Object Detection and Tracking using Deep Learning and Artificial Intelligence for Video Surveillance Applications

Mohana[1]
Department of Electronics and Communication Engineering,
RV College of Engineering® Bengaluru- 560059 and
affiliated to Visvesvaraya Technological University,
Belagavi Karnataka, India

HV Ravish Aradhya[2]
Department of Electronics and Communication Engineering,
RV College of Engineering®, Bengaluru- 560059 and
affiliated to Visvesvaraya Technological University,
Belagavi, Karnataka, India

Abstract—**Data is the new oil in current technological society. The impact of efficient data has changed benchmarks of performance in terms of speed and accuracy. The enhancement is visualizable because the processing of data is performed by two buzzwords in industry called Computer Vision (CV) and Artificial Intelligence (AI). Two technologies have empowered major tasks such as object detection and tracking for traffic vigilance systems. As the features in image increases demand for efficient algorithm to excavate hidden features increases. Convolution Neural Network (CNN) model is designed for urban vehicle dataset for single object detection and YOLOv3 for multiple object detection on KITTI and COCO dataset. Model performance is analyzed, evaluated and tabulated using performance metrics such as True Positive (TP), True Negative (TN), False Positive (FP), False Negative (FN), Accuracy, Precision, confusion matrix and mean Average Precession (mAP). Objects are tracked across the frames using YOLOv3 and Simple Online Real Time Tracking (SORT) on traffic surveillance video. This paper upholds the uniqueness of the state of the art networks like DarkNet. The efficient detection and tracking on urban vehicle dataset is witnessed. The algorithms give real-time, accurate, precise identifications suitable for real-time traffic applications.**

Keywords—Artificial Intelligence (AI); Computer Vision (CV); Convolution Neural Network (CNN); You Look Only Once (YOLOv3); Urban Vehicle Dataset; Common objects in Context (COCO); Object detection; object tracking

I. Introduction

Over the past years domains like image analysis and video analysis has gained a wide scope of applications. CV and AI are two main technologies dominating technical society. Technologies try to depict the biology of human. Human vision is the sense through which a perception of outer 3D world is perceived. Human Intelligence is trained over years to distinguish and process scene captured by eyes. These intuitions acts as a crux to budding new technologies. Rich resource is now accelerating researchers to excavate more details form the images. These developments are due to state-of the-art methods like CNN. Applications from Google, Facebook, Microsoft, and Snapchat are all results of tremendous improvement in Computer vision and Deep learning. During time, the vision-based technology has transformed from just a sensing modality to intelligent computing systems which can understand the real world. Computer vision applications like vehicle navigation, surveillance and autonomous robot navigation find Object detection and tracking as important challenges. For tracking vehicles and other real word objects, video surveillance is a dynamic environment. In this paper, efficient algorithm is designed for object detection and tracking for video Surveillance in complex environment.

Object detection and tracking goes hand in hand for computer vision applications. Object detection is identifying object or locating the instance of interest in-group of suspected frames. Object tracking is identifying trajectory or path; object takes in the concurrent frames. Image obtained from dataset is, collection of frames. Basic block diagram of object detection and tracking is shown in Fig. 1. Data set is divided into two parts. 80 % of images in dataset are used for training and 20 % for testing. Image is considered to find objects in it by using algorithms CNN and YOLOv3. A bounding box is formed across object with Intersection over union (IoU) > 0.5. Detected bounding box is sent as references for neural networks aiding them to perform Tracking. Bounded box is tracked in concurrent frames using Multi Object Tracking (MOT). Importance of this research work is used to estimate traffic density in traffic junctions, in autonomous vehicles to detect various kinds of objects with varying illumination, smart city development and intelligent transport systems [18]. Organization of paper is, Section II identifies research gap through extensive literature survey. Section III covers Fundamental Concepts of Object detection and Tracking. Section IV describes design, implementation details and specifications. Section V discusses simulation results and analysis. Section VI describes conclusions and future scope.

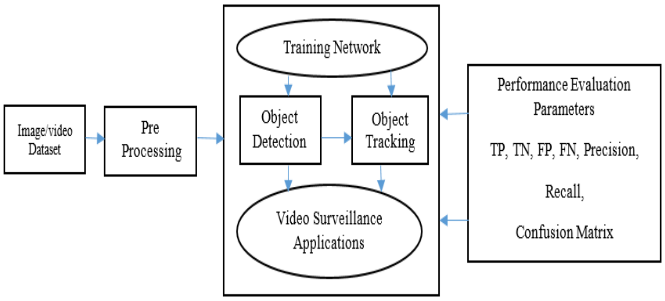


Fig. 1. Block Diagram of Object Detection and Tracking.

II. Literature Survey

Adopting Tile convolution neural network and recursive mode of same network helps in finding objects aiding applications for Driver assistance systems (DAS). Approach includes unsupervised training to help learn and modulate weights based on wide range of training data. Obstacle validation algorithms are included to reduce the count of valid detections [1]. Concepts like Optical flow and Histogram of magnitudes is used to analyze motion of objects, which are not evident to bare eyes. Detection of normal and abnormal events is achieved by classification and localization helping campus environment to differentiate between normal and abnormal events [2]. Features are extracted using pretrained network; classified results are differentiated using SVM. Approach helps in guiding the route for ITS [3]. Many approaches like feature extraction based on color and gradients fail to give spatial positioning in the image. The challenges are overcome by employing Analysis of principal components by PCANet [4] pipeline of image undistortion, image registration, classification and detections based on coordinates and velocities. Approach uses detectors like FAST, FREAK descriptors and followed by classification of Squeeze Net [5]. The workflow of candidate target generation, extracting features from candidate targets, the ground truth boxes around objects assist in tracking. The objects are classified using VGGNet [6]. CNN was designed to classify images, was repurposed to perform the object detection. The approach treats object detection as a relapse for object class to bounding objects detected. Series of gradual improvements has been witnessed from RCNN, Fast RCNN and faster RCNN then finally to YOLO. Instead of assessing image repetitively as in CNN, image is scanned once for all, thereby increasing the processing of frames per second (fps). YOLO is trained based on loss occurred unlike the traditional Classification approach [7]. Paper describes about video analytics part for road traffic. One of main application area apart from vehicle detection and tracking is vehicle counting. One of the novel algorithm called Single Shot Detector (SSD) is employed. Algorithm handles features like Binary large objects. It gives better results in applications like classification of objects. Object tracking employs concepts like background subtraction and virtual coil method. In terms of precision SSD outperforms YOLO versions. Swiftness and precision are always tradeoffs while selecting the right algorithm for object detection with the speed of 58fps performance metric for accuracy exceeds 85% [8], paper explains about upgradation to YOLO was made in the paper. Gradual updating has been witnessed throughout series of YOLO versions namely YOLOv1, YOLOv2, YOLOv3. YOLOv3 is state of the art technology. Upgradation such as thinner bounding boxes without affecting adjacent pixels. YOLOv3's implementation on COCO dataset shows mAP as good as SSD. YOLOv3 gives three times faster results. YOLOv3 promises in detecting smaller objects [9]. With increase in vehicle density in urban region, Single object tracking will no longer cater for the need. Multi object tracking is achieved by employing kernelized correlation filter (KCF). Many KCF are run in parallel. KCF is best suited when images have occlusions. KCF when combined with background subtraction yield reliable results on the urban traffic [10] [12] [14].

Deep Networks require more computer power and time, more data, better performance of Neural Nets. The success of any algorithm lies in parameter tuning. Algorithms are application specific. Fine-tuning of state of the art Neural Nets decreases training time while increasing accuracy. Results are dependent on dataset used, algorithm and network employed.

III. Object Detection and Tracking

There is a wide range of computer vision tasks benefiting society such as object classification, detection, tracking, counting, Semantic Segmentation, Captioning image, etc. Process of identifying objects in an image and finding its position is known as object detection.

Various object detection tasks as shown in Fig. 2. With advancements in field of computer vision assisted by AI, realization of tasks was realizable along t time scale. Semantic segmentation task of clustering pixels based on similarities. Classification + Localization and object detection method of identifying class of object and drawing a bounding box around it to make it distinct. Instance segmentation is semantic segmentation applied to multi objects. The general intuition to perform the task is to apply CNN over the image. CNN works on image patches to carry out the task many such salient regions can be obtained by Region-Proposal Networks like Region Convolution Neural network (RCNN), Fast- Region Convolutional Neural Network (Fast-RCNN), Faster- Region Convolutional Neural Network (Faster-RCNN). To perform selective search for object recognition Hierarchal Grouping Algorithm is used. Few bottlenecks by these approaches are mitigated by state-of the-art algorithms like You Only Look Once (YOLO), Single shot Detector (SSD). The efficient object detection algorithm is one which assures to give bounding box to all objects of vivid size to be recognized, with great computational capabilities, faster processing. YOLO and SSD assure to render promising results, but have a tradeoff between speed and accuracy. Hence, selection of algorithm is application specific [15].

