]	86.71	81.84	71.12	2
Faster R-CNN [52]	89.20	87.86	74.72	0.15
Complexer-YOLO[53]	79.43	71.97	67.62	0.06
IA-SSD (single)[54]	83.98	76.37	71.73	0.013
Cascade MS-CNN[55]	94.26	91.60	78.84	0.25
Proposed Method (with original VGG16)	88.35	86.87	77.68	0.14
Proposed Method (with modified Vgg16)	91.78	89.54	79.54	0.11

Author contributions Alam and Ahmed wrote the main manuscript, Alam and Asmari visualised, Salih did data curation, Khan did the re-writing, Mustafa, Mursaleen and Islam did the formal analyses. All authors reviewed the manuscript.

Funding Open Access funding enabled and organized by Projekt DEAL.

Declarations

Conflict of interest The authors declare that there is no conflict of interest regarding the publication of this paper.

Data Availability Statement The data is available as per request to corresponding author.

References

1. Bas, E., A.M. Tekalp, and F.S. Salman. *Automatic vehicle counting from video for traffic flow analysis*. in *2007 IEEE intelligent vehicles symposium*. 2007. Ieee.
2. Chen, R.-C.: Automatic License Plate Recognition via sliding-window darknet-YOLO deep learning. Image Vis. Comput. **87**, 47–56 (2019)
3. Hussain, T., et al.: Real time violence detection in surveillance videos using Convolutional Neural Networks. Multimedia Tools and Applications **81**(26), 38151–38173 (2022)
4. Zaman, K., et al.: Driver Emotions Recognition Based on Improved Faster R-CNN and Neural Architectural Search Network. Symmetry **14**(4), 687 (2022)
5. Shah, S.M., et al.: A driver gaze estimation method based on deep learning. Sensors **22**(10), 3959 (2022)
6. Ullah, R., et al.: Auction Mechanism-Based Sectored Fractional Frequency Reuse for Irregular Geometry Multicellular Networks. Electronics **11**(15), 2281 (2022)
7. Zaman, K., et al.: EEDLABA: Energy-Efficient Distance-and Link-Aware Body Area Routing Protocol Based on Clustering Mechanism for Wireless Body Sensor Network. Appl. Sci. **13**(4), 2190 (2023)
8. Hussain, T., et al.: Improving Source location privacy in social Internet of Things using a hybrid phantom routing technique. Comput. Secur. **123**, 102917 (2022)
9. Ojha, A., S.P. Sahu, and D.K. Dewangan. *VDNet: vehicle detection network using computer vision and deep learning mechanism for intelligent vehicle system*. in *Proceedings of Emerging Trends and Technologies on Intelligent Systems: ETTIS 2021*. 2022. Springer.
10. Dewangan, D.K. and S.P. Sahu. *Predictive control strategy for driving of intelligent vehicle system against the parking slots*. in *2021 5th international conference on intelligent computing and control systems (ICICCS)*. 2021. IEEE.
11. Dewangan, D.K. and S.P. Sahu. *Real time object tracking for intelligent vehicle*. in *2020 first international conference on power, control and computing technologies (ICPC2T)*. 2020. IEEE.
12. Ottakath, N., Al-Maadeed, S.: Vehicle instance segmentation polygonal dataset for a private surveillance system. Sensors **23**(7), 3642 (2023)
13. Dewangan, D.K. and S.P. Sahu, *Lane detection for intelligent vehicle system using image processing techniques*. Data Science: Theory, Algorithms, and Applications, 2021: p. 329–348.
14. Farid, A., et al.: A Fast and Accurate Real-Time Vehicle Detection Method Using Deep Learning for Unconstrained Environments. Appl. Sci. **13**(5), 3059 (2023)
15. Zaman, K., et al., *A novel driver emotion recognition system based on deep ensemble classification*. Complex & Intelligent Systems, 2023: p. 1–26.
16. Wen, X., et al.: Efficient feature selection and classification for vehicle detection. IEEE Trans. Circuits Syst. Video Technol. **25**(3), 508–517 (2014)
17. Tomasi, C., *Histograms of oriented gradients*. Computer Vision Sampler, 2012: p. 1–6.
18. Saipullah, K., et al., *COMPARISON OF FEATURE EXTRACTORS FOR REAL-TIME OBJECT DETECTION ON ANDROID SMARTPHONE*. Journal of Theoretical & Applied Information Technology, 2013. **47**(1).
19. Suykens, J., Vandewalle, J.: Neural Process. Lett **9**, 293 (1999)
20. Hsiao, E., et al., *A discriminatively trained, multiscale, deformable part model*. 2009.
21. Saini, S., et al. *An efficient vision-based traffic light detection and state recognition for autonomous vehicles*. in *2017 IEEE Intelligent Vehicles Symposium (IV)*. 2017. IEEE.
22. Phan, H.N., et al. *Occlusion vehicle detection algorithm in crowded scene for traffic surveillance system*. in *2017 International Conference on System Science and Engineering (ICSSE)*. 2017. IEEE.
23. Ding, L., et al. *Scale-aware RPN for vehicle detection*. in *Advances in Visual Computing: 13th International Symposium, ISVC 2018, Las Vegas, NV, USA, November 19–21, 2018, Proceedings 13*. 2018. Springer.
24. Ramraj, S., et al.: Experimenting XGBoost algorithm for prediction and classification of different datasets. International Journal of Control Theory and Applications **9**(40), 651–662 (2016)
25. Girshick, R., et al. *Rich feature hierarchies for accurate object detection and semantic segmentation*. in *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2014.
26. Suykens, J.A., Vandewalle, J.: Least squares support vector machine classifiers. Neural Process. Lett. **9**, 293–300 (1999)
27. He, K., et al.: Spatial pyramid pooling in deep convolutional networks for visual recognition. IEEE Trans. Pattern Anal. Mach. Intell. **37**(9), 1904–1916 (2015)
28. Ren, S., et al., *Faster r-cnn: Towards real-time object detection with region proposal networks*. Advances in neural information processing systems, 2015. **28**.
29. Lin, T.-Y., et al. *Microsoft coco: Common objects in context*. in *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6–12, 2014, Proceedings, Part V 13*. 2014. Springer.
30. Everingham, M., et al.: The pascal visual object classes (voc) challenge. Int. J. Comput. Vision **88**, 303–338 (2010)
31. Nguyen, H.: Improving faster R-CNN framework for fast vehicle detection. Math. Probl. Eng. **2019**, 1–11 (2019)
32. Yin, G., et al.: Research on highway vehicle detection based on faster R-CNN and domain adaptation. Appl. Intell. **52**(4), 3483–3498 (2022)
33. Torralba, A., Russell, B.C., Yuen, J.: Labelme: Online image annotation and applications. Proc. IEEE **98**(8), 1467–1484 (2010)

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.