



Article

A Robust Framework for Object Detection in a Traffic Surveillance System

Malik Javed Akhtar ¹, Rabbia Mahum ^{1,*}, Faisal Shafique Butt ², Rashid Amin ³, Ahmed M. El-Sherbeeny ⁴, Seongkwan Mark Lee ^{5,*} and Sarang Shaikh ⁶

- ¹ Department of Computer Science, University of Engineering and Technology-Taxila, Taxila 47050, Pakistan
² Department of Computer Science, Wah Campus, COMSATS University Islamabad, Wah Cantt 47040, Pakistan
³ Department of Computer Science, University of Chakwal, Chakwal 48800, Pakistan
⁴ Industrial Engineering Department, College of Engineering, King Saud University, Riyadh 11421, Saudi Arabia
⁵ Department of Civil and Environmental Engineering, UAE University, Al Ain P.O. Box 15551, Abu Dhabi, United Arab Emirates
⁶ Department of Information Security and Communication Technology, Norwegian University of Science and Technology (NTNU), 2815 Gjøvik, Norway
* Correspondence: rabbia.mahum@uettaxila.edu.pk (R.M.); marklee@uaeu.ac.ae (S.M.L.)

Abstract: Object recognition is the technique of specifying the location of various objects in images or videos. There exist numerous algorithms for the recognition of objects such as R-CNN, Fast R-CNN, Faster R-CNN, HOG, R-FCN, SSD, SSP-net, SVM, CNN, YOLO, etc., based on the techniques of machine learning and deep learning. Although these models have been employed for various types of object detection applications, however, tiny object detection faces the challenge of low precision. It is essential to develop a lightweight and robust model for object detection that can detect tiny objects with high precision. In this study, we suggest an enhanced YOLOv2 (You Only Look Once version 2) algorithm for object detection, i.e., vehicle detection and recognition in surveillance videos. We modified the base network of the YOLOv2 by reducing the number of parameters and replacing it with DenseNet. We employed the DenseNet-201 technique for feature extraction in our improved model that extracts the most representative features from the images. Moreover, our proposed model is more compact due to the dense architecture of the base network. We utilized DenseNet-201 as a base network due to the direct connection among all layers, which helps to extract a valuable information from the very first layer and pass it to the final layer. The dataset gathered from the Kaggle and KITTI was used for the training of the proposed model, and we cross-validated the performance using MS COCO and Pascal VOC datasets. To assess the efficacy of the proposed model, we utilized extensive experimentation, which demonstrates that our algorithm beats existing vehicle detection approaches, with an average precision of 97.51%.

Keywords: CNN (convolution neural network); YOLO (You Only Look Once); intersection over union (IoU); mAP (mean average precision)



Citation: Akhtar, M.J.; Mahum, R.; Butt, F.S.; Amin, R.; El-Sherbeeny, A.M.; Lee, S.M.; Shaikh, S. A Robust Framework for Object Detection in a Traffic Surveillance System. *Electronics* **2022**, *11*, 3425. <https://doi.org/10.3390/electronics11213425>

Academic Editor: Byung Cheol Song

Received: 29 September 2022

Accepted: 19 October 2022

Published: 22 October 2022

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Copyright: © 2022 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Multimedia has deeply penetrated many realms of life in the present generation of promptly emerging technologies. In daily life, many people utilize electronic devices for various video applications i.e., animated videos, activity recognition, movies, etc. Cameras have been utilized quickly over the last century for surveillance systems. A surveillance system is a systematic method of monitoring behavior, actions, or other changing information. This results in massive data accumulation in the form of images and video clips, and it can be a tiring task to extract relevant information from this multimedia content. The three essential phases in every surveillance system are object detection, tracking, and recognition.

Object identification and localization is the procedure of locating the location of objects in images and videos captured by surveillance cameras. The objects can be classified and detected in real-time using various computer vision techniques [1]. Furthermore, the objects are specified by employing rectangular bounding boxes. There exist various applications of object detection in industries and scientific research based on ML (Machine Learning) and DL (Deep Learning), for example face detection [2], text detection [3], pedestrian detection [4], logo recognition [5], object identification in the video [6], vehicle detection [7], disease detection [8], medical imaging [9] and many more. Moreover, for vehicle detection in autonomous driving systems, numerous challenges are still faced such as the algorithm's inability to detect faraway vehicles due to their small size, blurry conditions, night view, and rainy seasons because of less precision in localization of present studies.

Traditional techniques based on ML (Machine Learning) have been used mostly for object detection; the object area is computed first using a sliding window, then various features are mined, and finally, traditional classifiers such as SVM (Support Vector Machine) are used to classify the objects. Although the results are satisfactory, these methods are incapable of accurately detecting and classifying objects ignoring the underlying deep features. A researcher used the Haar feature descriptor to extract the linear, center, diagonal, and edge features before classifying the objects using a Support Vector Machine. Moreover, employing a hand-crafted feature descriptor requires human effort. As a result, researchers are concentrating their efforts on deep learning algorithms like CNN, R-CNN, and YOLO, which have greatly improved object detection performance.

Existing deep learning methods for object detection are dedicated to simplifying the network and speeding up the detection process. These results are heavily reliant on the accuracy of the proposals generated, such as when, for example, the researcher employs a faster R-CNN approach to detect and count vehicles. Although this technique can accelerate the detection process, it has lower detection accuracy than other traditional methods. Most importantly, these methods are incapable of detecting distant vehicles. We propose an improved Yolov2 algorithm with Densenet-201 as a base network in video surveillance systems to detect far-away vehicles that appear in small sizes.

The rest of the paper is structured in the following manner. Motivation is covered in Section 2, while the study's related work is presented in Section 3. The problem statement is discussed in Section 4, the methodology is discussed in Section 5 and the proposed approach is presented in Section 6. Section 7 presents the experiments, and Section 8 concludes the findings.

2. Motivation

The motivation of the proposed model is to investigate the issue of object detection (i.e., vehicle detection) in videos obtained from surveillance cameras employing the improved YOLOv2 technique. Moreover, the vehicles have various sizes in videos, and as a result, conventional approaches have a hard time detecting vehicles precisely. An improved YOLOv2 algorithm is developed in this paper to cope with this challenge.

The key advantages of doing this investigation are as below:

1. The proposed model enhances an end-to-end trainable DL (Deep Learning) model i.e., improved YOLOv2 to detect tiny vehicles (e.g., Car, Bus, Truck) for video surveillance systems. Two benchmarks are utilized to train the projected technique for vehicle detection, and the outcomes showed that our proposed algorithm outperforms the already-used methods.
2. We employed Densenet-201 as a base network in YOLOv2, which extricates the most exemplary features from the samples due to direct connections among all layers to the classification layer. Furthermore, the proposed algorithm localizes the tiny objects with high precision effectively.
3. For the cross-validation of our proposed model, we utilized MS COCO and Pascal VOC datasets. We achieved significant vehicle detection performance for our proposed model than existing techniques, which confirms that our proposed technique is robust.

4. The proposed model is an efficient algorithm and performs a fast automated feature extraction than the existing object detectors such as Faster RCNN. Moreover, it predicts the coordinates of bounding boxes and classification at the same time. It is easy and simple to employ for object detection in real-time videos.
5. The proposed model is lightweight and has fewer parameters than the original YOLOv2 model.

3. Literature Review

Employing computer vision techniques to accurately identify on-the-road vehicles is a thought-provoking issue that has been a scorching research topic for the past two decades [10]. The surveillance videos of traffic have the ever-changing background due to lighting effects. As a result, the exact size and location of vehicles are difficult to capture due to the simultaneous movements of the vehicles on the road.

Recently, DL (Deep Learning) models have piqued the interest of numerous scholars, and a plethora of deep learning object detection algorithms have been introduced. In comparison to traditional methods, manual feature extraction in machine learning object detection algorithms needs experts with years of experience in the associated domain. Whereas, deep learning models necessitate a large amount of data to automatically acquire the characteristics that can imitate differences in data, making it more demonstrative. Simultaneously, the procedure of feature extraction in the CNN layer in visual recognition mechanisms is similar to the human visual mechanism. Deep learning-based detection algorithms have achieved reasonable real-time performance compared to traditional algorithms in recent years, requiring a continuous increase in data volume, and constant updates of device hardware, and have attained recognition worldwide. Due to the better real-time accuracy and performance in the academic field, the deep learning vehicle detection algorithm has been gradually developed in two directions; one is focused on accuracy and the other one is on complexity.

For more than a decade, researchers have studied vehicle detection and recognition extensively in the literature. Previously, numerous handcrafted features were removed for vehicle detection, which requires manual intervention. Haar [11], HOG [12], and LBP [13] were the three most commonly used feature descriptors. The classification framework was evaluated for vehicle detection and found to be effective, i.e., a large number of vehicles were detected. Additionally, the HOG feature in conjunction with the Support Vector Machine classifier is commonly used with great success in vehicle detection. Moreover, the mentioned features and classifiers with broad applications in vehicle detection tasks, and statistical techniques using vertical and horizontal edge features were initiated for the detection of vehicles and vehicle tracking at night by placing the tail lights. Table 1 presents the recent works done on vehicle detection and classification.

Table 1. Summary of existing techniques for vehicle detection.