A. Convolutional Neural Networks (CNN)

CNN is widely used neural network architecture for computer vision related tasks. Advantage of CNN is that it automatically performs feature extraction on images i.e. important features are detected by the network itself.

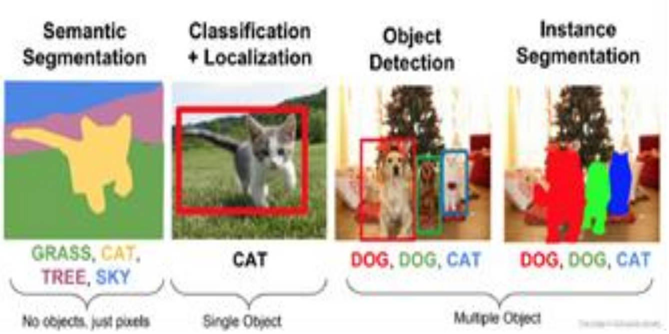


Fig. 2. Object Detection Tasks [7].

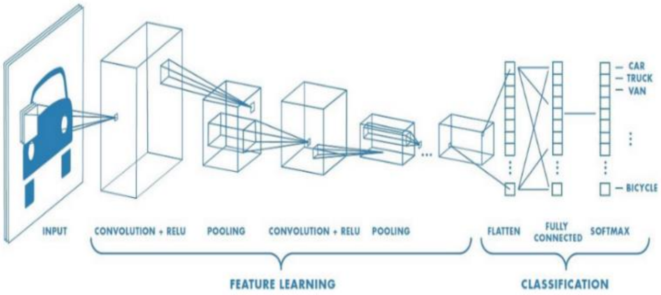


Fig. 3. Overview of CNN Architecture [2].

CNN is made up of three important components called Convolutional Layer, Pooling layer, fully connected Layer as shown in Fig. 3. Considering a gray scale image of size 32*32 would have 1024 nodes in multi-layer approach. This process of flattening pixels loses spatial positions of the image. Spatial relationship between picture elements is retained by learning internal feature representation using small squares of input data.

1) Convolutional layer: Convolutional Layer encompasses filters and feature maps. Filters are processors of a particular layer. These filters are distinct from one another. They take pixel value as input and gives out feature Map. Feature map is output of one filter layer. Filter is traversed all along the image, moving one pixel at a time. Activation of few neurons takes place resulting in a feature map.

2) Pooling layer: Pooling layer is employed to reduce dimensionality. Pooling layers are included after one or two convolutional layer to generalize features learnt from previous feature maps. This helps in reducing chances of over fitting from training process.

3) Fully connected layer: Fully connected layer is used at the end to assign the feature to class probability after extracting and consolidating features from Convolutional Layer and pooling later respectively. These layers use linear activation functions or softmax activation function.

B. *You Only Look Once (YOLOv3)*

YOLO version 1 and 2 applies softmax functions convert score into probabilities. This approach is feasible when objects are mutually exclusive only. YOLOv3 employs multi label classification. Independent logistic classifier is used to calculate likeliness of input belong to a specific label. Loss is calculated using binary-cross entropy of each label. Since we omit the softmax function complexity is reduced.

1) Optimization of Bounding Boxes: By using Logistic, regression YOLO v3 predicts the score of presence of object. A ground truth box is defined to all objects, if anchor box overlaps the most with ground truth box then objectness score is said to be 1. For the anchor boxes whose overlap is greater than the preselected threshold, the anchor box incurs null cost. Every ground truth box is mapped with only one anchor box. If anchor box is not selected and assigned to bounding box then no classification and localization loss is considered, only confidence loss is calculated.

The anchor box is regressed to the ground truth box by gradual optimization as shown in Fig. 4. Coordinate parameters are now defined as

$$b_x = \sigma(t_x) + c_x \tag{1}$$

$$b_y = \sigma(t_y) + c_y \tag{2}$$

$$b_w = p_w e^{t_w} \tag{3}$$

$$b_h = p_h e^{t_h} \tag{4}$$

Where, t_x, t_y, t_w, t_h are the predictions made by YOLO. c_x, c_y is top left corner of grid cell of the anchor. $p_w p_h$ are the width and height of anchor. b_x, b_y, b_w, b_h are predicted boundary box. $\sigma(t_o)$ is box confidence score.

2) Feature pyramid Network (FPN): YOLOv3 makes three predictions in every point of image. The prediction includes a bounding box, score of objectness followed by 80 class score hence we have S*S*[3*(4+1+80)] predictions. This approach is similar to feature pyramid networks.

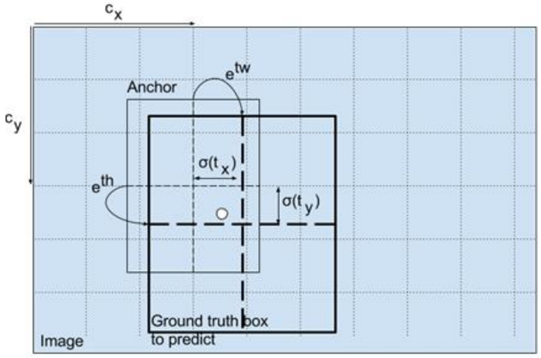


Fig. 4. Anchor Box Regression [9].

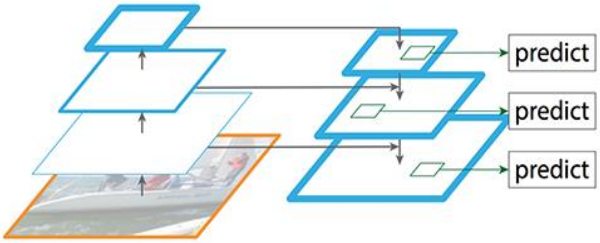


Fig. 5. Feature Pyramid Network [9].

Predictions are made at 3 different scales as in Fig. 5. The initial prediction is made at last feature map layer. Then feature map is up sampled by factor of 2. YOLOv3 merges feature map with up sampled feature using element wise addition. Convolutional layer is applied to obtain second predictions. Repeating second prediction will yield high semantic information.

Two stage algorithms from Region Proposal networks family of algorithms have two different networks for proposing regions and extracting features. FPS of RCNN is 7, which is quite low to handle real-time applications. One stage algorithm overcomes this drawback by employing single shot detectors. Single Shot detectors face trade-off between accuracy and real-time processing. The algorithm faces issues in identifying small objects or objects that are too close. Though SSD networks are equally in boom as much as YOLO, algorithm might outperform YOLO in terms of speed, but spatial resolution has dropped significantly and hence missing out in locating small objects. Solution to challenge is increasing image resolutions. YOLO family upgrades its accuracy, latency. YOLOv3 has DarkNet-53 has its backbone. The network has less BFLOP (Billion floating-point operations) compared to residual Network-512. The inclusion of Feature Pyramid network (FPN) helps in detecting objects that are small. FPN uses both bottom-down and a top-down pathway. Bottom-up approach is used for feature extraction. As we propagate through this approach, spatial resolution minimizes. Semantic value for each layer increases.

C. Object Tracking

Internet is the main network connecting millions of people in world. Main entertainment factor and the source of greater knowledge is video. Video is collection of frames. The negligible time gap between frames makes the stream of photos looks like flow of scenes. When designing algorithm for video processing. Videos are classified into two classes. Video stream is an ongoing process for video analysis. The processor is not aware of future frames. Video sequence is video of fixed length. All the consecutive frames are obtained prior to processing of current frame. Motion is distinct factor that differentiates video form frame. Motion is a powerful visual Que. Object properties and action can be realized by noticing only sparse points in the image.

D. Simple Online Real Time Tracking (SORT)

SORT is a realistic approach to achieve Multi Object Tracking (MOT). Performance of SORT is enhanced by ques such as appearance; this association of appearance to SORT enhances the performance of SORT and increases performance during Scenario like longer periods of occlusion. SORT is a framework that has Kalman filtering has its crux. Image by image data association is achieved by Hungarian method over an association metric like appearance that measures bounding box overlap.

