



JUVDsi v1: developing and benchmarking a new still image database in Indian scenario for automatic vehicle detection

Avirup Bhattacharyya¹ · Avigyan Bhattacharya¹ · Sourajit Maity¹ · Pawan Kumar Singh²  · Ram Sarkar¹

Received: 9 November 2021 / Revised: 11 October 2022 / Accepted: 3 February 2023 /
Published online: 2 March 2023

© The Author(s), under exclusive licence to Springer Science+Business Media, LLC, part of Springer Nature 2023

Abstract

Designing an automatic vehicle detection (AVD) system from still images or videos would be a useful tool to cater to the requirements of the traffic management system. Over the past few years, numerous databases have been developed for the use of researchers in this field of AVD. However, most of them are not acceptable in the Indian scenarios due to certain practical constraints like the road infrastructure, nature of congestion, and vehicle types commonly found in India. The aim of this research is to develop a still image database, named as *JUVDsi v1*, which includes nine different types of vehicle classes collected through mobile phone cameras in various ways for designing an automated traffic management system. Identifying and analyzing the shortcomings of existing databases, the developed database presents an improvement to address such bottle-necks. Furthermore, the efficiency of this database is evaluated using an ensemble of three state-of-the-art deep learning architectures. At first, each vehicle in the scene images is localized and categorized. Five base object detection models, namely, YOLOv3, Faster-RCNN, RFCN, SSDv1 and SSDLitev2 are used. Finally, the Weighted Boxes Fusion technique is used as the ensemble method (ensemble of best three out of the five base learners), thereby enhancing the performance obtained by the individual object detection models. The database can be found at: <https://github.com/JUVDsi/JUVD-Still-Image-database.git>.

Keywords Automatic vehicle detection · JUVD · Image database · Object detection · Deep learning · Ensemble

Avirup Bhattacharyya, Avigyan Bhattacharya and Sourajit Maity contributed equally to this work.

✉ Pawan Kumar Singh
pawansingh.ju@gmail.com

Extended author information available on the last page of the article

1 Introduction

Nowadays, a continuous increase in the number of vehicles on roadways emphasizes the need for automatic vehicle detection (AVD) and traffic information systems for real-time traffic monitoring and management. This requires the collection of accurate data, efficient vehicle detection techniques, and real-time traffic information in case of emergencies like car accidents that lead to traffic congestion and re-routing. Such systems can also be used in number plate detection and in collecting information about vehicles' manufacturer [27] and model [20] numbers which are required for security purposes.

Vehicle Make and Model Recognition (VMMR) [24] can be used in vehicle reidentification, vehicle monitoring, automated toll collection systems, etc. because it can recognize the maker as well as model of a vehicle from an image of a road scene. But accurate implementation of VMMR suffers from limitations due to the similar appearance of various vehicle models, and similarities among vehicles of different classes. On the other hand, appearances of same class of vehicles may change over time adding to the limitations of VMMR [38]. However, the concept of VMMR helps in streamlining the traffic management process by distinguishing heavy vehicles from lighter ones. Naseer et al. [29] have used the pre-trained VGG16 architecture to extract features, followed by feature reduction using Genetic Algorithm and finally, the Support Vector Machine (SVM) for classification purposes. Nazemi et al. [30] have encoded the Scale Invariant Feature Transform (SIFT) features for the input using Locality-constrained Linear Coding (LLC) method followed by classification. The You Only Look Once (YOLO) v5 model [55] is used by Wang et al. [46] for VMMR. Fig. 1 shows the pictorial difference between the AVD and VMMR problems.

AVD by human beings relies on license plates which may also suffer from some disadvantages. License plates can easily be damaged, modified, and re-created. Additionally, the text contained in some number plates can also be ambiguous. Distinguishing between '0' and 'o' or 'B' and '8' etc. are difficult. Automatic License Plate Recognition (ALPR) [16] systems may fail to correctly detect the actual number plates and might retrieve incorrect data to the database of make and model information. Therefore, traditional vehicle classification systems that rely on manual human observations or ALPR [6] systems, can hardly resolve the real-time constraints because both approaches have some limitations. Firstly, it is practically difficult for human observers to remember and efficiently distinguish among the wide variety and large number of vehicle makes and models. Secondly, it is very time-consuming and error-prone task for any human being. Fig. 1 shows the pictorial differences between the AVD and VMMR problems.

There are an enormous amount of publicly available databases [14, 18] in the field of AVD which are concerned about the diversity of object appearance, articulated objects, and number of categories without considering the effects of target size, image noise, and multispectral image technology. Also, there exist some databases [34, 43] that featured only one vehicle at a time. Therefore, all these databases can hardly be applied in case of Indian roads because of some characteristics that are commonly found in India or to be more specific in Indian sub-continent which are as follows:

- Absence of separate lanes for each kind of vehicle class on the road, e.g. vehicles of varied types use the same road at a time with a great speed mismatch, as in case of a bicycle and a car.
- Separate footpaths for pedestrians are absent in most of the roads. Footpaths are majorly occupied by roadside stalls and markets.

Fig. 1 Pictorial representation of: **a** AVD and **b** VMMR problems



- Poor condition of the roads e.g. roads are often damaged and muddy.
- Most of the roads in the semi-urban and rural areas are not well-lit during nighttime.
- Traffic signs are hardly found except in some metropolitan cities.
- People disobey traffic rules. Traffic police are not always available at many places. Traffic management system is handled by a combination of automated and manual method.

1.1 Contributions

This study aims to present *JUVDSi v1: Jadavpur University Vehicle detection still images, version 1*, a new still image database with proper annotation consisting of commonly found vehicles in Indian roads, specially designed considering the aforementioned characteristics of Indian roads. The data has been collected from various roads in Kolkata, one of the biggest metropolitan city in eastern India. Being one of the busiest and most populated cities of India, Kolkata experiences heavy traffic congestion, which makes it ideal to portray the traffic scenario of the Indian sub-continent as close to actual as possible. The main contributions of this work are as follows:

- The image database is properly annotated to measure the performance of any algorithm developed for automatic localization and classification of vehicles in an unconstrained environment.
- Different complexities are added to the images to make it challenging as well as realistic. The vehicles in the database, in addition to being small, exhibit different variabilities such as multiple orientations, single vehicle in one frame, multiple vehicles in one frame, changes in light conditions, specularities or occlusions.

- Nine different classes of vehicles are presented in this database – very few of the existing databases have considered such a variety of vehicle classes.
- Furthermore, each image is available in several spectral bands and resolutions. A precise experimental protocol is also given, ensuring that the experimental results obtained by different algorithms can be properly reproduced and compared.
- The results on this database are also evaluated using some state-of-the-art deep learning models. In doing so, three models namely, You Only Look Once (YOLO) [42], Region-Based Convolutional Neural Networks (R-CNN) [33], and Region-Based Fully Convolutional Networks (RFCN) [15] are used. Finally an ensemble method, called Weighted Boxes Fusion (WBF) [41] is implemented using the three models as base learners to illustrate the difficulties of the task and provide some results.
- To the best of our knowledge, application of these three base learners together as object detectors with the said ensemble approach is done for the first time in the domain of ADV using still images.

The remainder of this paper is written in the following structure: Section 2 presents state-of-the-art deep learning based approaches developed by the researchers to solve the problem of AVD as well as shortcomings of the existing still image databases. Section 3 details the database preparation, compilation and annotation of vehicle database and also introduces the pre-processing step that is done before proceeding to training whereas Section 4 describes the proposed database in detail. Section 5 illustrates the methods used to benchmark the proposed JUVDSi v1. Section 6 reports the results and discussion which leads to experimental evaluation of benchmarking vehicles database along with some limitations of the proposed database. Finally, the conclusion of our work is provided in Section 7.

2 Related work

AVD is often considered as one of the most challenging computer vision tasks, which has been long in the history of literature. This section discusses about the databases, that are publicly available to develop, validate and compare vehicle detectors, different ways to measure the performance of an algorithm, and the state-of-the-art approaches available for object detection. The databases which are available privately are found to be very costly, whereas the publicly available ones are being used frequently over the years by the researchers to get them almost close to 100% accuracy. Table 1 provides a summary of some standard AVD databases commonly used by the researchers.

2.1 Existing deep learning based methods for AVD

Liu et al. [26] studied the effects of large variance of scales in complex traffic scenes and presented Faster RCNN based two-stage detectors for generic object detection. But it has rarely been effective on vehicle detection in real applications, especially for vehicles of tiny scales. Another novel two-stage detector is proposed to perform tiny vehicle detection with high recall which consists of a backward feature enhancement network (BFEN) and a spatial layout preserving network (SLPN).

Although there have been a large number of studies exploring various types of deep learning methods for vehicle detection, there are a very few studies that compare and evaluate

Table 1 Databases for AVD commonly used in literature

Database name	Input type (image/video)	Number of vehicle classes	Purpose	Number of images	Availability	Folds
LISA [39]	Video	9	Localization	1200	Free	Train
VEDAI [32]	Image	9	Classification	4656	Free	Train/Test
iROADS [34]	Image	Multiclass	Localization	215,986	Free	Train
EAGLE Dataset [4]	Image	Multiclass	Classification	14,999	Paid	Train
KITTI [18]	Images	–	Localization	100	Free	Train/Test
DETRAC [47]	Video	4	Classification	200,000	Paid	Train/Test
BOXY Vehicle Detection dataset [7]	Images	2	Classification	30,000	Free	Train/Test
TME Motorway Dataset [9]	Video	–	Localization	1328	Free	Train
UIUC Image Database for car detection [1]	Images	–	Localization	9850	Free	Train/Test
BIT Vehicle Dataset [37]	Images	6	Classification	3475	Free	Train
Cityscapes Image Pairs [14]	Images/Videos	–	Localization	4000	Free	Train/Val
GTI Vehicle Image database [3]	Images	–	Localization	113,888	Free	Train/Val
LISA Traffic Light Dataset [40]	Images	–	Localization	4800	Free	Train/Test
Nepalese Vehicles [17]	Images/Videos	2	Classification	130,000	Free	Train
Rain and snow Traffic Surveillance [5]	Image	–	Localization	16,185	Free	Train
Stanford Cars Dataset [23]	Image	196	Classification	5000	Free	Train/Test
Semantic Segmentation of Self Driving cars [35]	Image	–	Localization	4006	Free	Train
ACDC [36]	Image	–	Segmentation	11,211	Free	Train/Test/ Validation
D2-City [10]	Videos	12	Segmentation and Tracking	1000 scenes	Free	Train/Test/ Validation
nuScenes [8]	Videos	23	Localization and Tracking	143,906	Paid	Train
ApolloScape [22]	Image	8	Segmentation and Tracking	1800 frames	Free	Train/Test
WildDash [52]	Videos	–	Segmentation	2500 panoramas	Free	Test
WildPASS [49]	Videos	–	Segmentation	12,500 frames, 313 various accident sequences	Free	Test
ISSAFE DADA-seg [53]	Images/videos	–	Segmentation	10,004	Paid	Evaluation
IDD [43]	Images	34	Segmentation	100,000	Free	Train/Test/ Validation
Berkeley Deepdrive BDD100K [51]	Videos	–	Localization	–	Free	Train

the detection time and detection accuracy of the mainstream deep learning object detection algorithms for vehicle detection. A study by Wang et al. [44] compared five independent deep learning target detection algorithms in road vehicle detection namely the faster R-CNN, R-FCN, Single Shot Detector (SSD), RetinaNet, and YOLOv3 and trained each model using the same KITTI [18] public database assigned to the same training scale. Three indicators are used to compare the comprehensive performance of the algorithms: (1) the recall rate and precision rate on the KITTI test set; (2) the average precision on the KITTI [18] test set; (3) the frame per second (fps). In addition, this work also used various models to detect vehicles in a real road scene to make an intuitive judgement on the applicability of each target detection algorithm for road vehicle detection.