Ref.	Methodology	Dataset	Evaluation Measures	Results
[14]	Real-time Vehicle Detection	The homemade dataset contains 9575 images.	YOLOv2 and Faster R-CNN.	An improved YOLOv2 framework is proposed and the results of the proposed YOLOv2 show that the accuracy is raised to 91.83%.
[15]	Object Detection	Pascal VOC Dataset 2007 and 2012, containing 20 categories of objects.	YOLOv1, R-CNN, and modified YOLO.	All the methods perform well but our modified model takes less processing time than the previous methods.

Table 1. Cont.

Ref.	Methodology	Dataset	Evaluation Measures	Results
[16]	Large-scale Vehicle Detection	A self-made dataset was used consisting of 8000 images obtained from 30 traffic surveillance cameras.	Feature descriptors, large-scale indexing, and feature selection.	A new method is proposed that could detect or track vehicles that are stolen in a challenging urban environment.
[17]	Vision-based Vehicle Detection	A KITTI benchmark dataset was used consisting of 11,129 images.	Deep learning, YOLOv3, and ORB algorithm.	The proposed method YOLOv3 measured two aspects i.e., vehicle driving direction and vehicle counting, and they achieved an accuracy of 92.3% and 93.2%.
[18]	Vehicle Detection and Recognition	A random dataset was used consisting of 20,000 images of which 5000 images were of vehicles.	Haar, AdaBoost algorithm, and LGBP histogram.	An accuracy of 97.3% was achieved when detecting vehicles in the traffic surveillance system and an accuracy of 92% was achieved when recognizing the vehicles.
[19]	Fast Vehicle Detection	The KITTI dataset has 15,000 images, while the LSVH dataset contains 16 videos.	CNN and SiNet.	The proposed method's detection accuracy is improved because of the usage of RoI pooling and a multi-branch detection network. The SiNet achieves an accuracy of 37 FPS.
[20]	Object Detection	COCO dataset.	Faster R-CNN-FPN architecture with ResNet50	The accuracy achieved for the COCO dataset is 89%.
[21]	Vehicle Detection	Two datasets i.e., BIT-Vehicle (9580 images) and CompCars (40,000 images) are used.	Improved YOLOv2 algorithm.	The proposed method achieved an accuracy of 94.78% in the BIT-Vehicle dataset.
[22]	Multi-Scale Vehicle Detection	A BIT-Vehicle public dataset was used consisting of 9580 images.	Improved YOLOv2 algorithm.	The proposed method revealed that we can better predict the sizes of vehicles. This method achieved superior performance accuracy of 97.30%.
[23]	Vehicle Detection	Two types of datasets i.e., KITTI and PASCAL VOC2007 are used.	CNN and HybridNet.	The experimental results on KITTI and PASCAL VOC 2007 show us that this method performs better than the previous methods.
[24]	Vehicle Detection	A BIT-Vehicle dataset was used consisting of 3618 daylight images and 1306 nightlight images.	Semi-supervised CNN and Softmax regression.	This proposed method acquired a 92% accuracy rate. The daylight recognition was 95.7% and the night-time recognition rate was 88.8%.

4. Problem Statement

A vast number of techniques have been developed in the past era that have added much-needed attention among researchers as a result of the development and improvements in the domain of CV due to their vast surveillance applications. As a result of advances in computer vision, the detection of objects in images is becoming increasingly important because it can benefit a vast number of applications, including human detection, face detection, vehicle detection, hammer detection, gun detection, knife detection, and many others. With the advancement of technology and the increased number of vehicles on

the road around the world, the traffic system has become increasingly reliant on automatic vehicle recognition systems. Consequently, the vehicle detection and recognition system must perform well in the context of time complexity and accuracy.

The main aim of this research paper is to investigate the challenge of vehicle detection in surveillance videos using deep learning. Due to the low-quality of surveillance images, lack of background information, low lighting, and distance it is quite difficult to detect vehicles.

Hence, we deduce a technique i.e., Improved YOLOv2 through surveillance videos that distinguish between vehicles and non-vehicles based on deep learning. From various research studies done previously, it is deduced that the current surveillance systems cannot efficiently tell us which kind of model should be used for what kind of images. Surveillance systems usually fail because they rely heavily on human operators who have physical restrictions in the form of lethargy or loss of attentiveness due to monitoring several screens for longer periods. These restrictions can be eased by enhancing the surveillance systems to automatically detect the various objects that are present in an image. These proficiencies can then enable surveillance systems to detect objects in various images. To have such proficiencies, we need to deduce a mechanism that can not only capture images but can also account for human emotion and behavior i.e., a method like object detection has to be introduced that can detect the difference between different objects in an image.

Although various traditional methods are used to detect vehicles in surveillance videos, the main problem is that the traditional methods are not as accurate and also, they are very expensive. Nowadays researchers focus on using deep learning methods to recognize vehicles. In this research, the Improved YOLOv2 algorithm, a type of DL (Deep Learning) technology, was utilized to detect various vehicles (e.g., Car, Bus, Truck) observed in surveillance cameras.

5. Materials and Methods

Conventionally, deep learning contains numerous layers of nonlinear processing modules to obtain the features. All layers are cascaded and take the output from the previous layer as input. Many researchers have attempted to build the network deeper and larger to investigate the potential of deep learning. However, it has a challenge with exploding or the vanishing gradient problem (VGP). As a result, many researchers build multiple different structures of deep learning.

A range of deep learning structures has been proposed such as AlexNet [25], ResNet [26], DenseNet [26], GoogLeNet [27], VGGNet [28]. The 2012 ImageNet Large Scale Visual Recognition Competition (ILSVRC) winner, AlexNet, is comparable to LeNet and has ReLU non-linearity and max-pooling. In the 2014 ILSVRC, VGGNet came at second place, with deeper networks (19 layers) than AlexNet. To extract sparse correlating features in feature map stacks, GoogLeNet, the ILSVRC 2014 winner, uses 1×1 convolution to minimize the dimensions of feature maps earlier than the expensive convolutions, as well as parallel routes with variable receptive field sizes. ResNet, the ILSVRC 2015 winner, proposes a 152-layer network with a minimum of 2 layers of skipped or shortcut connections. Whereas, each layer in DenseNet feeds forward the output of all preceding layers, providing $N(N + 1)/2$ connections in N layers, whereas outdated convolutional networks with N layers only deliver N connections. DenseNet is capable of performing better than the cutting-edge ResNet structure in the ImageNet classification test.

In this research, we proposed DenseNet201 as the base network in YOLOv2 for vehicle detection (e.g., Car, Bus, Truck) because of its remarkable performance. However, before going into detail about DenseNet201, the traditional convolution neural network (CNN) will be discussed first, followed by the distinctions between DenseNet and CNN.

5.1. Convolution Neural Network (CNN)

A standard convolution neural network (CNN) normally includes (i) Convolution (CONV) layer, (ii) Rectified linear unit (ReLU) layer, (iii) pooling (POOL) layer, (iv) Fully connected (FC) layer, and (v) Softmax layer [29]. The following are the functions of the several layers, with the convolution layer as the fundamental session of a CNN. Convolutional input with various kernels produces the feature maps. It can be expressed mathematically as presented in Figure 1.

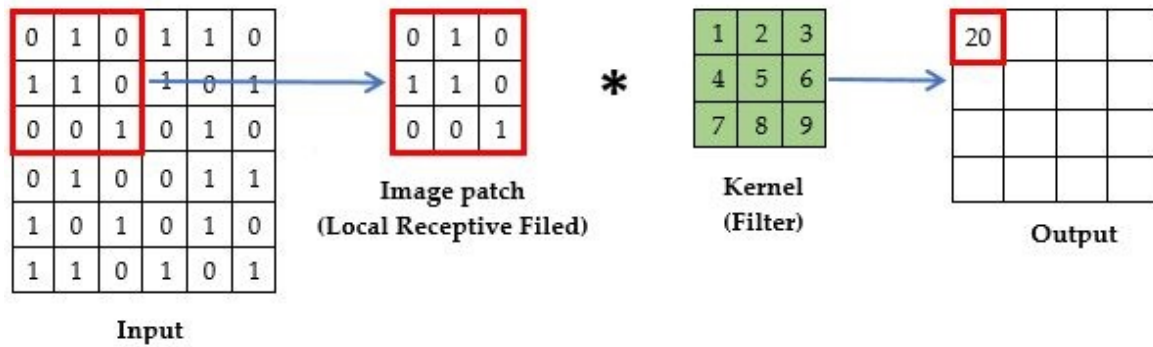


Figure 1. CNN Convolution operation (* represents multiplication).

Succeeding the convolution layer, there exists the ReLU nonlinear activation function, which is used to extract nonlinear features. The goal of the ReLU layer is to impart non-linearity to the network. It is mathematically defined as Equation (1).

$$relu(v) = \max(v, 0); \tag{1}$$

The pooling layer works by geographically resizing the feature maps to reduce the parameters, memory footprint, and network computation time. Each feature map is subjected to the pooling function, and the most common pooling approaches are max pooling as shown in Equation (2), and average pooling as presented in Equation (3).

$$a_k = \frac{1}{|R_k|} \sum_{j \in R_k} M_j \tag{2}$$

$$a_k = \max_{j \in R_k}(M_j) \tag{3}$$

M denotes the pooling region, while R_k represents the total elements along with the pooling region. The confidential scores will be calculated through fully connected layers and stored in a $1 \times 1 \times c$ volume. Each element represents class scores, while c refers to the categories.