1) Track Handling and state estimation: The assignment problem maps prediction of Kalman filter to that of newly arrived measurements. The task of associating two vectors is performed by Hungarian algorithm. Adding additional information like motion and appearance parameters in conjunction with association helps in better mappings.

$$d^{(1)}(i,j) = (d_j - y_j)^T s_i^{-1}(d_j - y_i) \quad (5)$$

Unlikely association can be removed by thresholding at 95% confidence interval. The decision is given with an indicator.

$$b_{i,j}^{(1)} = 1\left[d^{(1)}(i,j)\right] \leq t^{(1)} \quad (6)$$

When the motion uncertainty is large mahalanobis distance is not suitable, hence another metric to aid in association. Metric computes appearance descriptor for each bounding box detection d_j.

$$d^{(2)}(i,j) = min\left\{1 - {r_i}^T r_k^{(i)} \middle| r_k^{(i)} \in R_I\right\} \quad (7)$$

Combination of both metrics is

$$c_{i,j} = \lambda d^{(1)}(i,j) + (1-\lambda) d^{(2)}(i,j) \quad (8)$$

IV. Design and Implementation

A CNN is designed and is trained on Urban Vehicle dataset, which is an Indigenous dataset for traffic surveillance applications. Specifications of Urban Vehicle Dataset or on road vehicle dataset [11] is tabulated in Table I. And number of images considered for each class is tabulated in Table II.

In total there are 64339 images belonging to 4 classes that are taken under different times of the day and capturing conditions, including NIR images. The images are classified based on utility and size of vehicles. The auto class has images of three wheelers; Heavy class includes buses, trucks, Freight carriers; Light class has cars, SUVs and sedans and two wheelers include motorcycles and bicycles. Most of images have only one object belonging to its respective class.

Some of the sample images of dataset is shown in Fig. 6. Hardware and software requirements are tabulated in Table III and Table IV.

TABLE. I. Dataset Specifications

Source	Traffic Monitoring Cameras
Image Type	RGB, NIR
Image Extension	Jpg
Image Dimension	Variable Size
Image Quality	Medium to Blurred

TABLE. II. Urban Vehicle Dataset

Directory	**Autos**	**Heavy**	**Light**	**Two Wheelers**	**Total**
RGB_Day	2530	2915	12927	13340	31712
RGB_Evening	3122	3709	14654	10027	31512
NIR	124	127	524	340	1115

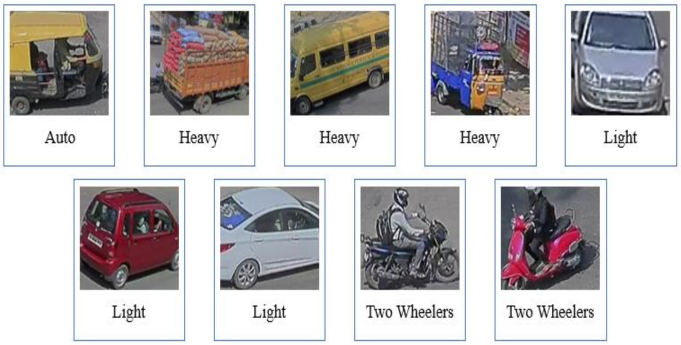


Fig. 6. Sample Images of urban Vehicle Dataset [11].

TABLE. III. HARDWARE REQUIREMENTS

Processor	Intel i7, 8th Gen quad core
Clock Speed	1.8 GHz
RAM	16 GB
Storage	500 GB SSD
GPU	Nvidia MX

TABLE. IV. SOFTWARE REQUIREMENTS

Distribution	Anaconda Navigator
API	Keras
Library	Tensor Flow, OpenCV
Packages	Matplotlib, numpy, pandas, scikitLearn
Language	Python
IDE	Spyder, Jupyter Notebook
GPU Architecture	CUDA
Applications	LabelImg, TensorBoard

A. *Neural Network Training*

Flowchart of neural network training is as shown in Fig. 7. First step in training a network using deep learning for an application is to prepare an appropriate dataset and make Train-Test Split depending on the available data. Suitable network is designed or selected (in case of Transfer Learning) for training [13]. Validation Loss is monitored throughout the training process to produce a very less constant value after few epochs, if not then the hyper parameter tuning is performed on model to give lowest possible validation loss values. Model with best validation loss is saved and tested on real world dataset. The model is said to be good if a descent precision and recall values are obtained for new datasets else the model needs to be trained on enhanced dataset for increased performance.

B. *Single Object Detection*

Fig. 8 shows flow chart of single object detection. Necessary libraries are imported first and training data is given input via the Google drive. Google-Colab, an online simulation tool for python and Tensor-flow algorithms was used. The algorithm then compiles data and learns form it in a supervised manner [16].

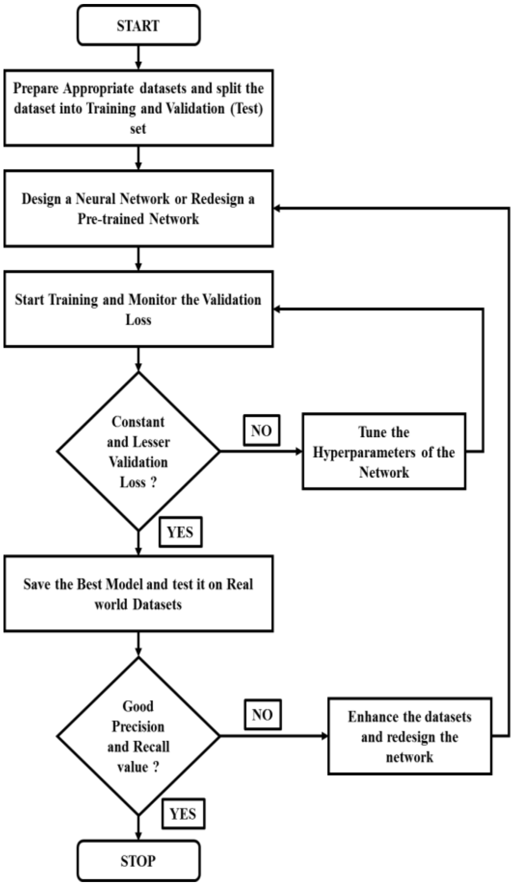

START
Prepare Appropriate datasets and split the dataset into Training and Validation (Test) set
Design a Neural Network or Redesign a Pre-trained Network
Start Training and Monitor the Validation Loss
Constant and Lesser Validation Loss ?
NO
Tune the Hyperparameters of the Network
YES
Save the Best Model and test it on Real world Datasets
Good Precision and Recall value ?
NO
Enhance the datasets and redesign the network
YES
STOP

Fig. 7. Flowchart of Neural Network Training.

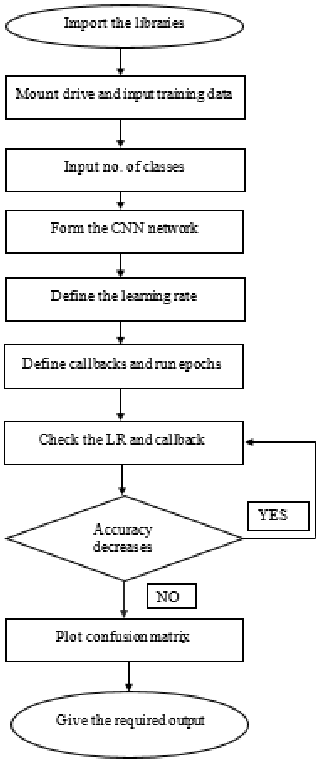


Fig. 8. Flowchart of Single Object Detection.

This algorithm can be described as supervised classification algorithm. Data flows through CNN layers and various operations are performed on data. The learning rate and callbacks are defined. Number of epochs and batch size is also defined. The epochs are then executed through which algorithm learns through training data. Training accuracy and training losses are constantly monitored. If training accuracy starts falling below a threshold, the callback function is invoked and epochs are stopped. Confusion matrix is then plotted using training and testing data. Various performance parameters can be defined and observed using the confusion matrix.

C. Multiple Object Detection

Fig. 9 describes working of YOLOv3 multiple object detection algorithm. An image is given as the input to algorithm and transformation is done using CNN. These transformations are done so that, input image is compatible to specifications of algorithm. Following this, flattening operation is performed. Flattening is converting data into a 1-dimensional array for inputting it to next layer. Flattening of output of convolutional layers is to create a single long feature vector and it is connected to final classification model, which is called a fully connected layer. By changing the score threshold, one can adjust how the ML model assigns these labels.

Object detection pipeline has one component for generating proposals for classification. Proposals are nothing but candidate regions for object of interest. Most of approaches employ a sliding window over feature map and assigns foreground/background scores depending on features computed in that window. The neighborhood windows have similar scores to some extent and are considered as candidate regions. This leads to hundreds of proposals. As the proposal generation method should have high recall, we keep loose constraints in this stage. However, processing these many proposals all through the classification network is cumbersome. This leads to a technique, which filters proposals based on some criteria called Non-Maximum Suppression. IOU calculation is actually used to measure the overlap between two proposals.

D. Multiple Object Tracking

In multiple object tracking, train the vehicle tracker using YOLOv3 and deep learning methods and optimize the detector's success rate by providing efficient vehicle detection results by testing trained vehicle detector on test data [17]. It consists of six phases such as loading data set, YOLOv3 design, training options configuration, object tracker training, and tracker evaluation, respectively. Flow chart of multiple object detection is shown in Fig. 10.