Wu et al. [48] presented a multi-camera vehicle detection system that used a novel multi-view region proposal network to localize the candidate vehicles on the ground plane. This method can tell the position of the vehicle on the ground plane by leveraging multi-view cross camera context and also detect partially and severely occluded vehicles in field traffic scenes.

Yang et al. [50] employed a deep convolutional network in order to obtain high-performance object detection and a data association-based multiple target tracking method is implemented by combining the appearance feature and the estimation of the motion state through the Kalman filter. The experimental results showed that the proposed method can reduce the number of identity switches and complete the detection and tracking of surface vehicles more accurately as compared to SORT (Simple Online and Real-time Tracking) algorithm. Also, this method has better flexibility and can be easily adapted to different situations by using different object detection frameworks.

Appathurai et al. [2] presented an effective traffic video surveillance system for detecting moving vehicles in traffic scenes. They developed a novel hybridization of ANN and oppositional gravitational search optimization algorithm (OGSA) based moving vehicle detection system. This consisted of two main phases: background generation and vehicle detection. After the background generation, the moving vehicle is detected using the ANN–OGSA model. The vehicles in a traffic surveillance video can be counted using different deep learning techniques designed by Mirthubasini and Santhi [28]. Vehicles are detected and multi-object tracking is done to perform AVD and localization. The CNN model can improve performance of detection and tracking. Finally, the objects have been counted only in the ROI. A study by Nguyen et al. [31] proposed an improved framework based on Faster R-CNN for fast vehicle detection. Firstly, Mobile Net architecture has been adopted to build the base convolution layer in Faster R-CNN. Then, NMS (Non-Maximum Suppression) algorithm after the region proposal network in the original Faster R-CNN has been replaced by the soft-NMS algorithm to solve the issue of duplicate proposals. Next, context-aware ROI pooling layer has been adopted to adjust the proposals to the specified size without scarifying important contextual information.

In the work by Wang et al. [45], reinforcement learning (RL) and multi-agent reinforcement learning (MARL) are applied to drive agents to generate samples, and use this to replace learning samples generated by expert experience. In addition, the deep Q-learning network (DQN) mechanism is applied in the agent training method to avoid dimensionality, and finally, multi-agent deep reinforcement learning (MDRL) is applied. This algorithm integrated the fuzzy calculation process that must be completed on MATLAB into the neural network during training, and this can greatly improve the calculation speed. Through the simulation experiments using python, it proved that this method can improve the accuracy of the algorithm up to about 96% and perform the best delay level.

Chen et al. [11] have proposed an improved vehicle detection algorithm based on SSD to improve accuracy, especially for detection of small vehicles. In this method, an inception block is added to the extra layer in the SSD before the prediction to improve its performance. Then a more suitable method for AVD is implemented to set the scales and aspect ratios of the default bounding boxes, which improves position regression and maintained the fast speed. The validity of their algorithm is verified on KITTI and UVD datasets. Compared to SSD, the algorithm achieved a higher mean average precision (mAP), while maintaining a fast speed.

2.2 Shortcomings of the existing still image databases

It is already evident from Table 1 that most of the available AVD databases considered a maximum of 2 or 3 vehicle classes under different environmental conditions such as daytime, night time, rainy season etc. Most of the previously developed databases are proposed for vehicle localization only, and very few of these have annotations. But for vehicle detection problem, only a few databases like VEDAI are available which may create biasness while training the model. Therefore, the scarcity of appropriate databases necessitates the need for such an AVD that can solve both localization as well as classification problems. Therefore, the primary motivation of this study is to develop a new database based on still images considering 9 different vehicle classes mostly seen on Indian roads under different environmental conditions. Though both VEDAI and JUVDSi v1 have 9 vehicle classes, *JUVDSi v1* considers 3000 images of exclusively on-road vehicles which are commonly found in India whereas, VEDAI considered 1200 images which included flight, train and boat etc. In Table 2 shows a detailed comparison of our developed database with some popularly used still image based AVD databases in literatures.

3 Database preparation

In the following subsections, we have discussed the database nomenclature, method of collection of data, preparations of images from video data, and annotations.

3.1 Database nomenclature

Our developed database is named as *JUVDSi v1*, where *JUVD* stands for ‘Jadavpur University Vehicle Detection’, and *si* stands for still image. The authors of this group belong to the research laboratory CMATER, at Computer Science and Engineering Department of Jadavpur

Table 2 Comparative analysis of our developed AVD database, *JUVDSi v1*, with some popular databases in terms of different aspects (still image only)

Database name	Number of images	Number of classes	Purpose	Availability	Train/Test
BOXY Vehicle Detection dataset [7]	200,000	N/A	Localization	Free	✓
VEDAI [32]	1200	9	Localization	Free	✓
iROADS [34]	4656	N/A	Localization	Free	✓
<i>JUVDSi v1</i> (current)	3000	9	Localization and Classification	Free	✓

Table 3 Sample image of different vehicle classes considered in *JUVDsi* v1 (digits in the caption indicate their class labels)

		
0: Car	1: Bus	2: Motorbike
		
3: Bicycle	4: Truck	5: Autorickshaw
		
6: Rickshaw	7: Van	8: Minitruck

University, Kolkata, India, where the database is prepared. Currently, we have developed version 1 of *JUVDsi* database. It is to be noted that the database has 9 different vehicle classes namely, ‘Car’, ‘Bus’, ‘Motorbike’, ‘Cycle’, ‘Truck’, ‘Autorickshaw’, ‘Rickshaw’, ‘Van’ and ‘Minitruck’. A sample image of each of the vehicle classes along with their class names and class labels are illustrated in Table 3. The database is made available freely at: <https://github.com/JUVDsi/JUVD-Still-Image-database.git> which can be used for the research purposes only.

3.2 Database collection

The data is collected from roads of Kolkata, India which represents real-time traffic scenarios as far as possible. Initially, videos of real-time traffic are taken and then image frames are extracted to prepare the still image by labelImg [35].

JUVDsi v1 includes images from a fixed position as well as from a moving vehicle during daytime and night time, and also images of overlap of multiple vehicles within one bounding box. This database incorporates traffic scenarios on Indian urban roads as realistic as possible.

Capturing devices: Three different mobile phone cameras are used to capture the video and still images:

- Redmi Note 7 s (1280x720p)

- Samsung Galaxy M51 (2400x1080p)
- Honor - HRY-AL00 (1080x2340p)

3.3 Video and image capturing protocols

1. Videos are captured from the footpath and from moving vehicles in portrait mode of the mobile phone camera.
2. While capturing video from footpath, an angle of 30° to 45° is tried to be maintained between the length of the road and camera.
3. Videos are recorded in two ways - making a positive angle and then a negative angle with the perpendicular of the road such that both the rear and front ends of the vehicle can be covered.
4. While taking still images, the vehicles are captured from a short distance.

3.4 Extracting frames from the captured videos

Each video is divided into image frames, taking one out of every 15 frames. The image frames are saved in JPEG file format after resizing each image into a fixed window size for easy processing by the image processing algorithms. The steps of this procedure are as follows:

1. Frames are cropped in such a way that the vehicles cover at least 30% and at most 70% of the image.
2. Now, image frames that are not very hazy, i.e., understandable and are noticeably different from any other frame in the set of selected frames, are selected.
3. The still images are resized to 640x320p for training purposes.

All the images, selected from video frames and the resized still images are randomly split into training set (70%) and test set (30%). The programmer can split the training data in training and validation set as per requirements.

3.5 Annotation of images

To train the object detectors using any machine learning or deep learning algorithm, the class label of the vehicles and the bounding box coordinates of each such vehicle are needed to be specified. Thus, proper annotation is an important requirement of the developed database. Besides, having annotation in the test set images would help the researchers for performance assessment when they will develop a new algorithm. For training a YOLOv3 model, five elements per object must be specified during data annotation which are: the class ID, X-coordinate of the centre of box, Y-coordinate of the same, width of box and height of box. This format of annotation has been given in TXT format. Some models require coordinate of left top corner of rectangle and coordinate of right bottom corner along with class ID. XML format contains this type of annotation. The selected images have been annotated using a standard tool labelImg tool [35]. Fig. 2 shows the annotation done on sample images using the said tool. It is to be noted that annotation information is stored in XML and TXT formats both. The format of the annotations can be found in the GitHub link where database is provided.

Annotations in TXT format for Fig. 2a are mentioned below:

Class label	X-coordinate of centre of box (normalized)	Y-coordinate of centre of box (normalized)	Width of box (normalized)	Height of box (normalized)
2	0.454688	0.407813	0.462500	0.803,125

Annotations in XML format:

```

<annotation>
  <folder>Cloudy</folder>
  <filename>cloudy (1).jpeg</filename>
  <path>E:\Dataset\Train\Daylight\Cloudy\cloudy (1).jpeg</path>
  <source>
    <database>Unknown</database>
  </source>
  <size>
    <width>640</width>
    <height>320</height>
    <depth>3</depth>
  </size>
  <segmented>0</segmented>
  <object>
    <name>Motorbike</name>
    <pose>Unspecified</pose>
    <truncated>0</truncated>
    <difficult>0</difficult>
    <bndbox>
      <xmin>143</xmin>
      <ymin>2</ymin>
      <xmax>439</xmax>
      <ymax>259</ymax>
    </bndbox>
  </object>
</annotation>

```

4 Database detail

The images in the database portray a typical Indian road scenario that can be used for developing a realistic AVD system. The images are captured during different times of the day and in different environmental conditions. During daytime, images and videos are captured in sunny as well as in cloudy weather. The videos and images, captured during the



Fig. 2 Sample frames taken from our database annotated using LabelImg tool



Fig. 3 Sample images taken from our database in sunny condition

night are all under clear skies. The difficulty in terms of classification of objects in the images varies in the database. There are images with only one object and also with multiple objects belonging to different vehicle classes in a single frame. The database contains some very complex images in both training and test sets with multiple objects of various classes which is a common sight in day-to-day Indian traffic. Every possible road and traffic scenarios are covered in this database including the congestion on the roads as well as various road conditions that are present in rural and urban areas in and around Kolkata, India. The videos are captured both from the side of the road and also while travelling on a motorbike, thus helping the researchers to make their models trained on the database robust to the angle of capture. The following images present the images captured during daytime (Fig. 3), night time (Fig. 4) and under cloudy conditions (Fig. 5) respectively.

4.1 Training set

The training set of the database, called *JUVDsi v1*, comprises 872 images captured in sunny weather conditions, 651 in cloudy conditions and 565 images captured during the night (clear skies). The number of objects of each class in the training data is shown in Fig. 6.