An individual neuron in the FC layer is linked to neurons in previous layers. In a typical CNN, all the layers are progressively associated, as shown in Equation (4).

$$m_r = F_r(m_{r-1}) \tag{4}$$

However, when the network grows deeper and larger, it is possible that the network could explode or the gradient would vanish. As a result, researchers offered various network architectures to solve the problem. ResNet, for example, changed this behavior by using a short link as shown in Equation (5).

$$m_r = F_r[(m_{r-1}) + m_{r-1}] \tag{5}$$

Rather than summing the feature maps' outputs of the layer to the incoming feature maps, DenseNet has direct connections among all layers and each current layer takes input from all previous layers. The expression is rewritten as Equation (6).

$$m_r = F_r[(m_0, m_1, m_2, \dots, m_{r-1})] \quad (6)$$

where r denotes the layer number's index, F denotes a non-linear function and m_r denotes the r -th layer's output.

5.2. Densenet-201

Due to the capacities of feature reusability by succeeding layers, the DenseNet-201 employs the condensed network, allowing the tremendously parametrically efficient model, which increases diversity in the succeeding layer input and enhances performance. The DenseNet201 has performed admirably on a variety of datasets, including ImageNet [30] and CIFAR-100 [31]. Direct connections from all preceding layers to all future layers are introduced to boost connectivity in the DenseNet201 architecture, as shown in Figure 2.

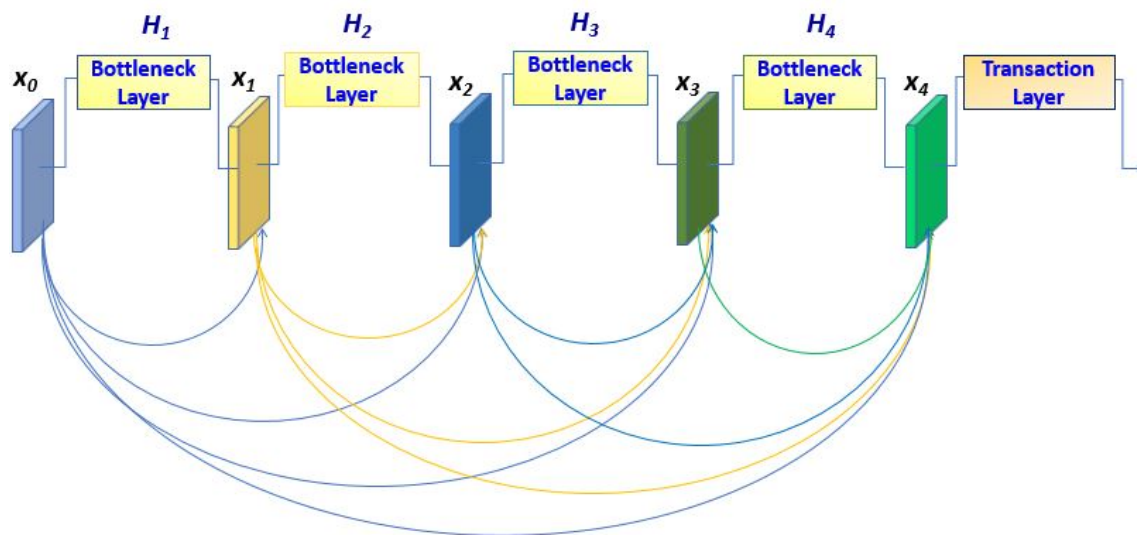


Figure 2. Direct connections in DenseNet201.

The advantages of DenseNet201, which includes 201 convolutional layers, are fewer vanishing-gradient problems, excellent feature distribution, feature reusability, and a fewer number of parameters.

Let's assume that an image m_0 is fed into a neural network with R layers and non-linear transformation $F_r(\cdot)$, where r is the index of the layer. ResNet's traditional skipping connections are included in the feed-forward network that bypasses the non-linear alteration with an identity function, as shown in Equation (7).

$$m_r = F_r(m_{r-1}) + m_{r-1} \quad (7)$$

ResNet has one advantage here that from initial layers till final layer, a gradient can move straight through the identity function. Whereas, direct end-to-end connections are used in the dense network to maximize the amount of information in each layer. The r -th layer receives all of the previous layer's information as shown in Equation (8).

$$m_r = F_r[(m_0, m_1, \dots, m_{r-1})] \quad (8)$$

In DenseNet, down sampling takes place at Dense Blocks, which are split into Transition layers; it contains a 1×1 convolutional layer (CONV) and a pooling layer (average) with BN (batch normalization). The bulks from the transition layer ultimately spread to

the dense layers. We transformed the entire average-pooling layer into a 2×2 max pool layer for network utility. BN (Batch normalization) is performed previously in each of the convolutional layers, making the model less complex. The hyperparameter k denotes the network’s growth rate, making the DenseNet capable of producing cutting-edge results. Pooling layers are eliminated, and the proposed detection layers are fully integrated and related to the classification layers for detection. Even deeper network designs than the 201-layer network can be found in DenseNet-264 [32]. Because we don’t want to cast a wide network, the 201-layer structure is suitable for detecting vehicles. Due to its manner, which reflects feature maps as a global mechanism of the network, DenseNet201 performs well even with a smaller growth rate. Figure 3 exhibits the DenseNet201 architecture:

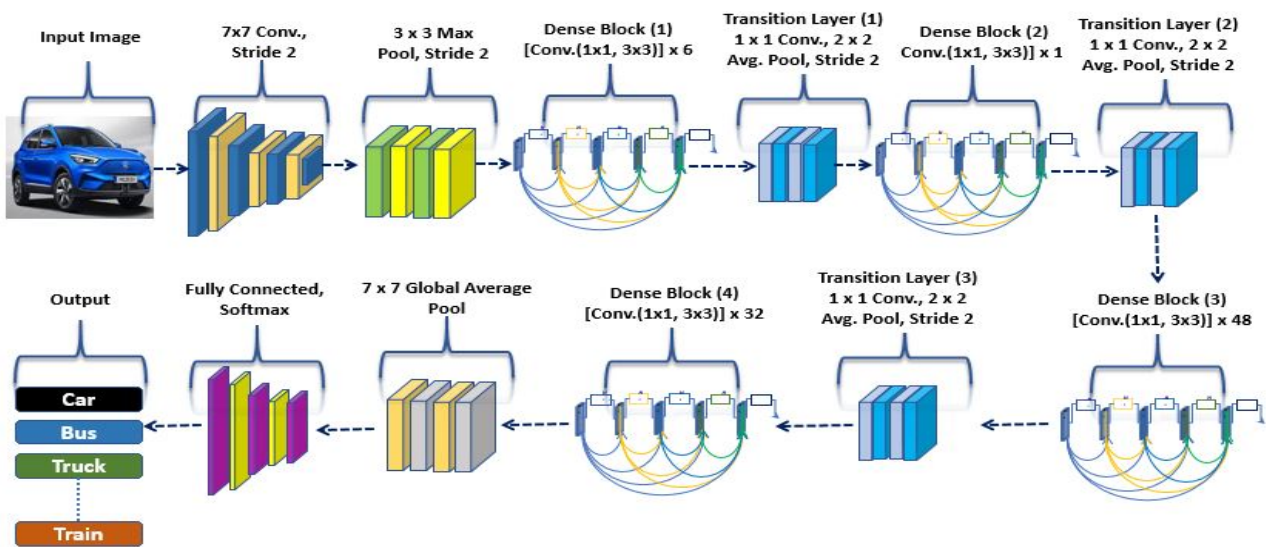


Figure 3. DenseNet201 architecture (feature extraction block).

DenseNet-201 is based on the transfer learning concept, having 201 depth layers and 20 million parameters that have been trained using more than one million images attained from the ImageNet dataset.

5.3. YOLO (You Only Look Once) Theory

YOLO is an abbreviation of “You Only Look Once” [33], an advanced, one-stage algorithm, to identify objects in real-time. The YOLO technique uses CNN, and object recognition is performed as a regression scenario. CNN is employed to predict various bounding boxes and class probabilities simultaneously. In comparison to Faster R-CNN, YOLO obtains location and category predictive information without a region proposal network (RPN).

5.4. Working Principle of YOLO

At the start, the network splits the input image into the $R \times R$ grid. When the central point of an object lies in a grid cell, that grid cell is responsible for the detection of that object. B bounding boxes and confidence scores are predicted in each grid cell for those bounding boxes. $Prob(Object)$ stands for whether there is a required object falling into this cell. The mathematical equation of confidence C in YOLO-v2 is shown in Equation (9).

$$C(Confidence) = Prob(Object) * IoU_{pred}^{truth} \tag{9}$$

Here, each grid cell predicts C conditional class probabilities, $Pr(Class | Object)$, $Prob(Object)$ is the probability of predicting whether the boundary object contains the vehicle object. If the object is present, $Prob(object)$ is equal to 1, otherwise it is equal to 0.