E. Performance Metrics

The trained model using deep learning must be evaluated for its performance on unseen data called as test dataset. The choice of performance metrics will influence the analysis of algorithms. This helps in identifying reasons for mis-classifications so that it can be corrected by taking necessary measures.

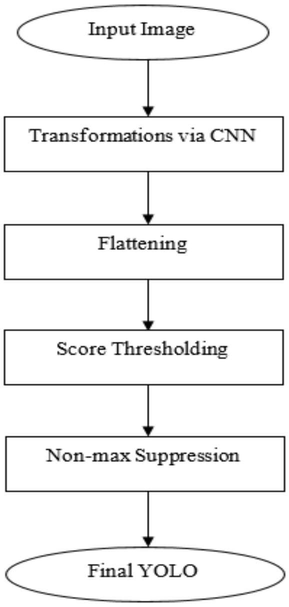


Fig. 9. Flowchart for Multiple Object Detection.

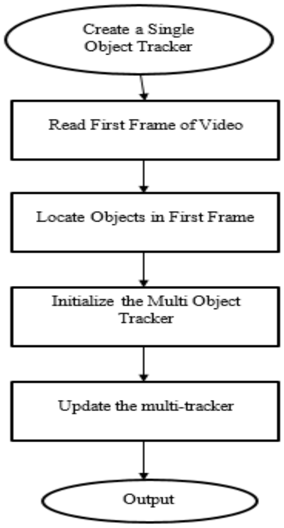

Create a Single Object Tracker
Read First Frame of Video
Locate Objects in First Frame
Initialize the Multi Object Tracker
Update the multi-tracker
Output

Fig. 10. Flow Chart of Multiple Object Tracking.

1) Confusion Matrix: It gives prediction information of various objects for binary classification as shown in Table V.

2) Accuracy and Loss: Accuracy measure is calculated by using formula $\frac{TP+TN}{TP+TN+FP+FN}$. The accuracy measure, as a stand-alone measure is not reliable since it gives equal costs for both type of errors and works well for a well-balanced dataset. The loss is calculated by loss functions of used for training, and average of the loss is calculated when used batch learning that computes loss after each training each batch.

3) Precision, Recall and F1- score: Precision is the percentage of classification results that are relevant. Recall is the percentage of total relevant results that are classified correctly by algorithm. F-1 score considers both precision and recall values hence must be maximized to make the model better.

TABLE. V. CONFUSION MATRIX FOR BINARY CLASSIFICATION

	Predicted class - 1	**Predicted class - 2**
Actual Class - 1	TP - True Positive Decision is correct	FN - False Negative Error – Type 1
Actual – Class 2	FP - False Positive Error – Type 2	TN - True Negative Decision is correct

The formulas to calculate these metrics are

$$Precision = \frac{TP}{TP+FP} \quad (9)$$

$$Recall = \frac{TP}{TP+FN} \quad (10)$$

$$F-1\,Score = 2 * \frac{Preciison*Recall}{Precision+Recall} \quad (11)$$

$$mAP = \frac{1}{No.of\ divisions} \sum_{r\text{€}(1,0.1,0.001)} p_{interp}(r) \quad (12)$$

The detected objects are bounded with bounding box. Tracking is performed on frames of the videos to identify objects in the successive frames using SORT. The evaluation metrics like True Positive (TP), True Negative (TN), False Positive (FP), False Negative (FN) thereby Precession, Recall and hence mean Average Precession (mAP) is calculated using Intersection over Union (IoU).

F. *Specifications used for Implementation*

Single object detection

Dataset used – On-Road Vehicle Dataset [11]

TABLE. VI. IMAGES USED FOR SINGLE OBJECT DETECTION

Number of classifiers	Three (Autos, Heavy, Light)
Total number of Input Images	12480
Training images	9984
Testing images	2496
Day images	7590
Evening images	4518
Night images	372

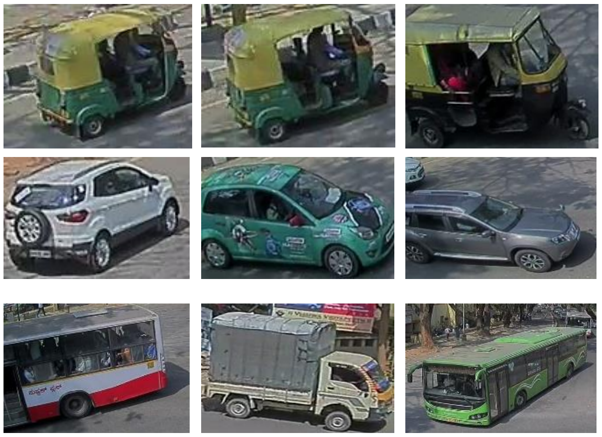


Fig. 11. Sample Day Images.

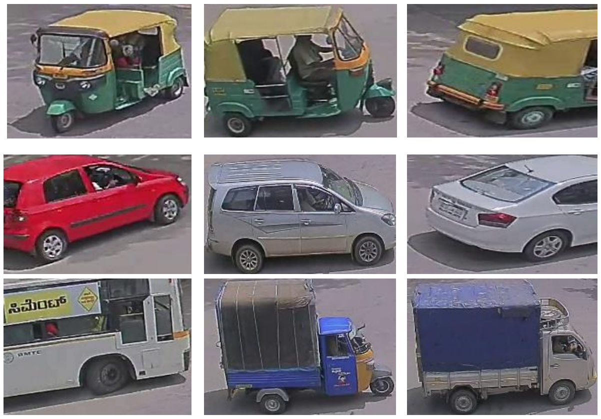


Fig. 12. Sample Evening Images.

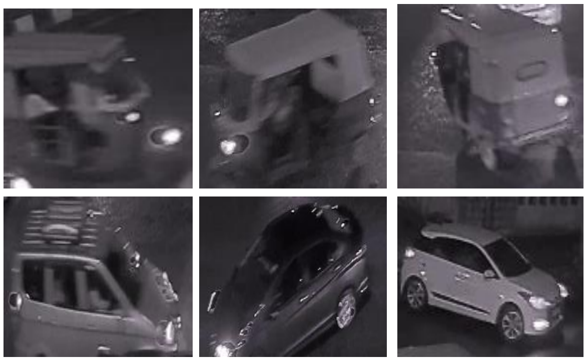


Fig. 13. Sample Night Images.

Table VI. Shows total number of images considered for implementation. In this dataset each image contains one object. Further images are captured in different intervals of time, such as day (7590), evening (4518) and night (372); sample images are shown in Fig. 11, 12 and 13 and is tabulated in Table VI.

1) Multiple object detection: Dataset used – KITTI Dataset. All the images are captured in daytime. Dataset contains 80 classes, five classes such as car, bus, truck, motor cycle and Train classes are implemented in this paper. All the images are captured in day time. Total number of images used for implementation 11682, tabulated in Table VII. Sample images of dataset are shown in Fig. 14. Each image contains multiple objects.

TABLE. VII. IMAGES USED FOR MULTIPLE OBJECT DETECTION

Number of classifiers	80
Classifiers used	5 (Car, Bus, Truck, Motor Cycle and Train)
Total number of Input Images	11682
Training images	9736
Testing images	1946

Fig. 14. Sample Images of KITTI Object Detection Dataset.

V. Simulation Results and Analysis

This section describes simulation results and performance parameters observed are accuracy, precision and recall. It also underlines the confusion matrices of different datasets and convolution layers of the algorithms.

A. Single Object Detection

CNN is designed for single object detection. The layers and each layer information as shown in Fig. 15.

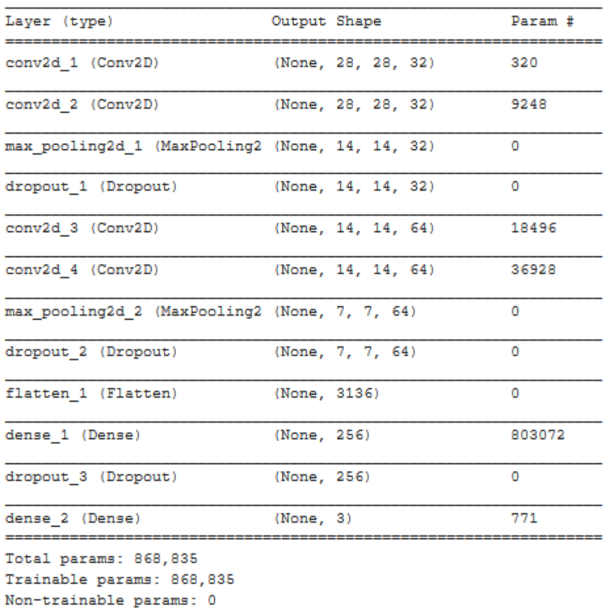
It encompasses the parameters that were included in each step, layer progression and output image size of every layer. Each layer divides the image matrix into its components and performs an operation on image. The output image size of various layers is different due to manipulations by each layer such as initially the output image size is 28×28 which then reduces to 14×14 due to the max pooling layer which chooses the max valued pixel from the surrounding pixels. It then reduces to 7×7 due to the second max pooling layer. This pixel is then flattened into 7×7×64 which are 3136 sized vector. This vector is then reduced to a less sized vector by proceeding layers and final calculation parameters are displayed.