There are 629 images with only a single object in the entire image and 1459 images with multiple objects in the entire image. In Fig. 7, a pie chart represents the number of images with a certain number of objects in a single frame. The number of images in a single frame in the training set varies from 1 to 4. High complexity images of more than 4 objects in one frame are also present. Fig. 8 is a bar graph showing the distribution of the number of objects of each class in the training set. Here, the number of images is along Y-axis and the number of objects of a class is along X-axis.

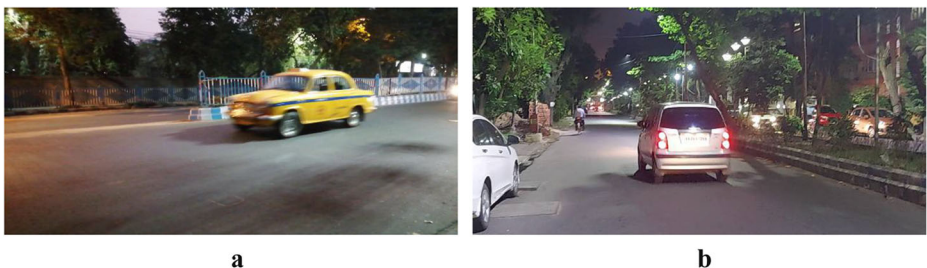
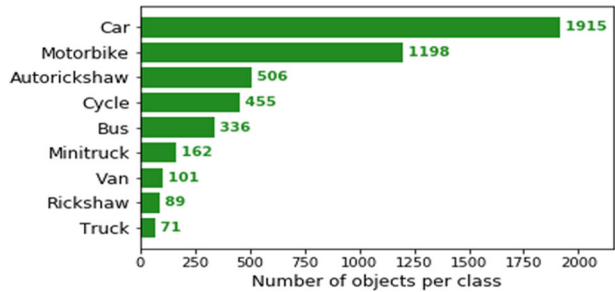


Fig. 4 Sample images taken from our database during night condition



Fig. 5 Sample images taken from our database in cloudy condition

Fig. 6 Graph showing the number of vehicles considered in the training set

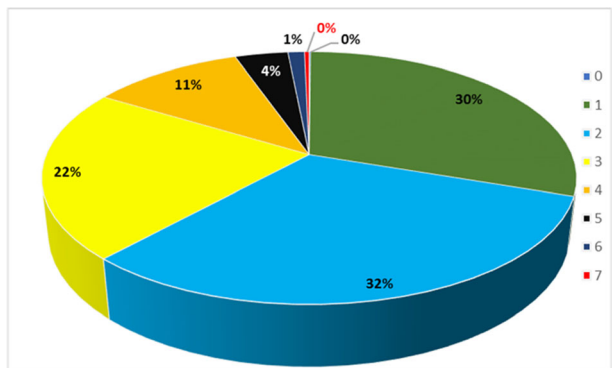


4.2 Test set

The test set of the database developed under the scope of the present work comprises a total of 913 images including 367 images captured in sunny weather conditions, 280 in cloudy conditions, and 266 images captured during the night (clear skies). The number of objects of each vehicle class in the test data has been shown in Fig. 9.

There are 211 images with single object in the entire image and 697 images with multiple objects in the entire image. Fig. 10 shows a pie chart depicting the number of objects in a frame in the test set of *JUVDsi v1* database. The distribution is similar to the one in the training set. Mainly there are images with object count ranges from 1 to 4. Complex images are present in the test set too.

Fig. 7 Pie chart showing the number of vehicles in a single frame in the training set of *JUVDsi v1* database



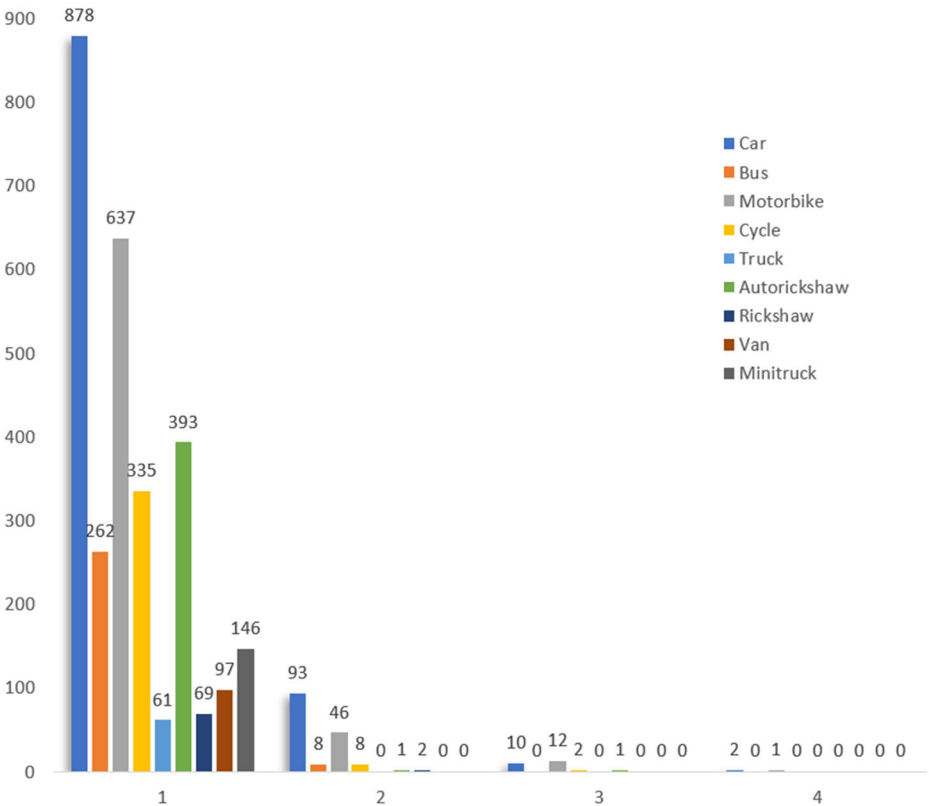


Fig. 8 Bar graph showing the distribution of the number of each vehicle class in the entire training set. For example, the number of images containing only one car is 878, the number of images containing two cars is 354

Fig. 9 Graph showing the number of different vehicle images considered in the test set of *JUVDisi v1* database

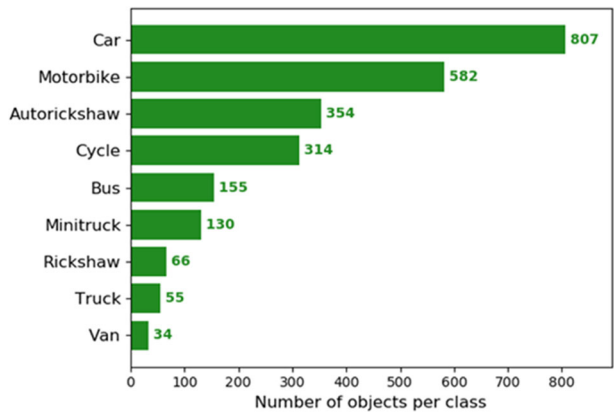


Fig. 10 Pie chart showing the number of vehicles in a single frame in the test set of *JUVDesi v1* database

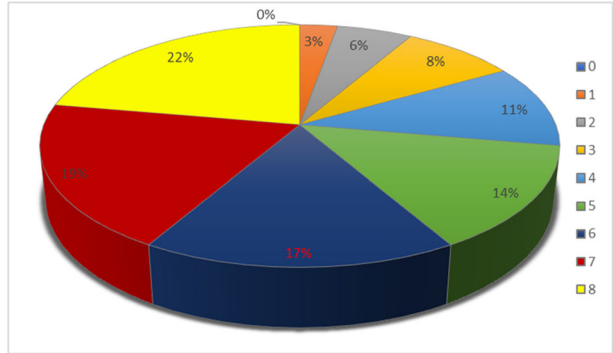


Figure 11 is a bar graph showing the distribution of the number of objects of each class in the test set. Here, the number of images is along Y-axis and the number of objects of a class is along X-axis. The number of cars in one image varies from 1 to 5. The number of images containing one car only is 381 while 145 images contain two cars while only 1 image contains 5 cars in it. This statistic is variable for each class as seen from the graph.

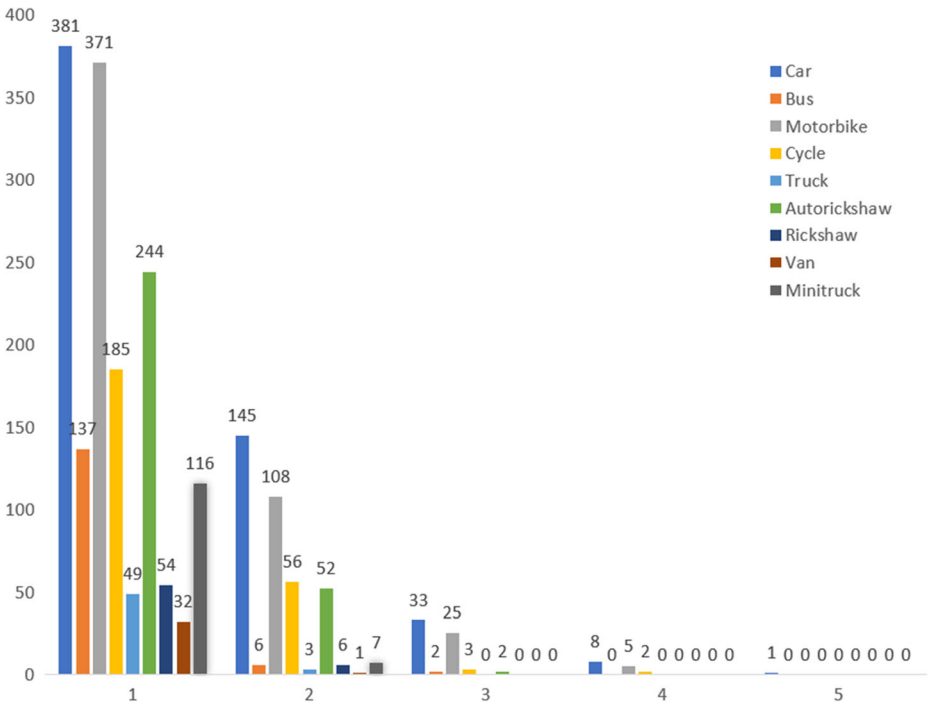


Fig. 11 Bar graph showing the distribution of the number of each vehicle class in the entire test set. For example, the number of images containing only one bus is 137, the number of images containing two cars is 6

5 Methods used to benchmark *JUVDisi v1*

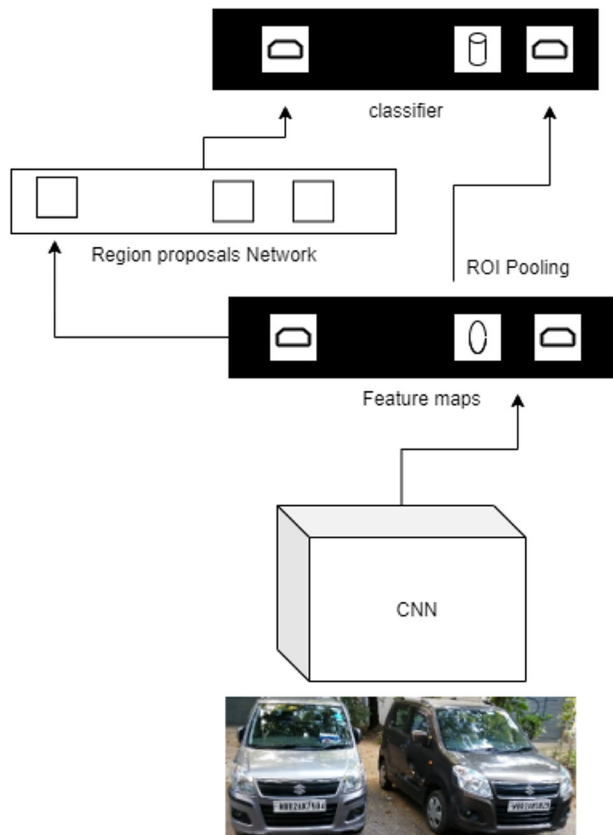
We have concentrated on the deep learning approaches to localize and classify the vehicles on our database. The deep learning methods usually require a large amount of data and learn the characteristics on their own which reflects the difference in data. Compared to other machine learning algorithms and traditional methods, deep learning methods have achieved real-time competitive performance and variants of deep learning models aim on exceeding the past results in a particular domain.