There are five components of the bounding box (x_0 , y_0 , wd , ht , confidence). The confidence score reflects how self-assured the model is in the predicted box containing an object and how correctly the box is that it predicts. The (x_0, y_0) coordinates refer to the center of the box related to the bound of the grid cell and these coordinate values lie between 0 and 1. The (wd, ht) box dimensions are width and height of the relative bounding box to the whole image and are also normalized to 0 and 1. The category probability p is calculated as shown in Equation (10).

$$\text{Prob}(Class_i | Object) * \text{Prob}(Object) * IoU_{pred}^{truth} = \text{Prob}(Class_i) * IoU_{pred}^{truth} \quad (10)$$

The confidence score is zero if no object lies in that cell. Otherwise, the confidence score should be equivalent to the intersection over union (IoU) of the actual and predicted boxes. Each grid cell creates B of these predictions, and there exist a total of $R \times R \times B \times 5$ outputs connected to bounding box predictions. The last layer of the pre-trained CNN model predicts the tensor of size $R \times R \times (B \times 5 + C)$, where C is several classes.

If multiple objects exist in a single grid cell then to resolve this problem, we utilized the concept of an anchor box. The anchor box enables the YOLOv2 to identify several objects in a single grid cell. Due to this, a new idea of an anchor box i.e., one more dimension, is added to the output labels by predefining several anchor boxes. After that, one object will be assigned to each anchor box. Figure 4 illustrates the framework of the YOLO methodology.

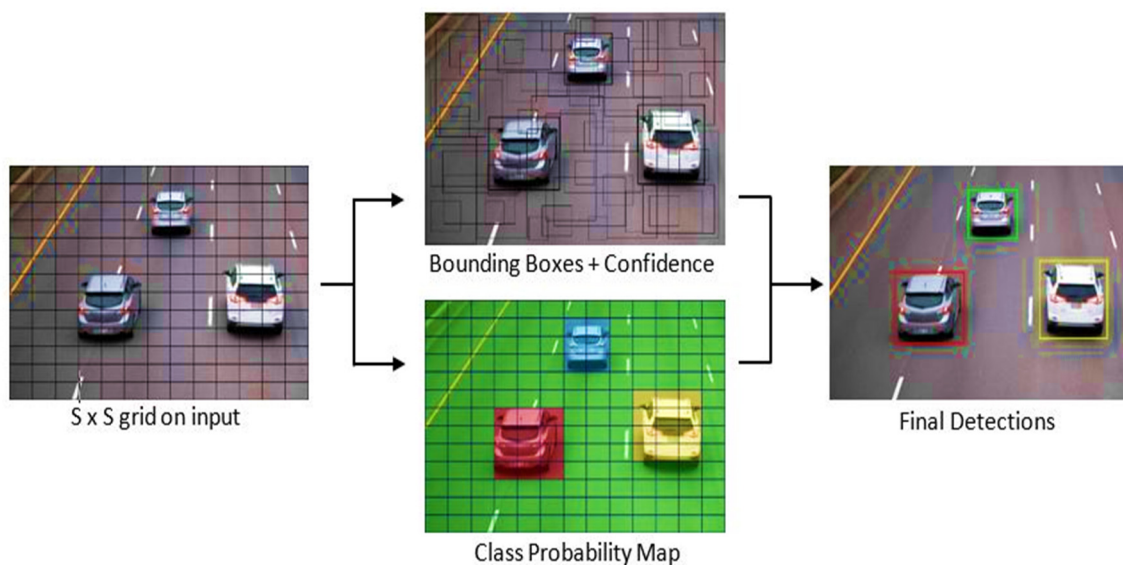


Figure 4. The framework of the YOLO methodology.

5.5. Loss Function

The loss is split into two sub-parts, a loss for localization for predicting bounding box offsets and a classification loss for predicting the probabilities of conditional class. The squared error sum is utilized to compute both parts. Two scale parameters are used to determine how much the loss from bounding box coordinates predictions should be increased λ_{coord} and how much we want to reduce the number of confidence score predictions for boxes that are lost without objects λ_{noobj} . As a result, the weighted technique is used to balance the various types of losses. Generally, λ_{coord} is set as 5 and λ_{noobj} set as 0.5 to minimize each loss. Otherwise, each loss may contribute differently to the overall loss, rendering certain losses unsuccessful for network training. The loss equation is shown in Equation (11):

$$\begin{aligned} & \lambda_{coord} \sum_{i=0}^{s^2} \sum_{j=0}^B \Pi_{ij}^{obj} [(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2] + \lambda_{coord} \sum_{i=0}^{s^2} \sum_{j=0}^B \Pi_{ij}^{obj} [(\sqrt{w_i} - \sqrt{\hat{w}_i})^2 + (\sqrt{h_i} - \sqrt{\hat{h}_i})^2] \\ & + \sum_{i=0}^{s^2} \sum_{j=0}^B \Pi_{ij}^{obj} (C_i - \hat{C}_i)^2 + \lambda_{noobj} \sum_{i=0}^{s^2} \sum_{j=0}^B \Pi_{ij}^{noobj} (C_i - \hat{C}_i)^2 + \sum_{i=0}^{s^2} \Pi_{ij}^{obj} \sum_{c \in classes} (p_i(c) - \hat{p}_i(c))^2 \end{aligned} \tag{11}$$

where x_i and y_i represent the center coordinates, w_i and h_i refer to the width and height of the box, C_i represents the confidence of the box, and $p_i(c)$ is the class probability related to the box of the i -th grid cell. Moreover, the equivalent predictions of x_i , y_i , w_i , h_i , C_i , and $p_i(c)$ are \hat{x}_i , \hat{y}_i , \hat{w}_i , \hat{h}_i , \hat{C}_i , and $\hat{p}_i(c)$, the weight of the loss coordinates is λ_{coord} , and λ_{noobj} represents the weight of the bounding boxes without any objects loss. S_2 indicates the $S \times S$ grid cells, B indicates the boxes whether there is an object that falls in the j -th bounding box of the i -th grid cell, and λ_{noobj} refers to the confidence consequence when there is no object. In Equation (11),

$$\lambda_{coord} \sum_{i=0}^{s^2} \sum_{j=0}^B \Pi_{ij}^{obj} [(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2]$$

is responsible for calculating the coordinate loss,

$$\lambda_{coord} \sum_{i=0}^{s^2} \sum_{j=0}^B \Pi_{ij}^{obj} [(\sqrt{w_i} - \sqrt{\hat{w}_i})^2 + (\sqrt{h_i} - \sqrt{\hat{h}_i})^2]$$

is responsible for computing the bounding box size loss,

$$\sum_{i=0}^{s^2} \sum_{j=0}^B \Pi_{ij}^{obj} (C_i - \hat{C}_i)^2$$

is responsible for determining the bounding box confidence loss with objects,

$$\lambda_{noobj} \sum_{i=0}^{s^2} \sum_{j=0}^B \Pi_{ij}^{noobj} (C_i - \hat{C}_i)^2$$

will calculate the bounding box confidence loss without objects, and

$$\sum_{i=0}^{s^2} \Pi_{ij}^{obj} \sum_{c \in classes} (p_i(c) - \hat{p}_i(c))^2$$

is responsible for calculating the class loss.

6. Proposed Solution

Our proposed solution network’s structure comprises (i) the Input layer, (ii) network for feature extraction, and (iii) detection network. The first stage in the network is to balance the size of an input image to 224×224 pixels, after which the scaled data is passed into DenseNet-201 for Feature Extraction. As previously indicated, we replaced the YOLOv2 baseline network Darknet-19 with DenseNet-201 and associated procedures, and now we are looking into the network’s detection adjustments. The complete structure of our proposed system is depicted in Figure 5.

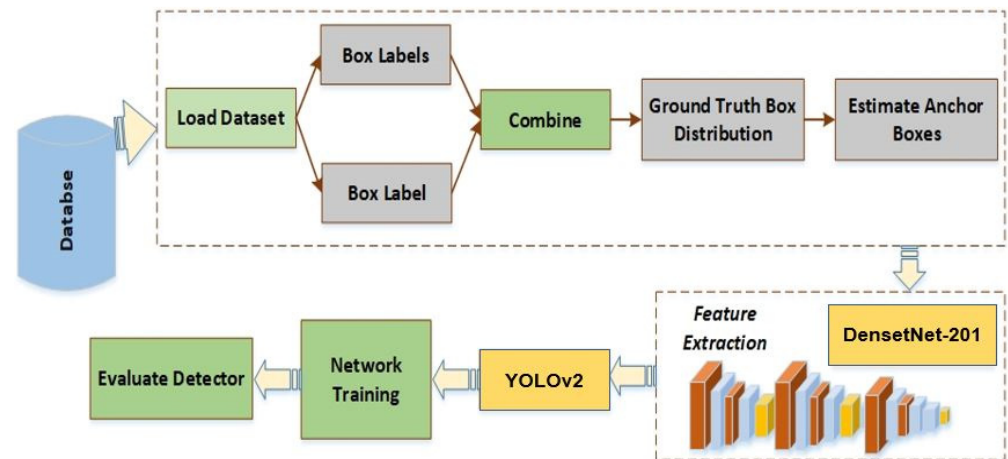


Figure 5. The overall structure of the proposed model.

7. Experimental Evaluation

7.1. Dataset

Dataset is the main foundation to estimate any model’s performance. Improving the recognition rate of the proposed model requires sufficient data for vehicle detection training. More training data can enhance the recognition and generalization rate as well as the robustness of the model, whereas overfitting problems may occur due to an insufficient amount of datasets. We used two datasets, Kaggle [34] vehicle and KITTI [35] datasets for the training and testing of the model. Moreover, the MS COCO [36] dataset and Pascal VOC [37] dataset were used to cross-validate the proposed model.