Designed neural network was trained and tested. Obtained training accuracy and loss as shown in Fig. 16 and 17. Obtained 82% training accuracy through training this model. The loss and accuracy are inversely proportional to each other. As the number of epochs increases, learning rate increases and hence loss decreases. Each time epochs is run, the model trains it itself and weights of the convolution networks gets updated to a more accurate value.

The CNN is successfully able to classify the given object as truck and car with an accuracy of 75.68% and 84.409% respectively as shown in Fig. 18.

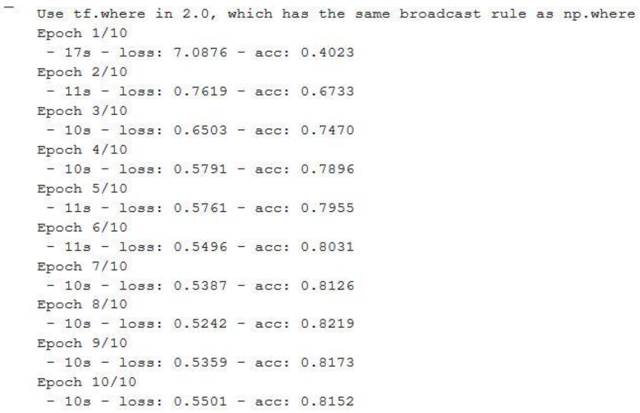
Upon simulation, it is able to correctly classify the vehicles by classifying a car with 79.853% accuracy and about 78.122% accuracy for the detection of an auto as shown in Fig. 19.

Upon simulation, it is able to correctly classify the vehicles by classifying it into a car with about 79.036 % accuracy and auto with about 80.064 % accuracy as shown in Fig. 20.



```
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_1 (Conv2D)            (None, 28, 28, 32)        320
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 28, 28, 32)        9248
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 14, 14, 32)        0
_________________________________________________________________
dropout_1 (Dropout)          (None, 14, 14, 32)        0
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 14, 14, 64)        18496
_________________________________________________________________
conv2d_4 (Conv2D)            (None, 14, 14, 64)        36928
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 7, 7, 64)          0
_________________________________________________________________
dropout_2 (Dropout)          (None, 7, 7, 64)          0
_________________________________________________________________
flatten_1 (Flatten)          (None, 3136)              0
_________________________________________________________________
dense_1 (Dense)              (None, 256)               803072
_________________________________________________________________
dropout_3 (Dropout)          (None, 256)               0
_________________________________________________________________
dense_2 (Dense)              (None, 3)                 771
=================================================================
Total params: 868,835
Trainable params: 868,835
Non-trainable params: 0
```

Fig. 15. Convolution Layers used in CNN.



```
Use tf.where in 2.0, which has the same broadcast rule as np.where
Epoch 1/10
 - 17s - loss: 7.0876 - acc: 0.4023
Epoch 2/10
 - 11s - loss: 0.7619 - acc: 0.6733
Epoch 3/10
 - 10s - loss: 0.6503 - acc: 0.7470
Epoch 4/10
 - 10s - loss: 0.5791 - acc: 0.7896
Epoch 5/10
 - 11s - loss: 0.5761 - acc: 0.7955
Epoch 6/10
 - 11s - loss: 0.5496 - acc: 0.8031
Epoch 7/10
 - 10s - loss: 0.5387 - acc: 0.8126
Epoch 8/10
 - 10s - loss: 0.5242 - acc: 0.8219
Epoch 9/10
 - 10s - loss: 0.5359 - acc: 0.8173
Epoch 10/10
 - 10s - loss: 0.5501 - acc: 0.8152
```

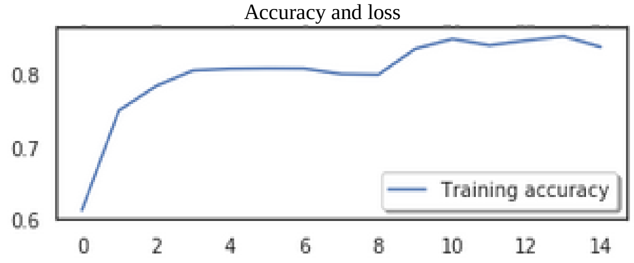


Fig. 16. Training Accuracy.

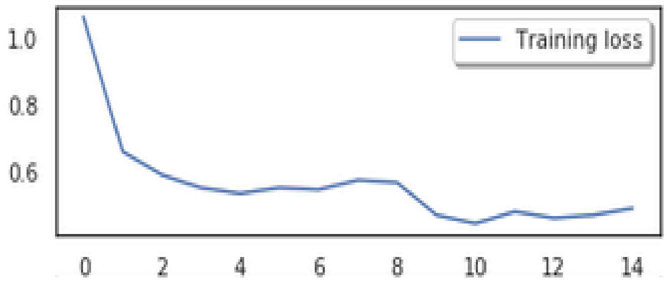


Fig. 17. Training Loss.

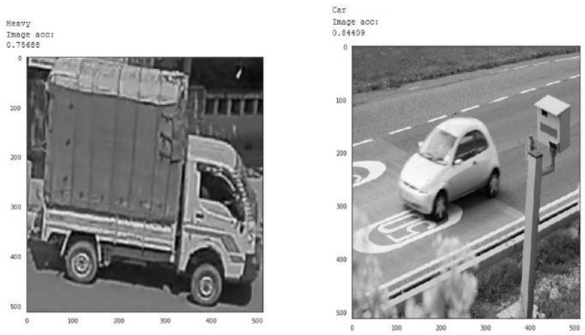


Fig. 18. Sample Simulation Results of Day Images.

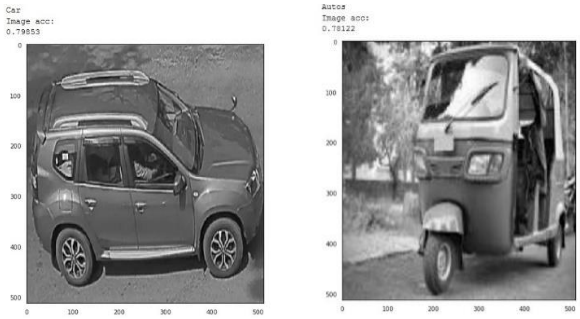


Fig. 19. Sample Simulation Results of Evening Images.

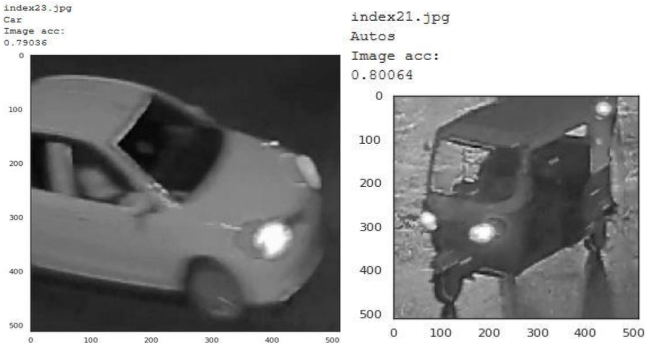


Fig. 20. Sample Simulation Results of Night Images.

Confusion matrix for day images is tabulated in Table VIII. The performance parameters are extracted from confusion matrix and tabulated in Table IX. Accuracy, precision and recall data is evident for autos, cars and heavy type of vehicles as shown in Fig. 21. The accuracy of autos and cars is almost identical while that of heavy vehicles is slightly better than that of others. Since the number of training images is more for the Day images, the results obtained are better than that of Evening and Night Dataset images. High precision indicates that, the algorithm returned substantially more relevant results than irrelevant ones while high recall means that an algorithm returned most of the relevant results.