Initially, multiple deep learning object detection models are trained and their results are reported. Based on the results the best three models are chosen on the basis of accuracy on the validation set (created by randomly splitting training set in 80% for training and 20% for validation). The prediction scores of these models are used as input to an ensemble method. The ensemble obtained better detection results than the base models. The three chosen base models and the ensemble model are discussed briefly in the following subsections.

5.1 Faster RCNN

The process of a Convolutional Neural Network (CNN) based deep feature extraction model is very similar to the human visual mechanism. Regional-CNN (RCNN) [33] extracts a bunch of regions from the given image using selective search, which is a region proposal algorithm,

Fig. 12 Architecture of faster R-CNN used in the present work



based on computing hierarchical grouping of similar regions based on color, texture, size and shape compatibility. Then it uses a CNN model to extract specific features which are finally used to detect objects. Fast R-CNN, (see the above reference) on the other hand, passes the entire image to ConvNet which generates regions of interest. It uses a single model which extracts features from the regions, classifies them into different classes, and returns the bounding boxes. Faster R-CNN fixes the problem of selective search by replacing it with Region Proposal Network (RPN). It first extracts feature maps from the input image using ConvNet and then passes those maps through a RPN which returns object proposals. Finally, these maps are classified and the bounding boxes are predicted. We have used Faster R-CNN method (refer to Fig. 12 for more detail) and trained it using Tensorflow API.

5.2 YOLO v3

YOLO is a real-time object detection algorithm and is one of the fastest object detection algorithms. YOLO v3 [42] uses the custom deep architecture Darknet-53. Originally, a 53-layer network was trained on Imagenet and another 53 layers are stacked onto it for detection. Thus, there is a 106-layer fully convolutional underlying architecture for YOLO v3. YOLO v2 [37] uses a custom deep architecture Darknet-19. Here, 11 layers are stacked on it, thus giving 30 fully connected convolutional layers. But it lacks upsampling, residual blocks and skip connections. YOLO v3 architecture contains all of these. Here, 1×1 detection kernels are applied on feature maps of three different sizes at three different places in the network for detection. YOLO v3 uses 9 anchor boxes.

Shape of detection kernel = $1 \times 1 \times ((num_of_classes + 5) \times 3)$

In our database in case of 9 different classes shape is $1 \times 1 \times 42$.

Prediction is made at three scales. In the 82nd layer, the first detection is made, followed by second detection in the 94th layer and the final detection is made in the 106th layer. Detections in multiple layers help in detecting small objects. Fig. 13 shows the overall architecture of our YOLO v3 method.

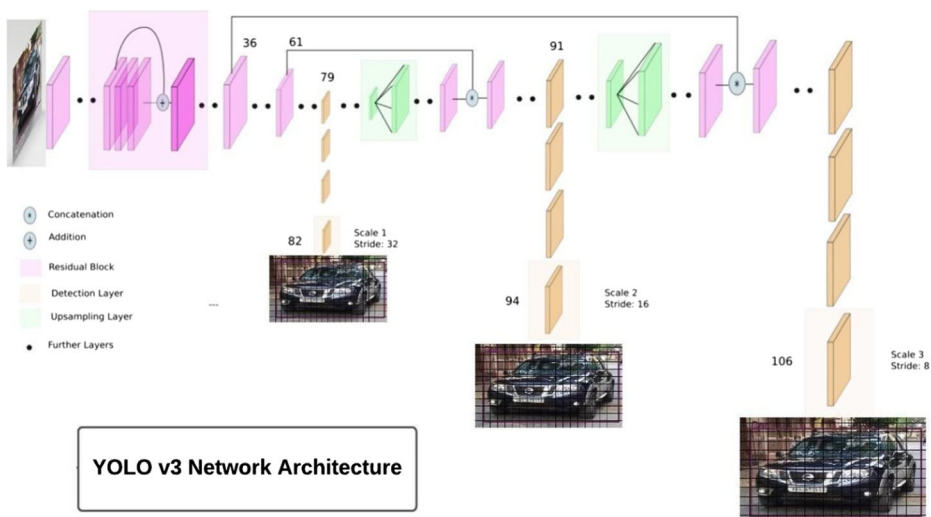


Fig. 13 Schematic diagram representing the YOLO v3 framework

5.3 Region-based fully convolutional network

For traditional RPN approaches such as R-CNN, Fast R-CNN and Faster R-CNN, region proposals are generated by RPN first. Then Region of Interest (RoI) pooling is used, and it goes through the fully connected (FC) layers for classification and bounding box regression. The process of using FC layers after ROI pooling does not share among ROI, and takes time, which makes RPN approaches slow. Besides, the FC layers increase the number of connections (parameters) which also increase the complexity of the model. In R-FCN [15], we still have RPN to obtain region proposals, but unlike R-CNN series, FC layers after ROI pooling are removed. Instead, all major complexities are moved before ROI pooling to generate the score maps. All region proposals, after ROI pooling, will make use of the same set of score maps to perform average voting, which is a simple calculation. Thus, there is no learnable layer after ROI layer which is nearly cost-free. As a result, R-FCN is even faster than Faster R-CNN with competitive mAP. The architecture of our R-FCN method is illustrated in Fig. 14.

5.4 Weighted boxes fusion (WBF)

To boost the accuracy of the overall AVD system, we have tried to ensemble the three base models. We have tested other techniques like NMS and soft NMS. They have produced better results than the base learners. But the WBF has obtained the best mAP [36] score on validation data among the ensemble methods. So we have benchmarked the results using WBF method. WBF [41] method combines predictions of different object detection models. It constructs the fused boxes by using the confidence scores and coordinates of the base learners. We can assign weights to the base models according to their importance. The WBF algorithm works in the following steps:

1. The predicted boxes from each model are added to a list L which is then sorted in decreasing order of the confidence scores C .
2. Two empty lists A and B are created. A stores clusters of boxes and B stores the final fused boxes. Each index in A contains a set of boxes (or single box), which form a cluster, and each position in B contains the fused box from the corresponding cluster in A .

A matching box in B is found by iterating through the predicted boxes L . Two boxes are said to match if $IoU > threshold$.

3. If the match is not found, the box is added from the list L to the end of lists A and B as new entries and proceed to the next box in the list L . If the match is found, this box is added to A at the position corresponding to the matching box in B .

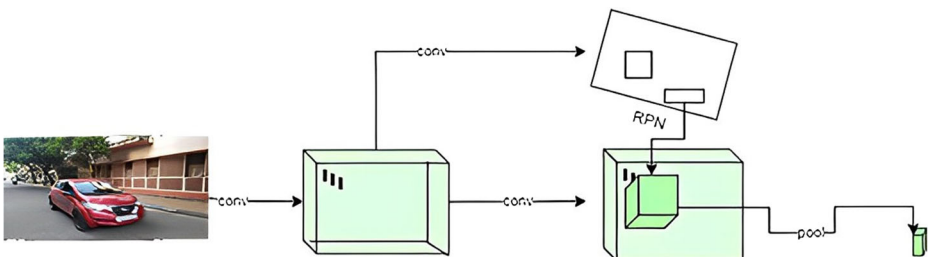


Fig. 14 Detailed architecture of R-FCN framework used in the present work

4. The fused box coordinates and confidence scores in B are recalculated, using all the boxes accumulated in the corresponding cluster in A.
5. The formulae for recalculating the coordinates are given below:

$$C = \frac{C_1 + C_2 + \dots + C_n}{n} \tag{1}$$

$$X1 = \frac{C_1 * LX1_1 + C_2 * LX1_2 + \dots + C_n * LX1_n}{C_1 + C_2 + \dots + C_n} \tag{2}$$

$$X2 = \frac{C_1 * LX2_1 + C_2 * LX2_2 + \dots + C_n * LX2_n}{C_1 + C_2 + \dots + C_n} \tag{3}$$

$$Y1 = \frac{C_1 * LY1_1 + C_2 * LY1_2 + \dots + C_n * LY1_n}{C_1 + C_2 + \dots + C_n} \tag{4}$$

$$Y2 = \frac{C_1 * LY2_1 + C_2 * LY2_2 + \dots + C_n * LY2_n}{C_1 + C_2 + \dots + C_n} \tag{5}$$

The overall procedure of the ensemble is shown in Fig. 15. An image is first predicted by the three base learners. The WBF method combines the bounding boxes of the three methods to give final bounding boxes. The advantage of the ensemble can be pictorially seen in Fig. 15. The night image, when experimented with YOLOv3, detects the bus in the middle, the cycle on its left, and the left-most bus which is partly present in the image. The objects are enclosed

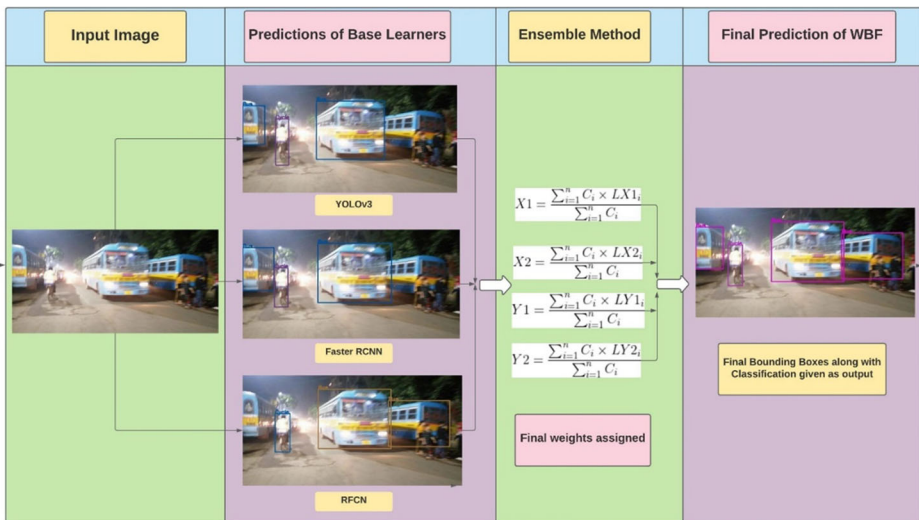


Fig. 15 Overview of the WBF method. The three base learners make their predictions. The WBF ensemble combines the predictions to produce the final set of bounding boxes

by the bounding boxes and labelled as found in the corresponding annotated image. Faster RCNN also detects the same objects as YOLOv3. Neither of these models detect the bus on the right in the image. However, the RFCN model detects the right-most bus which the previous two models have failed to detect. But the output of RFCN also has a flaw as it fails to detect the left-most bus which the other two models have detected successfully. The WBF method combines the output of the three models, thus giving an output where all three buses and the cycle have been detected accurately.