7.1.1. Kaggle Vehicle Dataset

The vehicle dataset available on Kaggle is used for experimental purposes. The dataset is split into two parts i.e., train set and the test set. The Kaggle vehicle dataset contains 22,852 training images and 5193 test images, containing a total of 28,045 images. There exist 17 classes (Ambulance, Car, Cart, Boat, Bus, Caterpillar, Helicopter, Barge, Bicycle, Segway, Limousine, Motorcycle, Tank, Taxi, Snowmobile, Truck, and Van). The class-wise distribution of Kaggle datasets is presented in Table 2.

Table 2. Comprehensive overview of the Kaggle dataset.

Class	Total Images (Kaggle)	Training Images	Testing Images
Ambulance	132	44	88
Barge	202	42	160
Bicycle	1618	122	1496
Boat	8695	786	7909
Bus	2133	351	1782
Car	6781	1391	5390
Cart	51	29	22
Caterpillar	331	45	286
Helicopter	532	15	517
Limousine	74	63	11
Motorcycle	2986	797	2189
Segway	153	65	88
Snowmobile	123	46	77
Tank	206	85	121
Taxi	748	221	527
Truck	2033	559	1474
Van	1111	396	715
Total	27,909	5057	22,852

7.1.2. KITTI

The KITTI dataset is freely available having 80,256 labeled objects in numerous images. We utilized 7481 training photos and 2000 test images. All of the images are colored and have been saved as “png” files. There are 80 classifiers (Car, Bus, Truck, Train, Motorcycle, etc.). The class-wise distribution KITTI dataset is described in Table 3.

Table 3. Class-wise distribution of KITTI datasets.

Total Number of Classifiers:		80	
Class	Total Images (KITTI)	Training Images	Testing Images
Car	11,682	7481	2000
Bus			
Truck			
Motorcycle			
Train			

7.1.3. Pascal VOC

Pascal VOC contains 20 different classes (Vehicle: train, bicycle, boat, bus, airplane, etc.), and 9963 images consisting of 24,640 annotated objects. For vehicle detection, we utilized various class samples from the Pascal VOC dataset. More precisely, we employed 800 images in total to evaluate our proposed classifier for the detection of vehicles.

7.1.4. COCO

Common Objects in Context (COCO) is one of the most famous open-source datasets for object identification and segmentation. Microsoft sponsors the COCO dataset, which contains over 300,000 images and 90 object types. In recent years, semantic segmentation has become the industry standard for image semantics understanding. Thus, we employed only 500 images exhibiting various vehicles from the COCO dataset. Various training samples are presented in Figure 6.



Figure 6. Various samples for training.

7.2. Metrics

To analyze the performance of the proposed system, we have utilized the metric of Accuracy [9], Intersection Over Union (IoU) [38], and mean Average Precision (mAP) [39].

Accuracy relies upon true positive (*TP*) [40], false positive [41] (*FP*), true negative (*TN*), and false negative (*FN*). Furthermore, the accuracy of the system indicates the correctly classified images by the proposed system. Equation (12) is presented below.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (12)$$

We have employed *mAP* i.e., the average precision to analyze the performance of our proposed detector. The Equation (13) is shown below, where *Q* denotes the total number of test images.

$$mAP = \sum_{i=1}^Q \frac{AP(q_i)}{Q} \quad (13)$$

7.3. Environment

We performed the experiments using a GPU NVIDIA card i.e., GEFORCE RTX 30 with 4 GB memory. The operating system was Windows 10 having a RAM of 16 GB. The experiment was performed using Matlab 2021a. We trained our classifier for various categories of vehicles employing parameters such as epochs: 100 and learning rate: 0.0001.

The primary goal of this paper is to propose an accurate approach for the detection of vehicles correctly. The various experiments performed can provide insight into the method's robustness and capacity to run in real-time scenarios. To achieve a reliable vehicle detector, we proposed an Improved YOLOv2 using DenseNet201 as the base algorithm employing a transfer learning (TL) mechanism. The proposed model is based on the outstanding performance of DenseNet as it performs on ImageNet dataset classification tasks. Figure 7 shows results of the proposed model for the detection of Vehicles using the Kaggle Vehicle dataset.

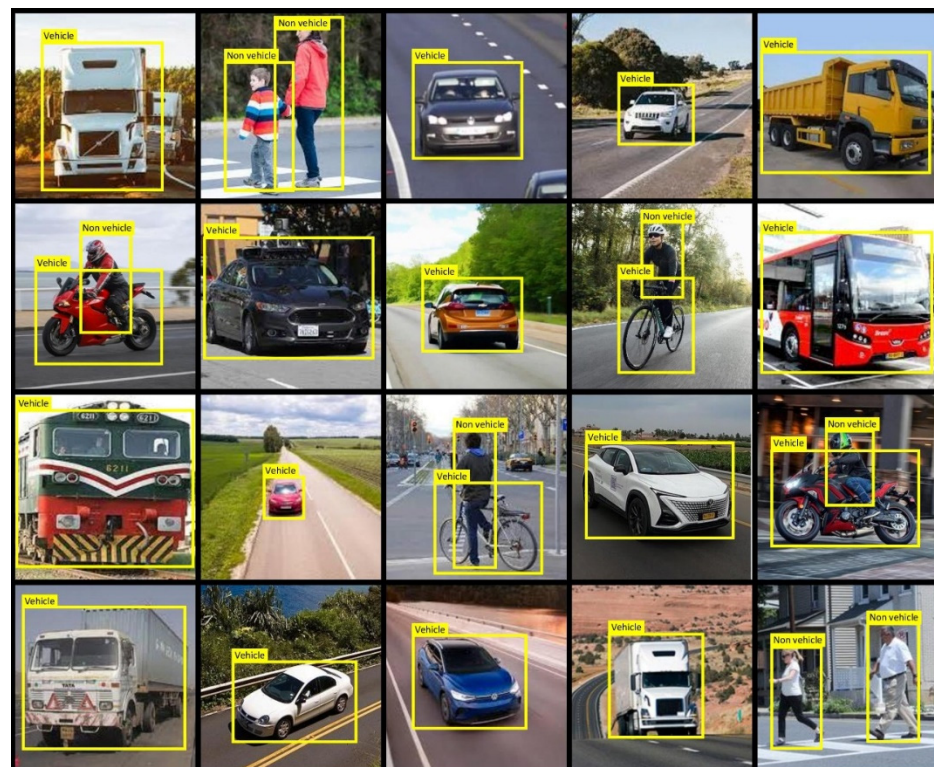


Figure 7. Results of the proposed model's detection by using the Kaggle Vehicle dataset.

7.4. Class-Wise Performance

The average precision (AP) for each vehicle class, was used to measure the performance of recognition. The average recognition performance is depicted by the mean Average Precision (mAP), whereas intersection over union (IoU) indicates the average localization performance. In Object detection, mAP and IoU are significant measures for evaluating a model's performance. Table 4 shows that the proposed upgraded YOLOv2 with Densenet201 has an mAP of 97.51% and an IoU of 97.06%. Improved YOLOv2 with Densenet20 worked well for single and multiple vehicle identification, according to our findings. In our purposed method, the mAP of Taxi and Van reaches up to 98.9%, while the remainder of the results ranges from 94.5% to 98.8%. In terms of localization and recognition accuracy, our proposed technique surpassed others.

Table 4. Class-wise performance over Kaggle and KITTI dataset.

Model	Class	mAP (%)	IoU (%)
Improved YOLOv2-DenseNet201	Ambulance	97.4	98.3
	Barge	98.6	98.2
	Bicycle	98.4	97.4
	Boat	96.4	98.2
	Bus	97.2	98.5
	Car	98.3	95.4
	Cart	94.5	96.3
	Caterpillar	95.3	96.2
	Helicopter	94.8	96.3
	Limousine	95.9	94.3
	Motorcycle	98.7	97.4
	Segway	98.6	97.3
	Snowmobile	98.5	97.8
	Tank	98.8	97.1
	Taxi	98.9	97.3
	Truck	98.5	97.4
	Van	98.9	96.7
	Average	97.51	97.06

7.5. Cross-Validation

The Pascal VOC and MS COCO datasets have been employed for the cross-validation of the proposed model. For vehicle detection, we employed various samples from Pascal VOC and MS COCO datasets. Using DenseNet-201, we determined the mAP for each of the 20 classes in the PASCAL VOC dataset for Improved YOLOv2, and we achieved 81% mAP, which was approximately 2 percent higher than YOLOv2. Furthermore, our proposed model achieved promising results and outperformed other detectors, as shown in Table 5. For 1000 iterations, the training took around one hour. It was exhibited that Fast RCNN [42] attained 70% mAP, YOLOv2 [43] achieved 76.8%, and Faster RCNN with ResNet [43] achieved 76.4% mAP. The highest mAP was 81%, which was attained by our proposed model, whereas the least mAP was 63.4% which was attained by YOLO [33]. Moreover, SSD300 [44] and SSD500 [44] achieved 74.3% and 76.8% mAP, respectively. On the other side, Faster RCNN along with VGG-16 [45] and Improved YOLOv3-Net [46] achieved 73.2% and 77.4% mAPs. It is concluded that our proposed algorithm transcends the existing models due to an improved base network DenseNet-201. Our base network retrieves the most relevant features, and due to dense connections the flow of information is

accurate till the last layer. More precisely our proposed model is robust, to perform accurate detection due to its dense architecture. In Figure 8, the comparison plot is depicted.