TABLE. VIII. CONFUSION MATRIX FOR DAY IMAGES

PREDICTED	Autos	cars	Heavy	All
Autos	2369	117	44	2530
cars	66	2413	70	2530
Heavy	95	89	2346	253

TABLE. IX. PERFORMANCE METRICS OF DAY IMAGES

	TP	TN	FP	FN	Precision	Accuracy	Recall
Auto	2369	4918	161	161	0.936	0.957	0.936
Cars	2413	4853	206	136	0.921	0.955	0.946
Heavy	2346	4965	114	184	0.953	0.960	0.927

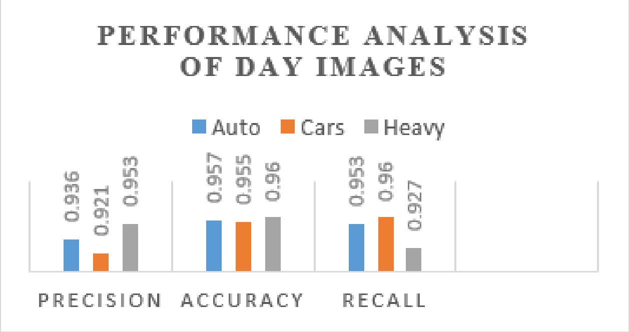


Fig. 21. Performance Analysis of Day Images.

Confusion matrix for evening images is tabulated in Table X. The performance parameters are extracted from confusion matrix and tabulated in Table XI. As shown in Fig. 22, accuracy of evening images has been reduced. This can be accounted due to decrease in number of training images given to the neural network. Because of this, weights may not be as accurate as the day dataset.

TABLE. X. CONFUSION MATRIX FOR EVENING IMAGES

PREDICTED	Autos	cars	Heavy	All
Autos	1456	28	22	1506
cars	16	1480	10	1506
Heavy	42	34	1430	1506

TABLE. XI. PERFORMANCE METRICS OF EVENING IMAGES

	TP	TN	FP	FN	Precision	Accuracy	Recall
Auto	1456	2954	58	50	0.961	0.916	0.966
Cars	1480	2950	62	26	0.959	0.920	0.982
Heavy	1430	2980	32	76	0.978	0.906	0.949

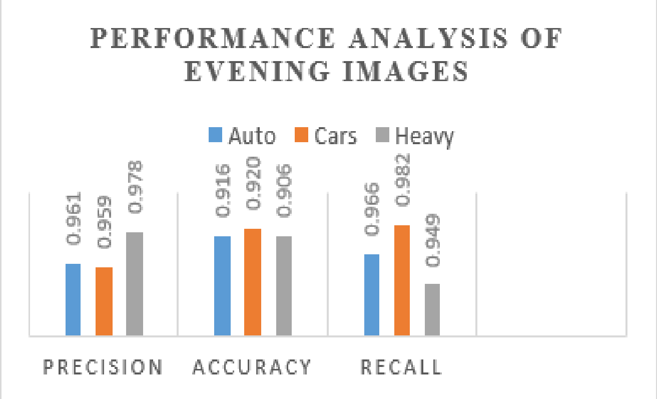


Fig. 22. Performance Analysis of Evening Images.

Confusion matrix for night images is tabulated in Table XII. The performance parameters are extracted from confusion matrix and tabulated in Table XIII. Accuracy of autos, cars and heavy has decreased considerably as shown in Fig. 23. The number of training images is low. Since the Neural Network has received less training images, the weights that are calculated are not very precise. Besides this, illumination of the image plays a major role in Object detection. Since night images have low illumination levels, the accuracy of prediction of class of image is low.

TABLE. XII. CONFUSION MATRIX FOR NIGHT IMAGES

PREDICTED	Autos	cars	Heavy	All
Autos	108	4	12	124
cars	14	100	10	124
Heavy	2	16	106	124

TABLE. XIII. PERFORMANCE METRICS OF NIGHT IMAGES

	TP	TN	FP	FN	Precision	Accuracy	Recall
Auto	108	232	16	16	0.87	0.913	0.87
Cars	100	228	20	24	0.833	0.881	0.806
Heavy	106	226	22	18	0.828	0.892	0.854

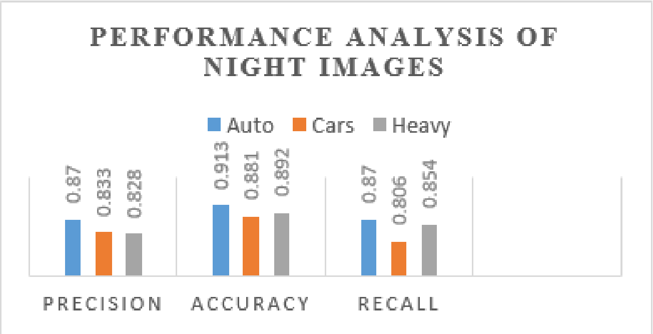


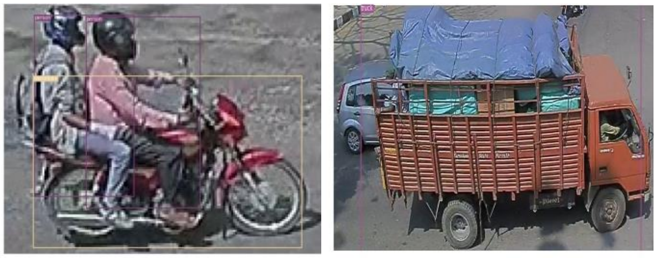
Fig. 23. Performance Analysis of Night Images.

B. Multiple Object Detection

The images collected from [11] have been given as test set. The dataset consisting of three different kinds of images such as day, evening and NIR images (Near Infrared). Object detections by YOLOv3 as shown in Fig. 24, 25 and 26.

The result in Fig. 27 shows algorithm can detect objects of any size and images captured from various camera angle and distance. This attribute is because of FPN used in YOLOv3. Thinner intact bounding boxes as shown in Fig. 28 ensures not to miss out any of the minute details and a greater IoU.

Image considered for performance analysis is as shown in Fig. 29. Performance metrics for car detection is tabulated in Table XIV. Image has various kinds of objects. The precision result is high since objects to be detected are not occluding one another and the recall is 0.8333 since the FN value is 1 as the auto in the image is detected but identified as truck. As result of misclassification, the recall value suffers a loss. Since the precision is high, the mAP value is 100% for given class and considered image.



For on road vehicle dataset [11].

Fig. 24. RGB_Day Image Detections.

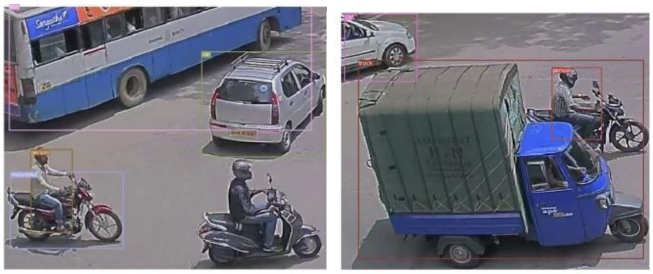


Fig. 25. RGB_Evening Images Detections.

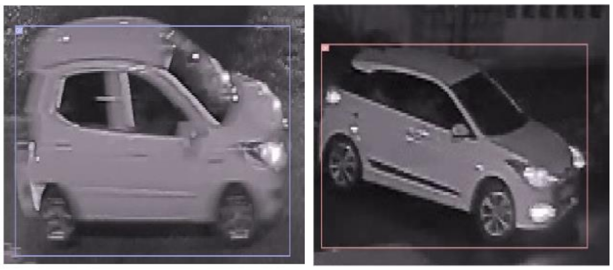


Fig. 26. NIR Images Detections.

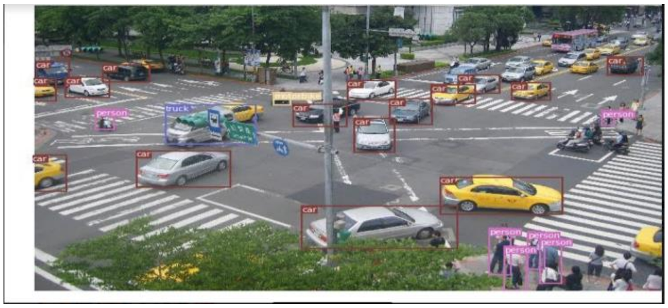


Fig. 27. YOLOv3 Results for COCO Dataset.

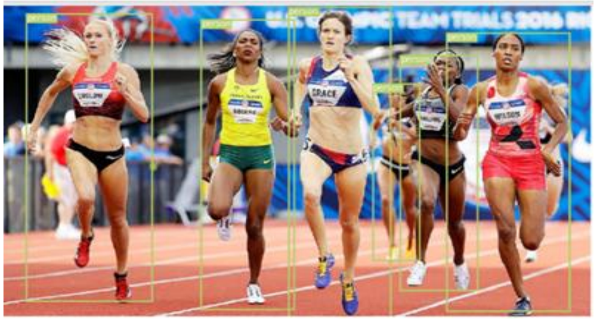


Fig. 28. YOLOv3 Results with Intact Boxes.