6 Results and discussion

It has been mentioned earlier that in the present work, we have developed a still image database, named as *JUVDsi v1*, considering Indian road scenario which can be used for vehicle localization along with vehicle type classification. In this section, we have reported the benchmark results obtained over the database using three state-of-the-art deep learning models, commonly used for object detection, along with a fusion method. Three base learners namely, YOLOv3, Faster RCNN, and RFCN are trained and tested on the database. These three models have been used for ensemble as they obtained the best detection accuracies on the validation set. YOLOv3 obtained 62.80% mAP score on the validation set, while Faster RCNN and RFCN obtained 65.32% and 67.63% mAP scores respectively. We have used mAP score as the metric to determine detection accuracy.

In the following subsections, we have defined the metric, used for determining the detection accuracy of the model and we have shown the results obtained by the base learners themselves and the results obtained after ensembling the three models by the WBF method.

6.1 Evaluation metrics

To determine detection accuracy, we have used the mAP score.

Intersection over Union (IoU) IoU [10] is used when calculating mAP. It denotes the amount of overlap between the predicted and ground truth bounding box. It lies between '0' and '1'. When there is no overlap between the boxes, IoU is 0 and it is 1 when two boxes completely overlap. For measuring an object as true positive, IoU must be above a threshold value. For calculating accuracy in this paper, we have used IoU threshold = 0.5.

True positive (TP) The number of bounding boxes that are correctly classified and have IoU > 0.5 with ground truth box.

False negative (FN) The number of ground truth boxes with no bounding box in the predictions with IoU > threshold or if the predicted bounding box with IoU > threshold doesn't make a mistake in classification.

False positive (FP) When a bounding box is predicted which should not be predicted.

Precision [21] Precision calculates the ability of a classifier not to label a true negative observation as positive.

$$\text{Precision} = \frac{TP}{TP + FP} \quad (6)$$

Recall (sensitivity) [21] Recall calculates the ability of a classifier to find positive observations in the database. If we want to be certain to find all positive observations, we can maximize recall.

$$\text{Recall} = \frac{TP}{TP + FN} \quad (7)$$

Average precision For each recall level, interpolated precision is calculated by taking the maximum precision measured for that recall level. The P-R curve for each class is made using the precision, recall and the calculated interpolated precision. The area under the curve is the Average Precision (AP) for the corresponding class.

Mean average precision (mAP) [36] mAP is the mean of the calculated AP scores for each class.

$$mAP = \frac{1}{N} \sum_{i=0}^{N-1} AP_i \quad (8)$$

Here, N is the number of classes and AP_i is the average precision for class i.

6.2 Result of three base learners

Graphs of the results of the base learners have been shown. Some models have given better detection accuracy than the others for some types of vehicles. The detection accuracies for each class have been given too.

Figure 16 shows the detection results of YOLOv3 on the test data. The number of TPs and the number of FPs per vehicle class are given in Fig. 16a, while Fig. 16b shows the mAP scores generated for each vehicle class. Similarly, Figs. 17 and 18 show the results obtained by the Faster R-CNN and RFCN models respectively .

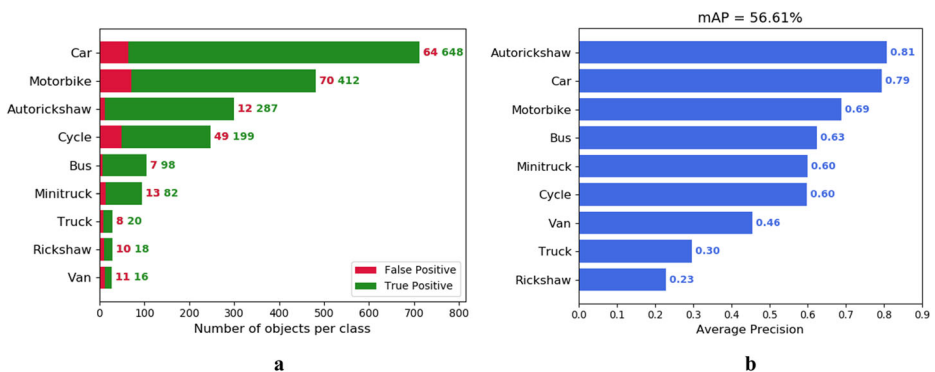


Fig. 16 Detection results obtained by YOLOv3 model on the test set. **a.** False positives and true positives of each class, **b.** mAP score

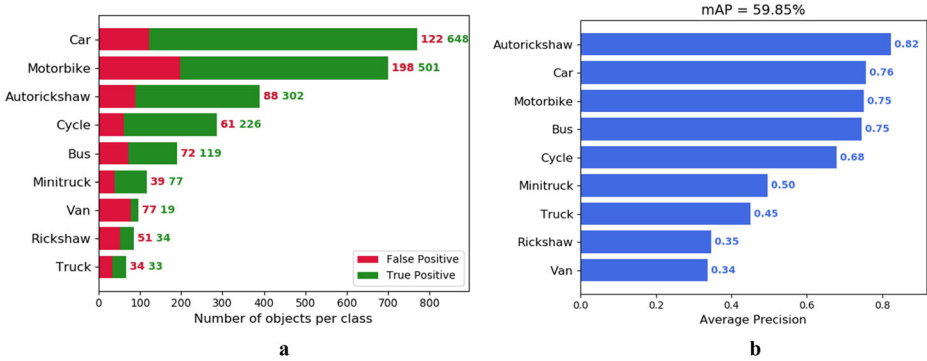


Fig. 17 Result obtained by Faster R-CNN method on the test set. **a.** False positives and true positives of each class, **b.** mAP score

Figure 19 shows a comparison of the output images provided by the three different base learners. The output images belong to different categories. The figure whose output is given in the left column is of the sunny category. The middle column shows output in cloudy condition and the right-most column shows the result of an image captured in the night time.

In Fig. 19, we have taken three scene images of varying difficulty and showed the result across different models. The image on the left column which is less complex than the others contains only one object of one class. The second object contains two objects of two classes. The third is the most complex of the three with multiple objects of multiple classes. The output of the three models have been shown in Fig. 20.

6.3 Results obtained using WBF

We have used an ensemble method that can achieve the results that are beyond the reach of the individual deep learning models used here as base learners. The ensemble is done using the WBF method, where confidence scores of the three base learners are taken as input. The weights assigned to YOLOv3, Faster RCNN and RFCN are natural numbers in the ratio of 1:2:2. We have experimented with other sets of weights but this set yields the best detection results among the other sets on the validation data. So, we have used this set of weights, to

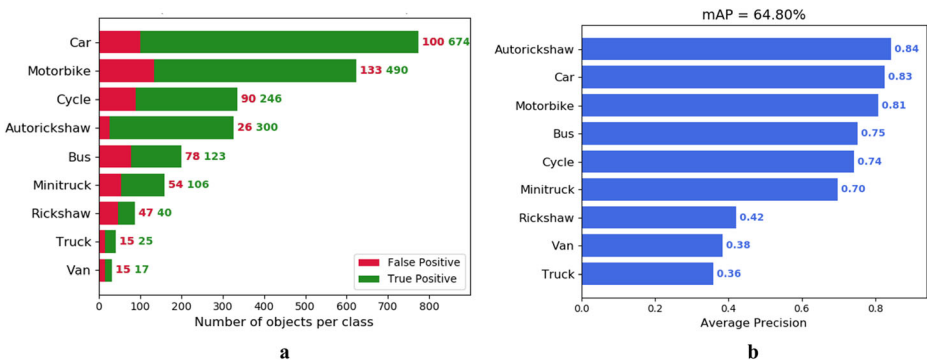


Fig. 18 Result obtained by RFCN method on the test set. **a.** False positives and true positives of each class, **b.** mAP score

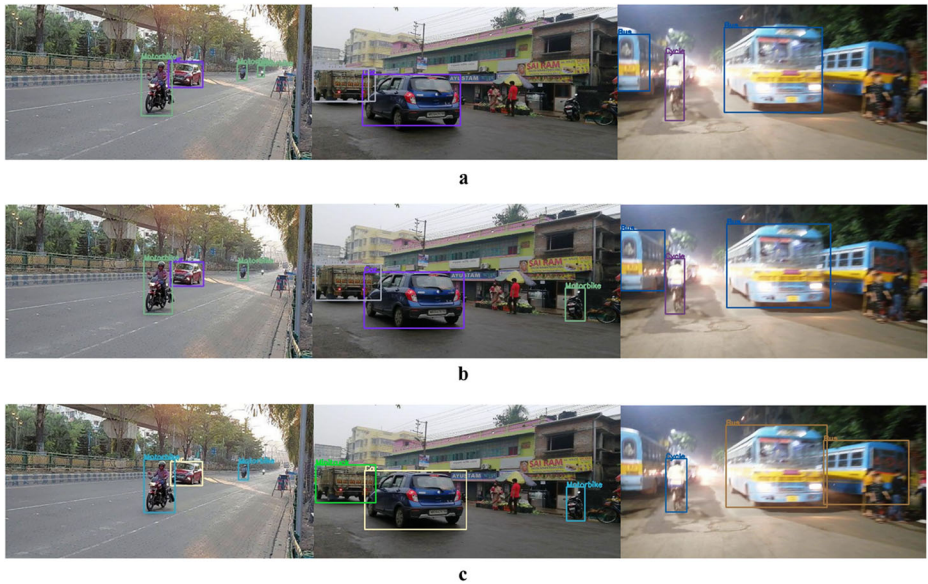


Fig. 19 Detection results on three sample images in three conditions provided by the three base learners: (a) YOLOv3, (b) Faster R-CNN and (c) R-FCN under the three different weather conditions on our developed database *JUVDSi v1*

benchmark the results. Using this set of weights, we got 79.64% detection accuracy on the validation set. In Fig. 21, we have shown some of the outputs obtained by the WBF method. In the following two subsections, we have provided mAP score obtained by WBF in different conditions. The first subsection contains results of predictions on images of all complexities in the test set. The whole test set predictions have been used for calculating the mAP. The second subsection contains results on images of lesser complexity. Images in the test set containing only one object in a frame have been segregated and their predictions are evaluated. The results are given in the second subsection.