Table 5. Comparison of different Network Models using PASCAL VOC 2007.

Models Names	mAP
Fast R-CNN [42]	70.0
YOLOv2 [47]	76.8
Faster R-CNN ResNet [43]	76.4
YOLO [33]	63.4
SSD300 [44]	74.3
SSD500 [44]	76.8
Faster R-CNN VGG-16 [45]	73.2
Improved YOLOv3-Net [46]	77.4
Improved YOLOv2 (DenseNet-201) Proposed Model	81.0

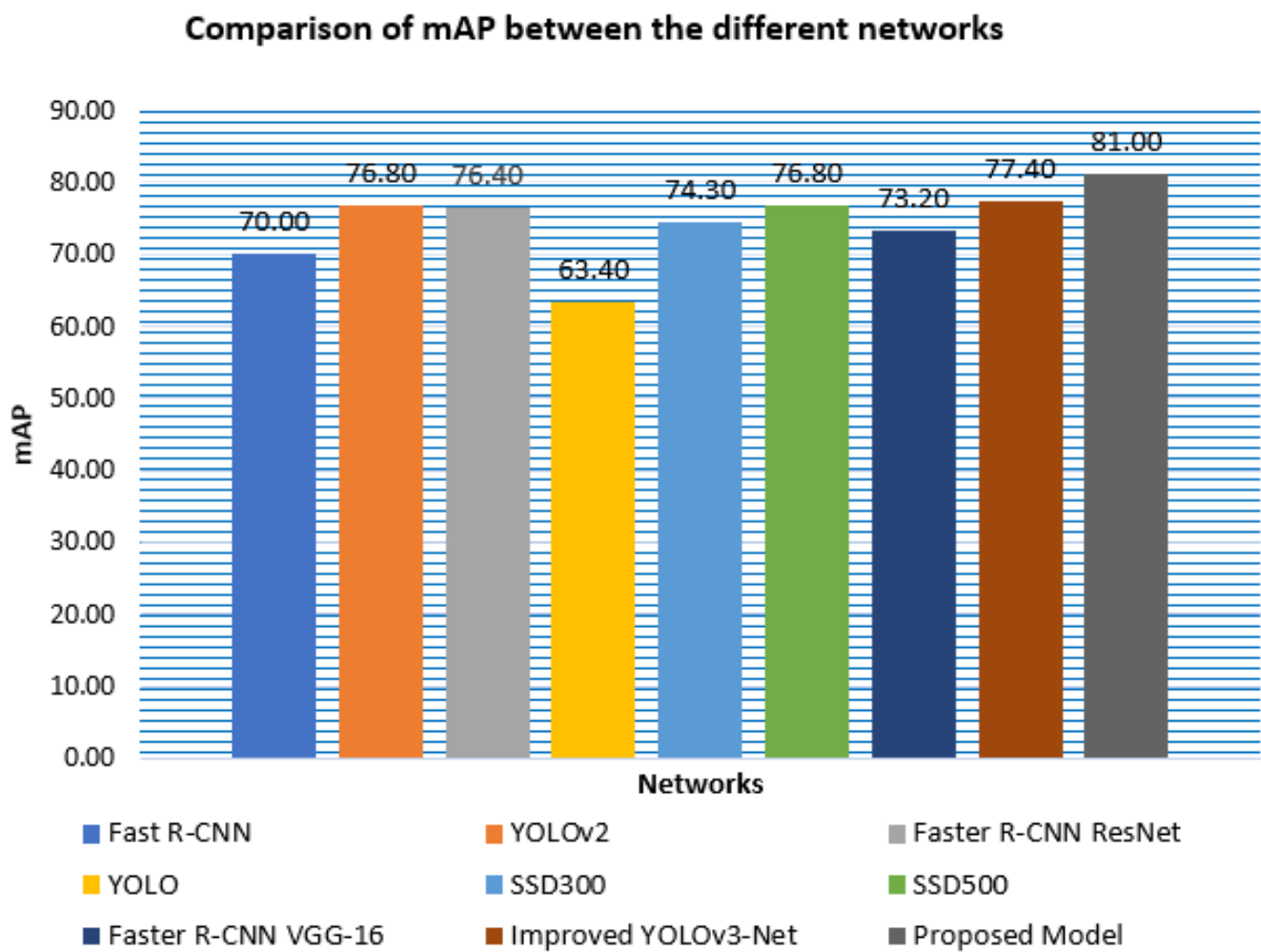


Figure 8. Comparison of mAP between the different networks with the proposed network.

7.6. Comparison with Existing Models

To evaluate the performance of our proposed model, we conducted two separate experiments. In the first experiment, we employed Pascal VOC 2007 to train our detector for vehicle detection. We analyzed the effectiveness of the proposed technique and matched it with predominant techniques over Pascal VOC 2007 dataset. We utilized only three class samples from the dataset as Bus, Car, and Truck. The proposed model performed significantly better than existing techniques. This training method employed a batch size of 64 and 0.001 is the learning rate. It was done using the IoU Threshold of 0.50. Four distinct dimensions of network models have been perceived such as Improved YOLOv2, YOLOv3, and YOLOv3-Net and our proposed model Improved YOLOv2-Net-201. The statistics are shown in Table 6. The best mAP of 82.7% was achieved for Improved YOLOv2-Net-201 due to the proposed dense architecture as the base network in YOLOv2. Each layer attains data from all the preceding layers and passes it to all coming layers. More precisely, the classification layer has a direct connection with previous layers, extracting the most valuable features for the detection of vehicles. Our proposed model is capable of significant vehicle detection and outperforms the existing techniques. The comparison plot is presented in Figure 9, exhibiting the better performance among existing models.

Table 6. Comparison of Proposed Networks with existing models on Pascal VOC Dataset.

NETWORK MODEL NAME	CORE NETWORK	AP (Car)	AP (Bus)	AP (Truck)	mAP
YOLOv2	Darknet19	78.9	67.3	72.3	72.8
YOLOv3	Darknet53	69.4	78.6	77.3	75.1
Improved YOLOv3-Net	DenseNet-121	86.2	79.8	77.5	81.1
Improved YOLOv2-Net-201 (Our)	DenseNet-201	88.3	81.2	83.9	82.7

Comparison Graph on Pascal VOC dataset

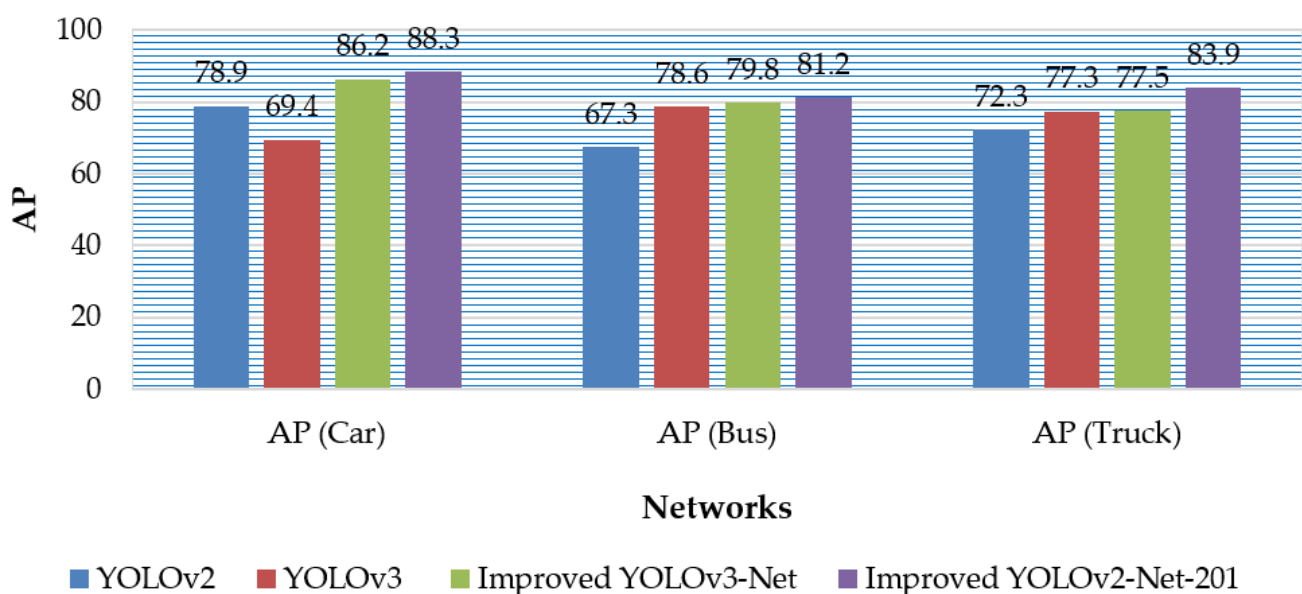


Figure 9. Comparison Graph with existing models for Car, Bus, and Truck samples using the PASCAL VOC dataset.