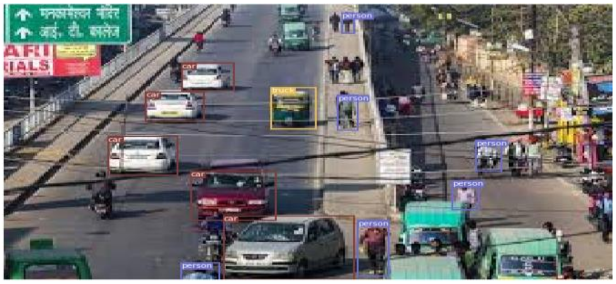


Fig. 29. Car Detection.

TABLE. XIV. PERFORMANCE METRICS FOR CAR DETECTION

	TP	TN	FP	FN	Precision	Recall
IoU at.25	5	0	0	1	1	0.8333
IoU at .50	5	0	0	1	1	0.8333
IoU at.75	4	0	0	1	1	0.8333

Image considered has different kinds of objects as shown in Fig. 30 and its performance metrics tabulated in Table XV. Precession result is varied with respect to different IoU's since variations in IoU will results in variations with respect to ground truth boxes. Greater the IoU lesser the detections and hence precession values incur loss. For given class and considered image mAP value is 78.57%. The decrease in mAP value is because of varied precision value.

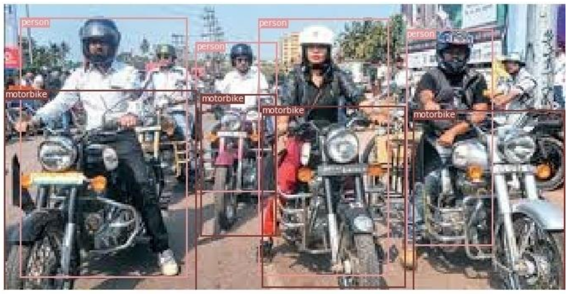


Fig. 30. Motorbike Detection.

TABLE. XV. PERFORMANCE METRICS FOR MOTORBIKE DETECTION

	TP	TN	FP	FN	Precision	Recall
IoU at.25	6	0	1	0	0.8571	1
IoU at .50	5	0	1	0	0.8333	1
IoU at.75	4	0	2	0	0.6667	1

Object detection in Video: video specifications

Time duration of video = 27 seconds

Type of file = MP4 File (.mp4)

Size = 1.65 MB (17,33,851 bytes)

Size on disk = 1.65 MB (17,36,704 bytes)

Number of Frames = 27*30 = 810

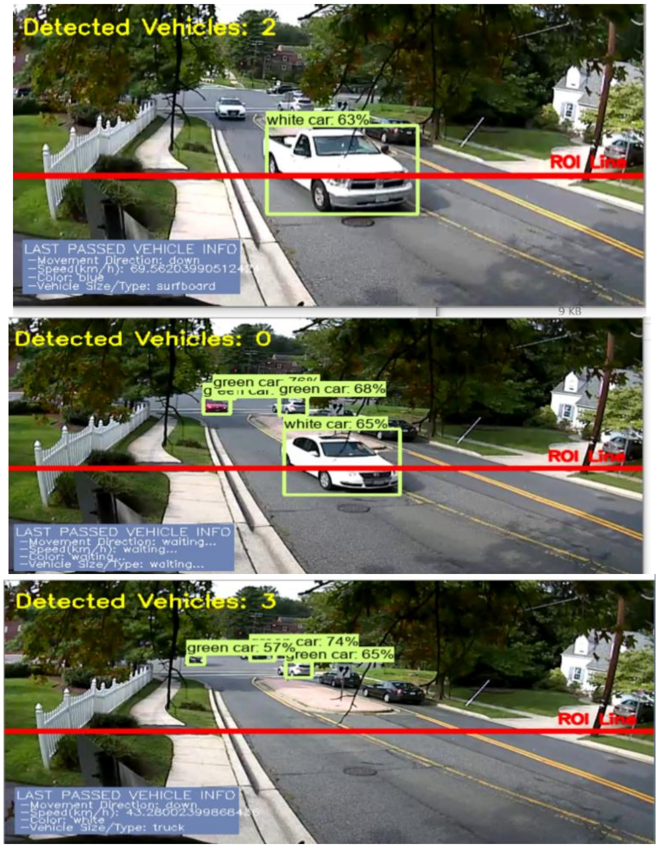


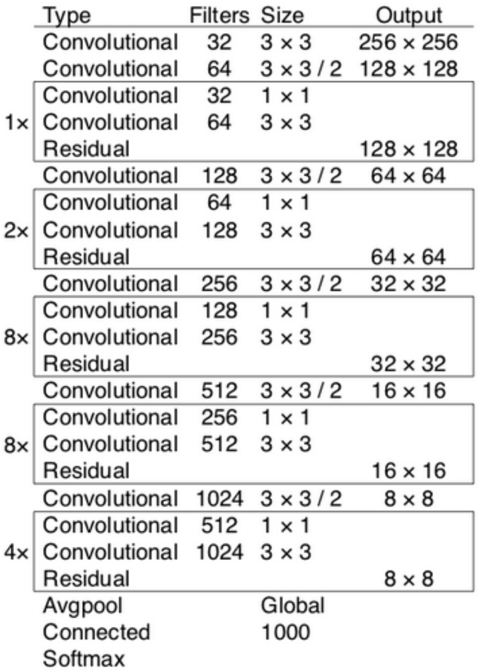
Fig. 31. Results of Object Detection in Video based on Region of Interest (ROI).

Fig. 31 shows detection of objects in video. In addition, the parameters measured are speed and color of vehicle, vehicle type, direction of vehicle movement and number of vehicles in ROI.

1) For KITTI dataset: All images of on road vehicle dataset contains singe object in each image. Hence, for multiple object detection COCO and KITTI vehicle detection dataset is used for simulation. In this dataset, each image contains multiple objects of same class or multiple objects of different class. Objects are detected using YOLOv3 algorithm. Neural network layers information as shown in Fig. 32.

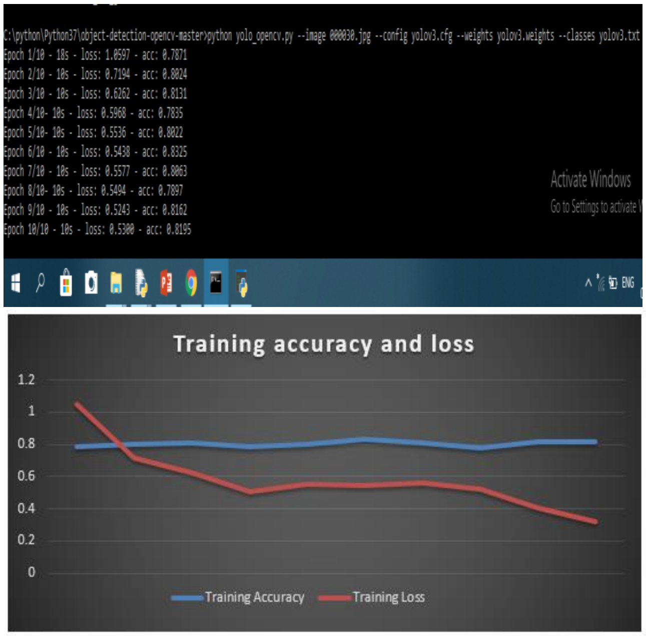
2) Convolution Layers: The convolutional layers of YOLOv3 algorithm when stacked are formed. It contains of 53 convolutional layers, each followed by batch normalization layer and Leaky ReLU activation. No form of pooling is used, and a convolutional layer with stride 2 is used to down-sample the feature maps. This helps in preventing loss of low-level features often attributed to pooling. At the end, all operations performed by the convolutional layers, average pooling and softmax operations are completed.

KITTI dataset contains 80 classes of objects. In this paper, five classes such as cars, truck, bus, train, and motorcycle objects images are considered for simulation. Fig. 33 shows training accuracy and loss values of YOLOv3 for multiple object detection. Fig. 34, 35, 36, 37, 38, 39, 40 and 41 shows multiple object detection using YOLOv3. The YOLOv3 algorithm has successful detected and classified the objects as Car, Truck, Train, Motorcycle. A total of 11,682 images were used from KITTI dataset where 9736 was used for training and 1946 was allocated for testing. An important parameter considered here is IOU which describes the overlap factor of one detected object from the other. It is seen that as the number of vehicles in the image increases the bounding boxes are overlapped.



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

Fig. 32. Convolutional Layers used in YOLOv3.



```
C:\python\Python37\object-detection-opencv-master>python yolo_opencv.py --image 000030.jpg --config yolov3.cfg --weights yolov3.weights --classes yolov3.txt
Epoch 1/10 - 18s - loss: 1.0597 - acc: 0.7871
Epoch 2/10 - 10s - loss: 0.7194 - acc: 0.8024
Epoch 3/10 - 10s - loss: 0.6262 - acc: 0.8131
Epoch 4/10- 10s - loss: 0.5968 - acc: 0.7835
Epoch 5/10- 10s - loss: 0.5536 - acc: 0.8022
Epoch 6/10 - 10s - loss: 0.5438 - acc: 0.8325
Epoch 7/10 - 10s - loss: 0.5577 - acc: 0.8063
Epoch 8/10- 10s - loss: 0.5494 - acc: 0.7897
Epoch 9/10 - 10s - loss: 0.5243 - acc: 0.8162
Epoch 10/10 - 10s - loss: 0.5300 - acc: 0.8195
```

Fig. 33. Training Accuracy and Loss Values of YOLOv3 for Multiple Object Detection.