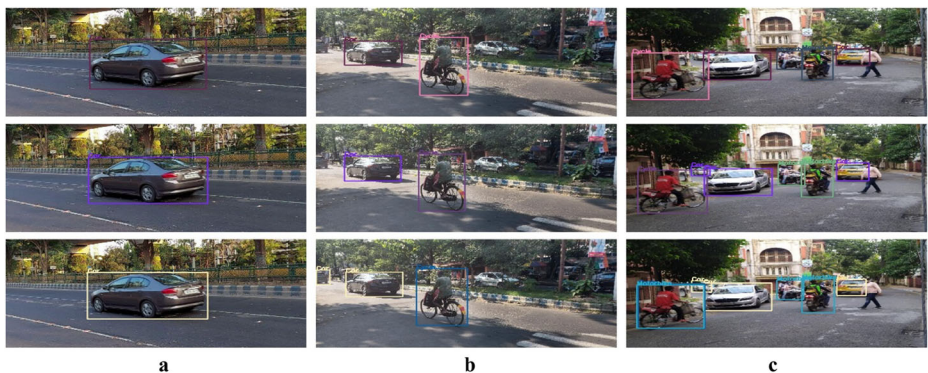


Fig. 20 Result of the three object detection models on three images of varied complexities. The top row contains output of YOLOv3, the middle row contains output of Faster RCNN and the bottom row contains output of R-FCN. The leftmost column contains images with only one object, the middle one with two objects and the rightmost one with multiple objects belonging to multiple classes. **a.** Easy, **b.** Medium, **c.** Hard



Fig. 21 Some example outputs produced by the ensemble method covering the different conditions of day and night times and also covering images having both single and multiple vehicles (belonging to different classes)

6.3.1 Experimentation using entire test set

In this subsection, we have shown the results obtained by the WBF method after ensembling the three base models. The P-R curves for all the 9 classes have been shown in Fig. 22. The curves are obtained after finding the outputs on the test data. The area under the curve (AUC) is a measure of AP for the corresponding class. From the curves, we can see that the AUC for vehicles like autorickshaw and car are much greater than that of a van or a rickshaw. Thus, the AP for autorickshaws and cars is more than that of vans and rickshaws. Therefore, the model fails to classify vans and rickshaws properly on a

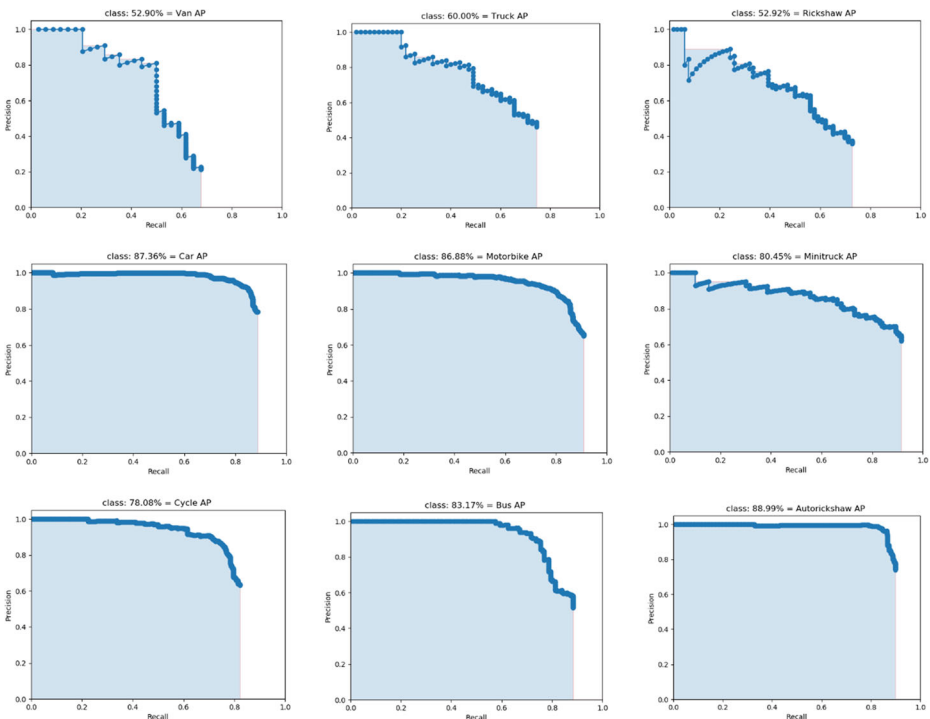


Fig. 22 P-R curves for each class as obtained by the WBF method on test set

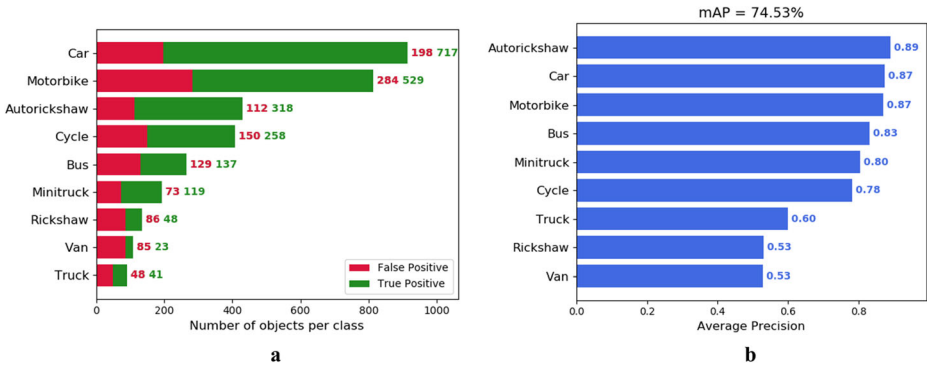


Fig. 23 Detection results obtained by ensemble of YOLOv3, Faster RCNN and RFCN by WBF method. **a.** False positives and true positives of each class, **b.** mAP score

number of occasions. The mean of all the APs gives the mAP score. The mAP score, number of TPs and number of FPs per class obtained using WBF method when tested on the entire test set have been shown in Fig. 23.

6.3.2 Single object test data

The images with single object in the frame are segregated, and the detection accuracy is calculated which is shown in Fig. 24. The weights of the models used while testing are obtained by training on the entire train data. This implies that they are not separately trained for single object containing images. It can be seen that there is a significant increase in mAP score when images with one object are tested. Therefore, the models perform much better on less complex images.

6.3.3 Test data and results for each weather condition

In this subsection, we have shown the performance obtained by the proposed WBF method for each weather condition present in our developed database. The weights of the models used while testing are obtained by training on the entire train data. This implies that they are not separately trained on images of that condition excluding the others. Fig. 25 shows the results on different conditions of day and weather. Table 4 shows the results obtained by the model on the entire test set.

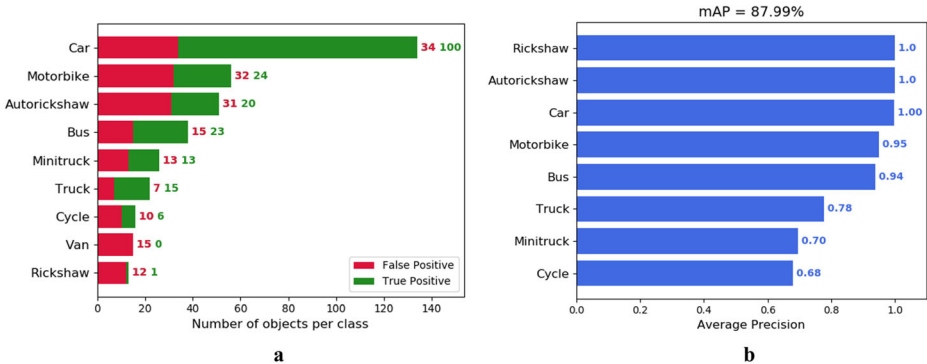
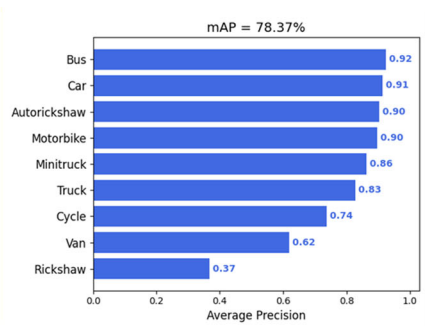
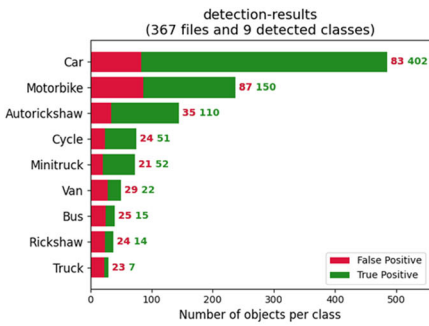
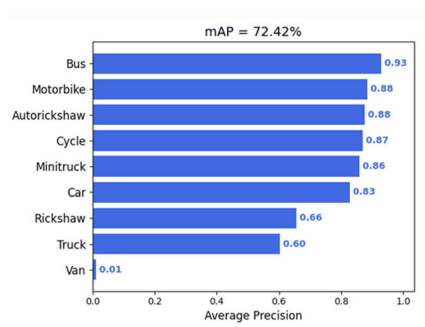
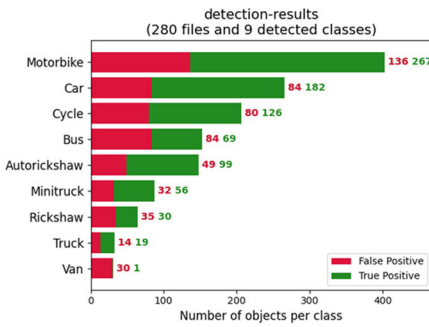


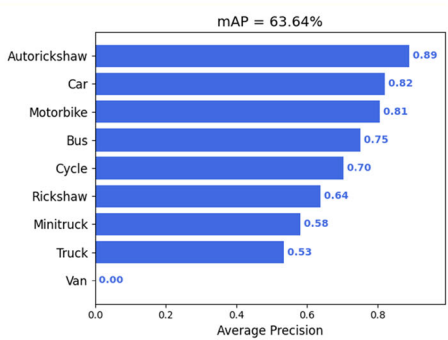
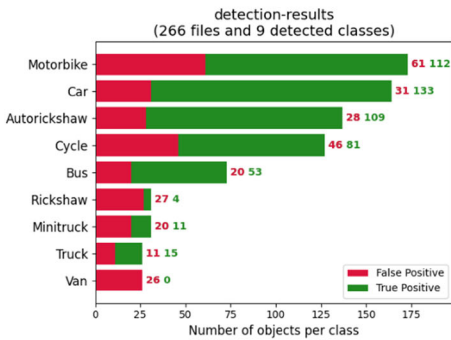
Fig. 24 Result of WBF on images containing single objects. **a.** False positives and true positives of each class, **b.** mAP score



a



b



c

Fig. 25 Results obtained after applying WBF method on the test set of our developed *JUVDsi v1* database for different conditions of day and weather. **a.** Day-sunny condition, **b.** Day-cloudy condition, **c.** Night-clear sky condition

Table 4 Comparison of mAP score obtained on the test set by the base learners and WBF

Model	mAP(IoU>0.5)
YOLOv3	56.61%
Faster RCNN	59.85%
RFCN	64.80%
Ensemble using WBF	74.53%

6.4 Results of panoptic segmentation on *JUVDSi v1* database

In this subsection, we have shown the visual results of panoptic segmentation on our developed *JUVDSi v1* database using two state-of-the-art models, namely, DeepLab-panoptic [13] and DETR-panoptic [54].

The DeepLab-panoptic model has been trained on the Cityscapes dataset [14] with the backbone as *SWideRNet – SAC – (1, 1, x)* [12]. Here, x can be 1, 3, 4, 5 but for our case, the value of x has been experimentally set to 3, scaling the backbone layers (excluding the stem) of Wide-ResNet-41 by a factor of x . This backbone only employs the Switchable Atrous Convolution (SAC) without the Squeeze-and-Excitation modules. The segmentation results on testing DeepLab-panoptic model on our database are shown in Fig. 26.