In the second phase, the COCO dataset has been used to train the detector for vehicle detection like Buses, Car, and Trucks. The statistics are shown in Table 7. The best mAP of 75.1% was achieved by our proposed model, and the least mAP was 60% attained by the original YOLOv2. Meanwhile, YOLOv3 and Improved YOLOv3 achieved 66.2% and 71.2% mAPs, respectively. Our proposed model has attained the best performance among existing models. Our proposed model effectively identifies the vehicles more than the predominant models. Moreover, our model is based on DenseNet which overcomes the problem of vanishing gradient and is better in terms of compactness than ResNet. The comparison graph for performance over the COCO dataset is shown in Figure 10.

Table 7. Comparison of Proposed Network with existing models on the COCO dataset.

NETWORK MODEL NAME	CORE NETWORK	AP (Car)	AP (Bus)	AP (Truck)	mAP
YOLOv2	Darknet19	67.5	61.5	51.1	60.0
YOLOv3	Darknet53	72.5	59.8	66.4	66.2
Improved YOLOv3-Net	DenseNet-121	76.5	65.3	72.9	71.6
Improved YOLOv2-Net-201 (Our)	DenseNet-201	79.9	73.0	72.3	75.1

Comparison Graph on COCO dataset

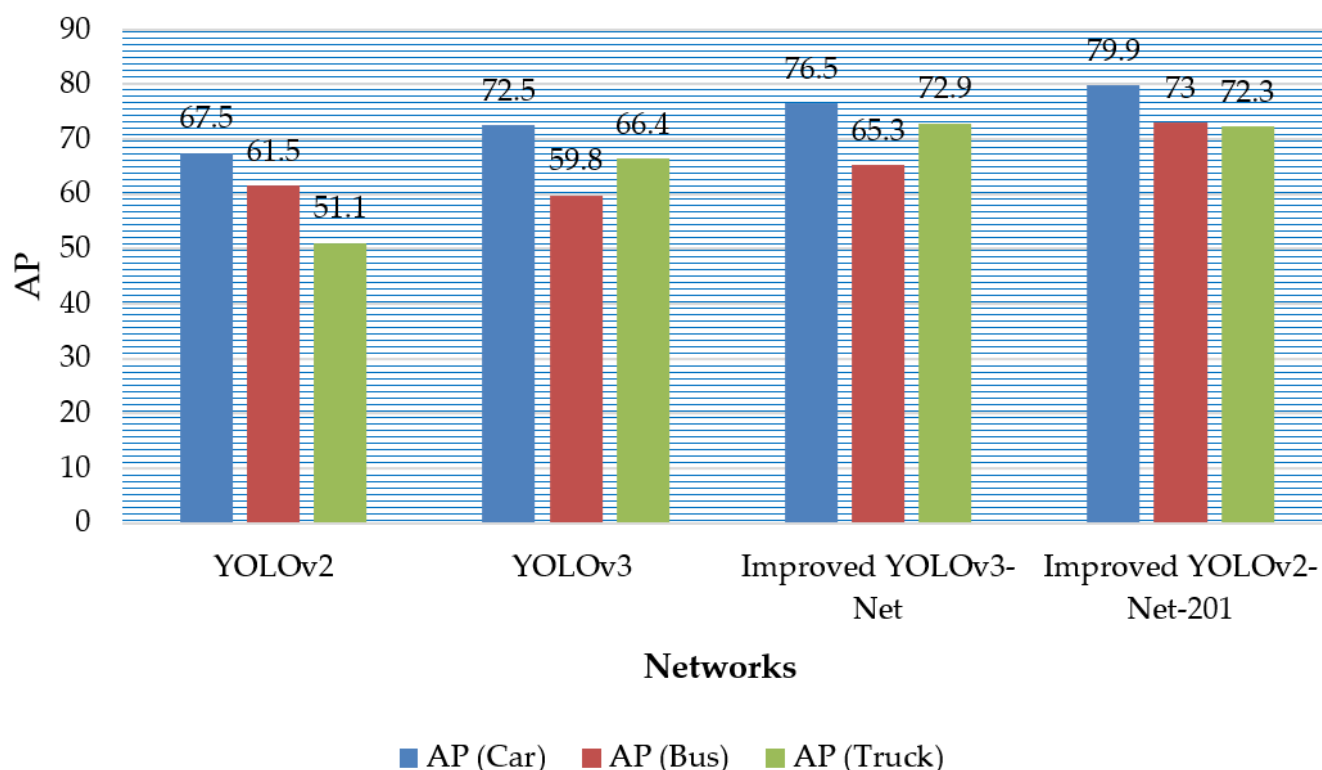


Figure 10. Comparison Graph with existing models for Car, Bus, and Truck samples using the COCO dataset.

8. Conclusions

In this study, an innovative and vigorous system for Vehicle detection is proposed using a deep neural network established on YOLOv2 (You Only Look Once). Our proposed technique uses DenseNet-201 as a Feature Extraction network swapping darknet18 in the original YOLOv2. We employed two benchmarks such as the Kaggle vehicle dataset

and the KITTI dataset as: 70% for training and 30% for testing of our proposed model. Moreover, we utilized samples from 17 classes exhibiting various vehicles such as buses, trucks, cars, carts, bikes, etc. We performed extensive experimentation to evaluate the performance of the proposed model and achieved better average precision for our model than existing techniques. Moreover, our proposed model is more compact and utilizes more representative features due to dense connections among layers. More precisely, each coming layer is directly connected with all previous layers till the classification layer in our proposed base network, and this mechanism ensures a good flow of information from the input layer to the last one. Furthermore, our proposed model detects tiny vehicles with more precision and more accurately calculates bounding boxes due to compactness in the base network than the original YOLOv2. We also performed cross-validation to determine the robustness of our proposed technique using two prominent datasets, Pascal VOC and COCO. We attained excellent performance for our proposed model compared to state-of-the-art techniques, achieving 81% mAP. We believe that our proposed model is robust and an effective framework for vehicle detection such as for cars, buses, trucks, etc. In the future, we aim to modify and fine-tune our model to attain better accuracy and mAP for vehicle detection along with classification. Moreover, we will try to utilize our framework for other object detection applications such as abnormal activity detection.

Author Contributions: Conceptualization, S.M.L. and R.M.; methodology, F.S.B.; software, R.A.; validation, A.M.E.-S., S.M.L. and S.S.; formal analysis, M.J.A.; investigation, R.M.; resources, R.M.; data curation, R.A.; writing—original draft preparation, F.S.B.; writing—review and editing, A.M.E.-S.; visualization, S.M.L.; supervision, S.S.; project administration, M.J.A.; funding acquisition, A.M.E.-S. All authors have read and agreed to the published version of the manuscript.

Funding: The authors extend their appreciation to King Saud University for funding this work through researchers supporting project number (RSP-2021/133), King Saud University, Riyadh, Saudi Arabia.

Data Availability Statement: Data sharing does not apply to this article as authors have used publicly available datasets, whose details are included in the “experimental results and discussions” section of this article. Please contact the authors for further requests.

Acknowledgments: The authors extend their appreciation to King Saud University and UET Taxila for supporting this work.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Zhou, X.; Gong, W.; Fu, W.; Du, F. Application of deep learning in object detection. In Proceedings of the 2017 IEEE/ACIS 16th International Conference on Computer and Information Science (ICIS), Wuhan, China, 24–26 May 2017; pp. 631–634.
2. Taigman, Y.; Yang, M.; Ranzato, M.A.; Wolf, L. Deepface: Closing the gap to human-level performance in face verification. In Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition, Columbus, OH, USA, 23–28 June 2014; pp. 1701–1708.
3. Zhang, Z.; Zhang, C.; Shen, W.; Yao, C.; Liu, W.; Bai, X. Multi-oriented text detection with fully convolutional networks. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 4159–4167.
4. Chen, X.; Wei, P.; Ke, W.; Ye, Q.; Jiao, J. Pedestrian detection with deep convolutional neural network. In *Asian Conference on Computer Vision*; Springer: Cham, Switzerland, 2015; pp. 354–365.
5. Hoi, S.C.; Wu, X.; Liu, H.; Wu, Y.; Wang, H.; Xue, H.; Wu, Q. Logo-net: Large-scale deep logo detection and brand recognition with deep region-based convolutional networks. *arXiv* **2015**, arXiv:1511.02462.
6. Kang, K.; Li, H.; Yan, J.; Zeng, X.; Yang, B.; Xiao, T.; Zhang, C.; Wang, Z.; Wang, R.; Wang, X. T-cnn: Tubelets with convolutional neural networks for object detection from videos. *IEEE Trans. Circuits Syst. Video Technol.* **2017**, *28*, 2896–2907. [[CrossRef](#)]
7. Fan, Q.; Brown, L.; Smith, J. A closer look at Faster R-CNN for vehicle detection. In Proceedings of the 2016 IEEE Intelligent Vehicles Symposium (IV), Gotenburg, Sweden, 19–22 June 2016; pp. 124–129.
8. Litjens, G.; Kooi, T.; Bejnordi, B.E.; Setio, A.A.A.; Ciampi, F.; Ghahfarokian, M.; Van Der Laak, J.A.; Van Ginneken, B.; Sánchez, C.I. A survey on deep learning in medical image analysis. *Med. Image Anal.* **2017**, *42*, 60–88. [[CrossRef](#)]
9. Mahum, R.; Rehman, S.U.; Okon, O.D.; Alabrah, A.; Meraj, T.; Rauf, H.T. A novel hybrid approach based on deep CNN to detect glaucoma using fundus imaging. *Electronics* **2021**, *11*, 26. [[CrossRef](#)]