As it can be seen from Fig. 26a and b, the cars are quite accurately classified and segmented for images of both daylight and night conditions. However, the model trained on Cityscapes dataset is not able to identify the rickshaw class at all (as referred from Fig. 26c), which is an important intend of transport in Indian localities. The panoptic map along with the panoptic overlay as labelled above has been given.

The DETR-panoptic model has been trained on the very popular COCO (panoptic) dataset [25] with the backbone as ResNet-101 [19]. The model predicts a box and a binary mask for each object query after which the predictions are filtered with confidence less than 80%. Finally, the remaining masks are merged together using a pixel-wise argmax technique. We also obtain a alluring visualization by leveraging Detectron2's plotting utilities as shown in Fig. 27.

Some more results using DETR-panoptic model on sample images (considering different weather conditions) taken from our developed *JUVDSi v1* database are shown in Figs. 28, 29 and 30.

From the results, it can be observed that some vehicle classes like cars, trucks and motorcycles are more or less properly identified and segmented as evident from Figs. 28 and 29. However, in Fig. 30, it can be noticed that the vehicle class 'Van' which is a distinct class of our developed database is being misclassified as 'truck' whereas 'autorickshaw' class is being misclassified as 'bus' (see Fig. 30b for more detail) which in some cases are not being identified at all (Fig. 30c).

6.5 Analysis of the obtained results

- Out of the three base models, RFCN provides the highest mAP score of 64.80% followed by Faster RCNN and YOLOv3 which are 59.85% and 56.61% respectively.

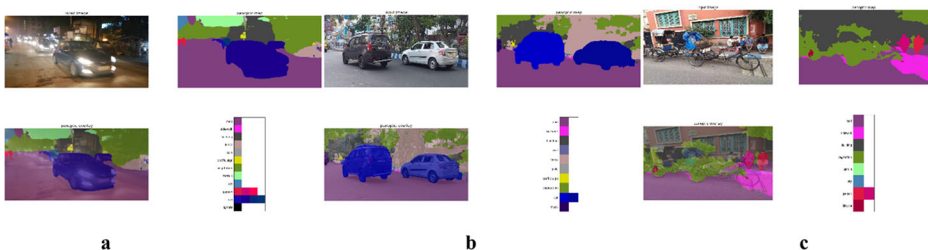


Fig. 26 Segmentation results obtained after testing DeepLab-panoptic model on our developed *JUVDSi v1* database

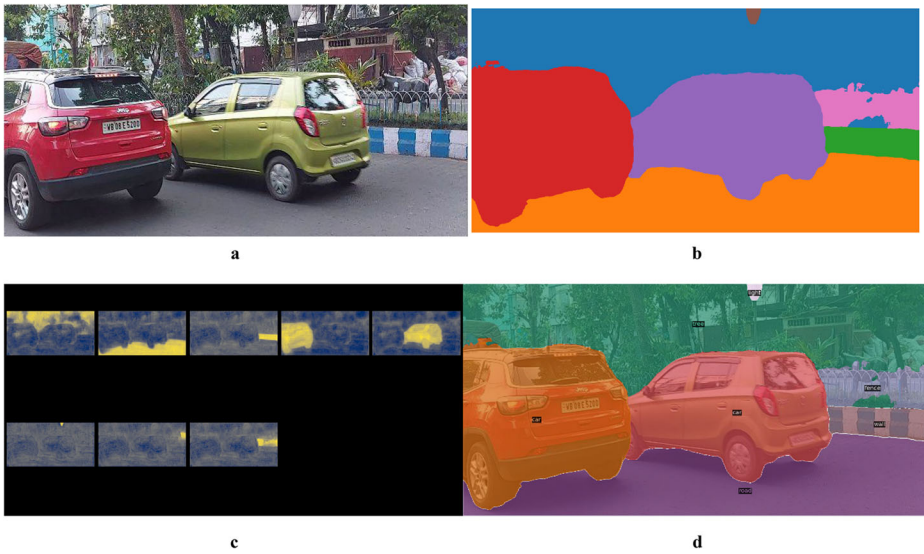


Fig. 27 Output images obtained after testing DETR-panoptic model our developed *JUVDisi v1* database. **a.** input image, **b.** DETR-panoptic-output, **c.** individual query masks, **d.** Output plotted using Detectron2



Fig. 28 Output images obtained after testing DETR-panoptic model our developed *JUVDisi v1* database. **a.** input image, **b.** detr-output



Fig. 29 Output images obtained after testing DETR-panoptic model our developed *JUVDisi v1* database. **a.** input image, **b.** detr- output

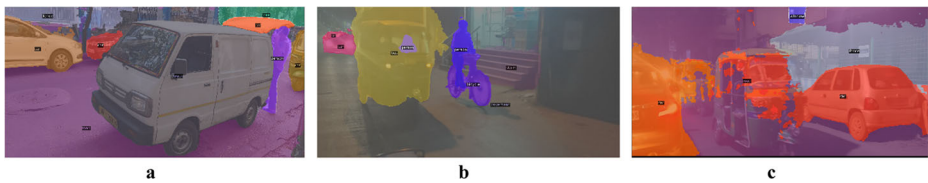


Fig. 30 Output images obtained after testing DETR-panoptic model our developed *JUVDsi v1* database

- The ensemble is done using the WBF method. The weights given to YOLOv3, Faster RCNN and RFCN are natural numbers in the ratio 1:2:2. The weights are determined experimentally.
- YOLOv3, RFCN and Faster RCNN are chosen to form the ensemble as they produce better results among other state-of-the-art object detectors such as Single Shot Detector version 1(SSDv1) which gives a detection accuracy of 42.36% only and SSDLitev2 obtains 43.11% mAP score.
- From the accuracy scores given in Table 5, it can be seen that the ensemble boosts the mAP score considerably. We have also tried NMS for ensemble but that has produced a mAP score of 66.58% only. So we have used the WBF method for benchmarking.
- As mentioned before, the developed database, *JUVDsi v1*, contains complex images with multiple objects belonging to same or different classes. The models perform considerably better when there is a single object in the frame. The mAP score obtained by WBF is 87.99% when tested on single object images.
- Both in the training and test sets, there is a huge class imbalance issue. Some classes of vehicles, for example, cars and motorbikes are frequent in the images of the training set, while trucks and vans are a few in number. In the training set, there are 1915 cars which are enough for the models to properly detect cars in the test set. The number of cars in the test set is 807, thus leading to better model accuracy while detecting cars as well. Meanwhile, there are only 101 vehicles of class van in the training set and 34 of those in the test set. Thus, the base learners do not have enough training data for proper learning of the images belonging to van class. Faster RCNN has the poorest mAP score for van i.e. 34%, while YOLOv3 has the highest mAP score for van which is 46%. The WBF ensemble boosts the mAP for van to 53%. The score could have been much higher if the class imbalance was not so high.
- Autorickshaws and e-Rickshaws have a very similar shape. From a distant view, it is hard to distinguish them. In our database, rickshaws are very few. So the models have, in some cases, misclassified some e-rickshaws and labelled them as autorickshaws. A method to resolve this problem is to collect more data for e-rickshaws so the base models can distinguish autorickshaws and e-rickshaws in a better way.

Table 5 Comparison of mAP score obtained on the test set of *JUVDsi v1* database by the base learners

Model	mAP(IoU < 0.5)
YOLOv3	56.61%
Faster RCNN	59.85%
RFCN	64.80%
SSD v1	42.36%
SSDLite v2	43.11%

6.6 Limitations

In this research work, we have incorporated only 3000 images in the *v1* of this database which may not be enough for training deep learning models properly. It is to be noted that we have taken images captured during day and night time only. We have not incorporated images taken in rainy or foggy conditions in either time of the day. In addition, only 9 vehicle classes are taken into consideration in the present database which does not include all the varieties of vehicle classes generally seen in an Indian road scenario. However, there are various models of similar vehicle types also available. To overcome these discrepancies of the present database, we are working on *version 2* of this database which will include more number of images captured in both day and night time during all possible weather conditions such as rainy, fog, sunny as well as cloudy weather. Additionally, we will also try to include more number of vehicle classes which will be beneficial for performing research work in any Indian road conditions where multiple vehicles may overlap one another.

7 Conclusion

AVD has become a part and parcel of real-time traffic management due to a wide range of issues including the increasing number of vehicles on roads. To come up with a model that can be used in practice, a pragmatic image/video database is a basic need, which developers can apply to measure the performance of their methods regarding both localization and classification of the vehicles in an automated way. However, most of the available databases are developed for the localization problem only and the number of freely available databases which can be used for both localization and classification are very less in number. Therefore, to bridge this research gap, in the present work, we have developed an image based database which serves as appropriate for Indian roads and can also be used for doing both localization and classification of the vehicles. For this, we have provided the required annotation of the images. The issues confronted regarding Indian roads such as large number of vehicles during peak hours, congestion of roads, worse condition of roads, traffic snarls add up to such issues for researchers. On the other hand, the available databases in literature mostly considered the developed countries while capturing the videos or images, which cannot portray these typical scenarios found in the Indian sub-continent. This database is made publicly available for research purposes only. We have benchmarked the results using three state-of-the-art deep learning models namely, YOLOv3, Faster RCNN and RFCN along with a standard ensemble method called WBF. WBF helps for filtering the predictions from base object detection models. Finally, we have achieved a mAP score of 74.53% which is competent considering the complexities found in the images.

Though, to the best of our knowledge, we have developed a database for the first time considering the Indian scenario to be used for AVD, there are certain limitations of this research work. Keeping those in mind, we have set some future plans for further progress which are listed below.

- We have around 3000 images in the version 1 of the database, which may not be enough for proper training of the deep learning models. Hence, we plan to collect more images for further work.

- We are trying to make arrangements to address the class imbalance present in the current version by collecting more data of trucks, vans and rickshaws as they are very less in number.
- We plan to capture images in different weather conditions like foggy, dark night, and rainy conditions etc.
- We will work towards incorporating more vehicle classes that are seen on Indian roads.
- We will work on achieving better results (both in localization and in classification) by developing a more efficient deep learning model than the existing ones so that it can deal with overlapped vehicles in a single frame.
- Lastly, we have another plan to develop a video database for AVD.

Acknowledgements The authors would like to thank the Center for Microprocessor Applications for Training Education and Research (*CMATER*) research laboratory of the Computer Science and Engineering Department, Jadavpur University, Kolkata, India for providing us the infrastructural support.

Data availability The databases generated during and/or analyzed during the current study are available in the GitHub repository, <https://github.com/JUVDsi/JUVD-Still-Image-database.git>

Declarations

Conflict of interest The authors declare there is no conflict of interest.

References

1. Agarwal S, Awan A, Roth D (2004) Learning to detect objects in images via a sparse, part-based representation. *IEEE Trans Pattern Anal Mach Intell* 26(11):1475–1490
2. Appathurai A, Sundarasekar R, Raja C, Alex EJ, Palagan CA, Nithya A (2020) An efficient optimal neural network-based moving vehicle detection in traffic video surveillance system. *Circ Syst Signal Process* 39(2):734–756
3. Arróspide J, Salgado L, Camplani M (2013) Image-based on-road vehicle detection using cost-effective histograms of oriented gradients. *J Vis Commun Image Represent* 24(7):1182–1190
4. Azimi SM, Bahmanyar R, Henry C, Kurz F (2020) EAGLE: large-scale vehicle detection dataset in real-world scenarios using aerial imagery. In: 2020 25th international conference on pattern recognition (ICPR), pp 6920–6927
5. Bahnsen CH, Moeslund TB (2018) Rain removal in traffic surveillance: does it matter? *IEEE Trans Intell Transp Syst* 20:2802–2819
6. Batra P et al (2022) A novel memory and time-efficient ALPR system based on YOLOv5. *Sensors* 22(14):5283
7. Behrendt K (2019) Boxy vehicle detection in large images. *IEEE/CVF International Conference on Computer Vision Workshop (ICCVW) 2019*:840–846
8. Caesar H, Bankiti V, Lang AH, Vora S, Liong VE, Xu Q, Krishnan A, Pan Y, Baldan G, Beijbom O (2019) nuScenes: a multimodal dataset for autonomous driving. *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) 2020*:11618–11628
9. Caraffi C, Vojřit T, Trefný J, Šochman J, Matas J (2012) A system for real-time detection and tracking of vehicles from a single car-mounted camera. In: 2012 15th international IEEE conference on intelligent transportation systems, pp 975–982
10. Che Z et al (2019) D²-City: a large-scale dashcam video dataset of diverse traffic scenarios. *arXiv Prepr:arXiv1904.01975*
11. Chen W, Qiao Y, Li Y (2020) Inception-SSD: an improved single shot detector for vehicle detection. *J Ambient Intell Humaniz Comput*:1–7

12. Chen L, Wang H, Qiao S (2020) Scaling wide residual networks for panoptic segmentation. *ArXiv:abs/2011.11675*
13. Cheng B, Collins MD, Zhu Y, Liu T, Huang TS, Adam H, Chen LC (2020) Panoptic-deeplab: a simple, strong, and fast baseline for bottom-up panoptic segmentation. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp 12475–12485
14. Cordts M, Omran M, Ramos S, Rehfeld T, Enzweiler M, Benenson R, Franke U, Roth S, Schiele B (2016) The cityscapes dataset for semantic urban scene understanding. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR) 2016*:3213–3223
15. Dai J, Li Y, He K, Sun J (2016) R-FCN: object detection via region-based fully convolutional networks. *arXiv Prepr:arXiv1605.06409*
16. Du S, Ibrahim M, Shehata MS, Badawy W (2013) Automatic license plate recognition (ALPR): a state-of-the-art review. *IEEE Trans Circuits and Syst Video Technol* 23:311–325
17. El-Sayed RS, El-Sayed MN (2020) Classification of vehicles' types using histogram oriented gradients: comparative study and modification. *IAES Int J Artif Intell* 9:700–712
18. Geiger A, Lenz P, Urtasun R (2012) Are we ready for autonomous driving? The Kitti vision benchmark suite. In: *2012 IEEE conference on computer vision and pattern recognition*, pp 3354–3361
19. He K, Zhang X, Ren S, Sun J (2015) Deep residual learning for image recognition. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR) 2016*:770–778
20. Hsieh J-W, Chen L-C, Chen D-Y (2014) Symmetrical SURF and its applications to vehicle detection and vehicle make and model recognition. *IEEE Trans Intell Transp Syst* 15(1):6–20
21. Bhattacharya D, Bhattacharyya A, Agrebi M, Roy A, Singh PK (2022) DFE-AVD: deep feature ensemble for automatic vehicle detection. In: *Proceedings of international conference on intelligence computing systems and applications (ICICSA 2022)*
22. Huang X et al (2018) The apolloscape dataset for autonomous driving. *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*:954–960
23. Krause J, Stark M, Deng J, Fei-Fei L (2004) 3D object representations for fine-grained categorization: supplementary material. *IJCV*:1
24. Lee H, Ullah I, Wan W, Gao Y, Fang Z (2019) Real-time vehicle make and model recognition with the residual SqueezeNet architecture. *Sensors* 19(5):982.
MDPI AG. Retrieved from <https://doi.org/10.3390/s19050982>
25. Lin T-Y et al (2014) Microsoft coco: common objects in context. *European conference on computer vision*: 740–755
26. Liu W, Liao S, Hu W (2019) Towards accurate tiny vehicle detection in complex scenes. *Neurocomputing* 347:24–33
27. Manzoor MA, Morgan Y, Bais A (2019) Real-time vehicle make and model recognition system. *Mach Learn Knowl Extr* 1(2):611–629
28. Mirthubashini J, Santhi V (2020) Video based vehicle counting using deep learning algorithms. In: *2020 6th international conference on advanced computing and communication systems (ICACCS)*, pp 142–147
29. Naseer S, Shah SMA, Aziz S, Khan MU, Iqtidar K (2020) Vehicle make and model recognition using deep transfer learning and support vector machines. In: *2020 IEEE 23rd international multitopic conference (INMIC)*, pp 1–6
30. Nazemi A, Azimifar Z, Shafiee MJ, Wong A (2019) Real-time vehicle make and model recognition using unsupervised feature learning. *IEEE Trans Intell Transp Syst* 21(7):3080–3090
31. Nguyen VD, Le A, Duong TM, Debnath NC (2020) Robust and real-time obstacle region detection based on depth feature for vehicle detection.
International Conferences on Artificial Intelligence and Computer Vision
32. Razakarivony S, Jurie F (2016) Vehicle detection in aerial imagery: a small target detection benchmark. *J Vis Commun Image Represent* 34:187–203
33. Ren S, He K, Girshick RB, Sun J (2015) Faster R-CNN: towards real-time object detection with region proposal networks. *IEEE Trans Pattern Anal Mach Intell* 39:1137–1149
34. Rezaei M, Terauchi M (2014) iROADS dataset (intercity roads and adverse driving scenarios). *EISATS, Set*, p 10
35. Sagar A, Soundrapandiyan R (2020) Semantic segmentation with multi scale spatial attention for self driving cars. *IEEE/CVF International Conference on Computer Vision Workshops (ICCVW) 2021*:2650–2656
36. Sakaridis C, Dai D, Gool LV (2021) ACDC: the adverse conditions dataset with correspondences for semantic driving scene understanding. *IEEE/CVF International Conference on Computer Vision (ICCV) 2021*:10745–10755

37. Sang J, Wu Z, Guo P, Hu H, Xiang H, Zhang Q, Cai B (2018) An improved YOLOv2 for vehicle detection. *Sensors* 18(12):4272.
MDPI AG. Retrieved from <https://doi.org/10.3390/s18124272>
38. Saravi S, Edirisinghe EA (2013) Vehicle make and model recognition in CCTV footage. In: 2013 18th international conference on digital signal processing (DSP), pp 1–6
39. Sivaraman S, Trivedi MM (2010) Improved vision-based lane tracker performance using vehicle localization. *IEEE Intelligent Vehicles Symposium 2010*:676–681
40. Sivaraman S, Trivedi MM (2013) Integrated lane and vehicle detection, localization, and tracking: a synergistic approach. *IEEE Trans Intell Transportation Systems* 14:906–917
41. Solovyev R, Wang W, Gabruseva T (2021) Weighted boxes fusion: ensembling boxes from different object detection models. *Image Vis Comput* 107:104117
42. Tian Y, Yang G, Wang Z, Wang H, Li E, Liang Z (2019) Apple detection during different growth stages in orchards using the improved YOLO-V3 model. *Comput Electron Agric* 157:417–426
43. Varma G, Subramanian A, Nambodiri AM, Chandraker M, Jawahar CV (2018) IDD: a dataset for exploring problems of autonomous navigation in unconstrained environments. *IEEE Winter Conference on Applications of Computer Vision (WACV) 2019*:1743–1751
44. Wang H, Yu Y, Cai Y, Chen X, Chen L, Liu Q (2019) A comparative study of state-of-the-art deep learning algorithms for vehicle detection. *IEEE Intell Transp Syst Mag* 11:82–95
45. Wang Z, Wan Q, Qin Y, Fan S, Xiao Z (2020) Research on intelligent algorithm for alerting vehicle impact based on multi-agent deep reinforcement learning. *J Ambient Intell Humaniz Comput* 12:1–11
46. Wang D, Al-Rubaie A, Alsarkal Y, Stincic S, Davies J (2021) Cost effective and accurate vehicle make/model recognition method using YoloV5. In: 2021 international conference on smart applications, communications and networking (SmartNets), pp 1–4
47. Wen L, Du D, Cai Z, Lei Z, Chang M, Qi H, Lim J, Yang M, Lyu S (2015) UA-DETRAC: a new benchmark and protocol for multi-object detection and tracking. *Comput Vis Image Underst* 193:102907
48. Wu H, Zhang X, Story BA, Rajan D (2019) Accurate vehicle detection using multi-camera data fusion and machine learning. In: ICASSP 2019–2019 IEEE international conference on acoustics, speech and signal processing (ICASSP), pp 3767–3771
49. Yang Z, Pun-Cheng LSC (2018) Vehicle detection in intelligent transportation systems and its applications under varying environments: a review. *Image Vis Comput* 69:143–154
50. Yang J, Li Y, Zhang Q, Ren Y (2019) Surface vehicle detection and tracking with deep learning and appearance feature. In: 2019 5th international conference on control, automation and robotics (ICCAR), pp 276–280
51. Yu F, Chen H, Wang X, Xian W, Chen Y, Liu F, Madhavan V, Darrell T (2018) BDD100K: a diverse driving dataset for heterogeneous multitask learning. *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) 2020*:2633–2642
52. Zende O, Honauer K, Murschitz M, Steininger D, Domínguez GF (2018) WildDash - creating hazard-aware benchmarks. *European Conference on Computer Vision*:402–416
53. Zhang J, Yang K, Stiefelhagen R (2020) ISSAFE: improving semantic segmentation in accidents by fusing event-based data. *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) 2021*: 1132–1139
54. Zhu X, Su W, Lu L, Li B, Wang X, Dai J (2020) Deformable DETR: deformable transformers for end-to-end object detection. *ArXiv:abs/2010.04159*
55. Zhu X, Lyu S, Wang X, Zhao Q (2021) TPH-YOLOv5: improved YOLOv5 based on transformer prediction head for object detection on drone-captured scenarios. *IEEE/CVF International Conference on Computer Vision Workshops (ICCVW) 2021*:2778–2788

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

Springer Nature or its licensor (e.g. a society or other partner) holds exclusive rights to this article under a publishing agreement with the author(s) or other rightsholder(s); author self-archiving of the accepted manuscript version of this article is solely governed by the terms of such publishing agreement and applicable law.

Affiliations

Avirup Bhattacharyya¹ · Avigyan Bhattacharya¹ · Sourajit Maity¹ · Pawan Kumar Singh²  · Ram Sarkar¹

Avirup Bhattacharyya
avirupbhattacharyya2001@yahoo.com

Avigyan Bhattacharya
avigyanbhattacharya123@gmail.com

Sourajit Maity
sourajit.cse.ju@gmail.com

Ram Sarkar
ramjucse@gmail.com

¹ Department of Computer Science and Engineering, Jadavpur University, 188, Raja S.C. Mallick Road, Kolkata, West Bengal 700032, India

² Department of Information Technology, Jadavpur University, Jadavpur University Second Campus, Plot No. 8, Salt Lake Bypass, LB Block, Sector III, Salt Lake City, Kolkata, West Bengal 700106, India