10. Sun, Z.; Bebis, G.; Miller, R. Monocular precrash vehicle detection: Features and classifiers. *IEEE Trans. Image Process.* **2006**, *15*, 2019–2034. [[PubMed](#)]
11. Bai, H.; Wu, J.; Liu, C. Motion and haar-like features based vehicle detection. In Proceedings of the 2006 12th International Multi-Media Modelling Conference, Beijing, China, 4–6 January 2006; p. 4.
12. Wei, Y.; Tian, Q.; Guo, J.; Huang, W.; Cao, J. Multi-vehicle detection algorithm through combining Harr and HOG features. *Math. Comput. Simul.* **2019**, *155*, 130–145. [[CrossRef](#)]
13. Qin-jun, Q.; Yong, L.; Da-wei, C. Vehicle detection based on LBP features of the Haar-like Characteristics. In Proceedings of the 11th World Congress on Intelligent Control and Automation, Shenyang, China, 29 June–4 July 2014; pp. 1050–1055.
14. Yang, W.; Zhang, J.; Wang, H.; Zhang, Z. A vehicle real-time detection algorithm based on YOLOv2 framework. In *Proceedings of the Real-Time Image and Video Processing 2018, Orlando, FL, USA, 21 May 2018*; International Society for Optics and Photonics: Bellingham, WA, USA, 2018; p. 106700N.
15. Ahmad, T.; Ma, Y.; Yahya, M.; Ahmad, B.; Nazir, S. Object detection through modified YOLO neural network. *Sci. Program.* **2020**, *2020*, 8403262. [[CrossRef](#)]
16. Feris, R.S.; Siddiquie, B.; Petterson, J.; Zhai, Y.; Datta, A.; Brown, L.M.; Pankanti, S. Large-scale vehicle detection, indexing, and search in urban surveillance videos. *IEEE Trans. Multimed.* **2011**, *14*, 28–42. [[CrossRef](#)]
17. Song, H.; Liang, H.; Li, H.; Dai, Z.; Yun, X. Vision-based vehicle detection and counting system using deep learning in highway scenes. *Eur. Transp. Res. Rev.* **2019**, *11*, 51. [[CrossRef](#)]
18. Tang, Y.; Zhang, C.; Gu, R.; Li, P.; Yang, B. Vehicle detection and recognition for intelligent traffic surveillance system. *Multimed. Tools Appl.* **2017**, *76*, 5817–5832. [[CrossRef](#)]
19. Hu, X.; Xu, X.; Xiao, Y.; Chen, H.; He, S.; Qin, J.; Heng, P.-A. SINet: A scale-insensitive convolutional neural network for fast vehicle detection. *IEEE Trans. Intell. Transp. Syst.* **2018**, *20*, 1010–1019. [[CrossRef](#)]
20. González, J.L.S.; Zaccaro, C.; Álvarez-García, J.A.; Morillo, L.M.S.; Caparrini, F.S. Real-time gun detection in CCTV: An open problem. *Neural Netw.* **2020**, *132*, 297–308.
21. Sang, J.; Wu, Z.; Guo, P.; Hu, H.; Xiang, H.; Zhang, Q.; Cai, B. An improved YOLOv2 for vehicle detection. *Sensors* **2018**, *18*, 4272. [[CrossRef](#)]
22. Wu, Z.; Sang, J.; Zhang, Q.; Xiang, H.; Cai, B.; Xia, X. Multi-scale vehicle detection for foreground-background class imbalance with improved YOLOv2. *Sensors* **2019**, *19*, 3336. [[CrossRef](#)]
23. Dai, X. HybridNet: A fast vehicle detection system for autonomous driving. *Signal Process. Image Commun.* **2019**, *70*, 79–88. [[CrossRef](#)]
24. Dong, Z.; Wu, Y.; Pei, M.; Jia, Y. Vehicle type classification using a semisupervised convolutional neural network. *IEEE Trans. Intell. Transp. Syst.* **2015**, *16*, 2247–2256. [[CrossRef](#)]
25. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. Imagenet classification with deep convolutional neural networks. *Adv. Neural Inf. Process. Syst.* **2012**, *25*, 84–90. [[CrossRef](#)]
26. Targ, S.; Diogo, A.; Lyman, K. Resnet in resnet: Generalizing residual architectures. *arXiv* **2016**, arXiv:1603.08029.
27. Szegedy, C.; Liu, W.; Jia, Y.; Sermanet, P.; Reed, S.; Anguelov, D.; Erhan, D.; Vanhoucke, V.; Rabinovich, A. Going deeper with convolutions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 1–9.
28. Simonyan, K.; Zisserman, A. Very deep convolutional networks for large-scale image recognition. *arXiv* **2014**, arXiv:1409.1556.
29. Zeng, N.; Wang, Z.; Zhang, H.; Liu, W.; Alsaadi, F.E. Deep belief networks for quantitative analysis of a gold immunochromatographic strip. *Cogn. Comput.* **2016**, *8*, 684–692. [[CrossRef](#)]
30. Deng, J.; Dong, W.; Socher, R.; Li, L.-J.; Li, K.; Li, F.F. Imagenet: A large-scale hierarchical image database. In Proceedings of the 2009 IEEE Conference on Computer Vision and Pattern Recognition, Miami, FL, USA, 20–25 June 2009; pp. 248–255.
31. Krizhevsky, A. Learning Multiple Layers of Features from Tiny Images. Available online: <https://www.cs.toronto.edu/~kriz/learning-features-2009-TR.pdf> (accessed on 13 May 2022).
32. Huang, G.; Liu, Z.; Van Der Maaten, L.; Weinberger, K.Q. Densely connected convolutional networks. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 4700–4708.
33. Redmon, J.; Divvala, S.; Girshick, R.; Farhadi, A. You only look once: Unified, real-time object detection. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 779–788.
34. Vehicle Data Set. Available online: <https://www.kaggle.com/datasets/iamsandeepprasad/vehicle-data-set> (accessed on 12 April 2020).
35. Kitti. Available online: http://www.cvlibs.net/datasets/kitti/eval_object.php (accessed on 10 May 2012).
36. Lin, T.-Y.; Maire, M.; Belongie, S.; Hays, J.; Perona, P.; Ramanan, D.; Dollár, P.; Zitnick, C.L. Microsoft coco: Common objects in context. In *Proceedings of the European Conference on Computer Vision, Zurich, Switzerland, 6–12 September 2014*; Springer: Cham, Switzerland, 2015; pp. 740–755.
37. Everingham, M.; Van Gool, L.; Williams, C.K.; Winn, J.; Zisserman, A. The pascal visual object classes (voc) challenge. *Int. J. Comput. Vis.* **2010**, *88*, 303–338. [[CrossRef](#)]
38. Munir, M.H.; Mahum, R.; Nafees, M.; Aitazaz, M.; Irtaza, A. An Automated Framework for Corona Virus Severity Detection Using Combination of AlexNet and Faster RCNN. *Control. Syst. Eng.* **2022**, in press.

39. Mahum, R.; Rehman, S.U.; Meraj, T.; Rauf, H.T.; Irtaza, A.; El-Sherbeeney, A.M.; El-Meligy, M.A. A novel hybrid approach based on deep cnn features to detect knee osteoarthritis. *Sensors* **2021**, *21*, 6189. [[CrossRef](#)]
40. Mahum, R.; Irtaza, A.; Nawaz, M.; Nazir, T.; Masood, M.; Shaikh, S.; Nasr, E.A. A robust framework to generate surveillance video summaries using combination of zernike moments and r-transform and deep neural network. *Multimed. Tools Appl.* **2022**. [[CrossRef](#)]
41. Mahum, R.; Munir, H.; Mughal, Z.U.; Awais, M.; Sher Khan, F.; Saqlain, M.; Mahamad, S.; Tlili, I. A novel framework for potato leaf disease detection using an efficient deep learning model. *Hum. Ecol. Risk Assess. Int. J.* **2022**, *11*, 1–24. [[CrossRef](#)]
42. Girshick, R. Fast r-cnn. In Proceedings of the 2015 IEEE International Conference on Computer Vision, Santiago, Chile, 7–13 December 2015; pp. 1440–1448.
43. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778.
44. Liu, W.; Anguelov, D.; Erhan, D.; Szegedy, C.; Reed, S.; Fu, C.-Y.; Berg, A.C. Ssd: Single shot multibox detector. In Proceedings of the 2016 European Conference on Computer Vision, Amsterdam, The Netherlands, 11–14 October 2016; pp. 21–37.
45. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster r-cnn: Towards real-time object detection with region proposal networks. *Adv. Neural Inf. Process. Syst.* **2015**, *28*, 91–99. [[CrossRef](#)]
46. Sri Jamiya, S. An Efficient Algorithm for Real-Time Vehicle Detection Using Deep Neural Networks. *Turk. J. Comput. Math. Educ. TURCOMAT* **2021**, *12*, 2662–2676.
47. Redmon, J.; Farhadi, A. YOLO9000: Better, faster, stronger. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 7263–7271.