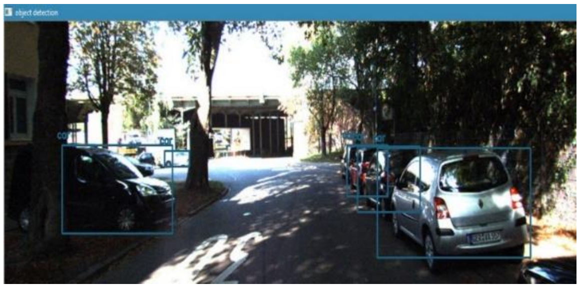


Fig. 34. Multiple Object Detection – Cars.

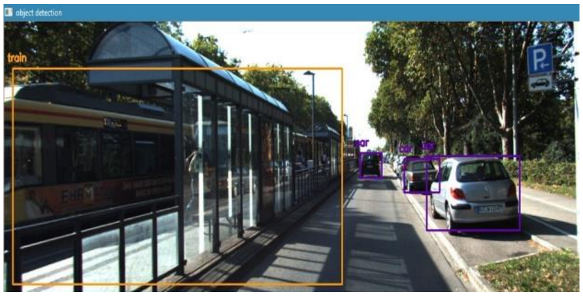


Fig. 35. Multiple Object Detection –Cars and Train.

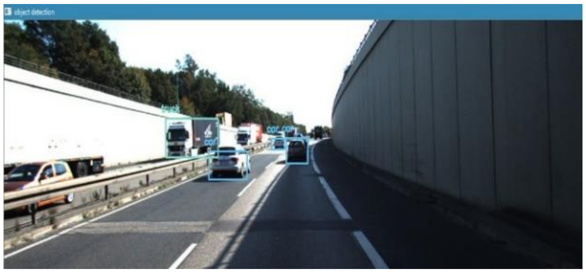


Fig. 36. Multiple Object Detection –Cars and Truck.

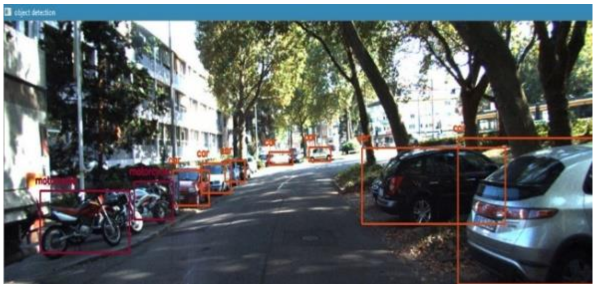


Fig. 37. Multiple Object Detection – Cars and Motor Cycle.

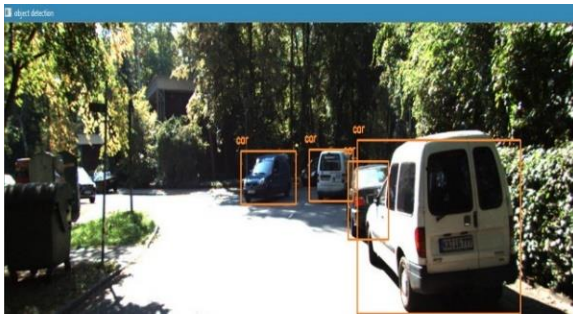


Fig. 38. Multiple Object Detection – Four cars are detected and two cars are not detected.

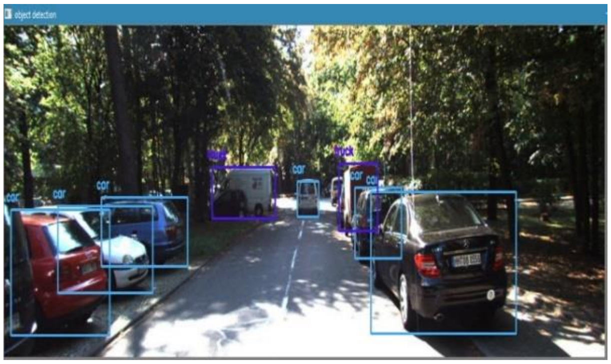


Fig. 39. Multiple Object Detection – All types of cars are detected.

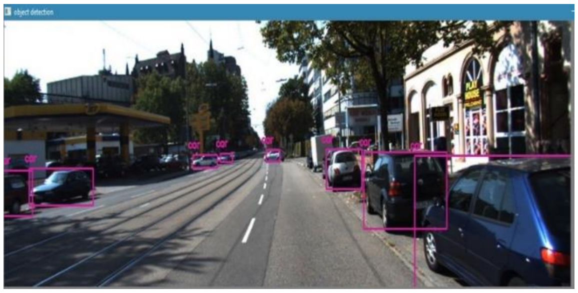


Fig. 40. Multiple Object Detection – Two cars inside petrol bunk not detected.

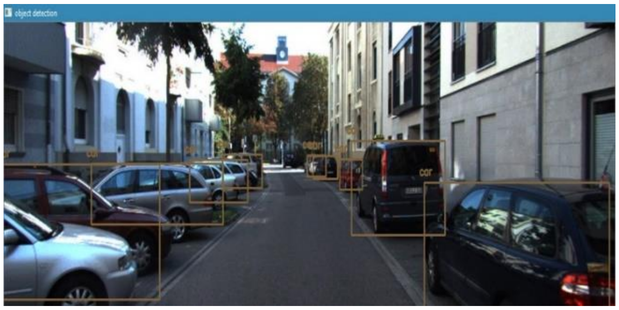


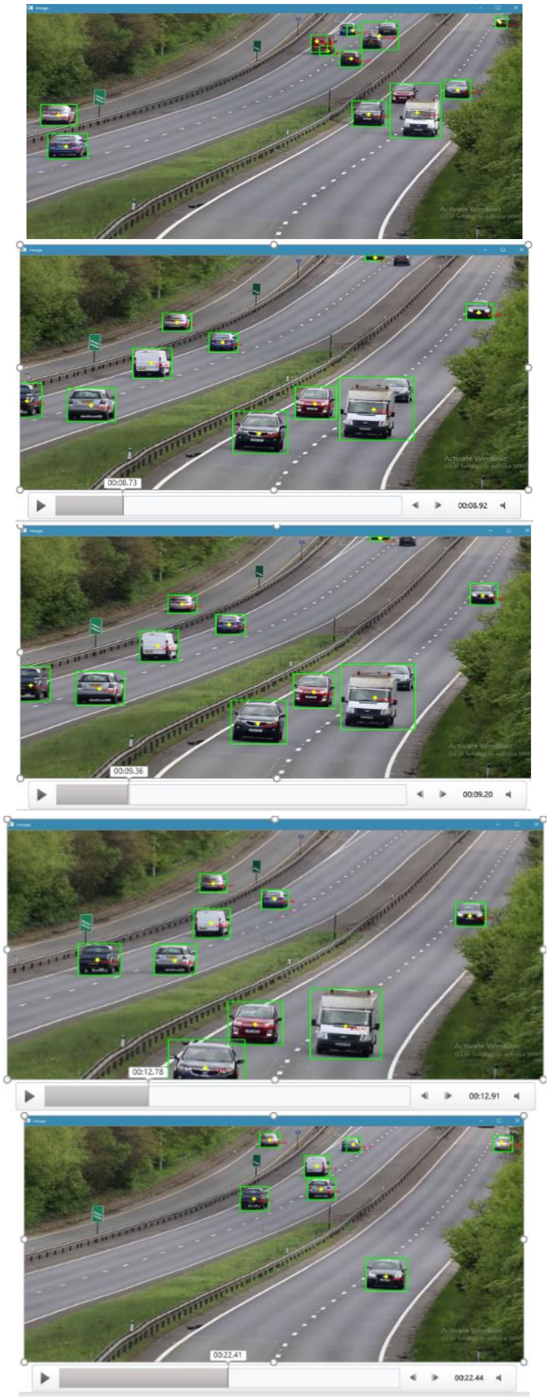
Fig. 41. Multiple Object Detection – All cars are detected except one car.

C. Multiple Object Tracking for Trafic Surveillance Video Dataset Specifications

Name – A3 Road Traffic UK HD – rush hour – British Highway Traffic May, 2017
Format - Mp4
Size - 12.36Mb
Time frame - 245 sec
Video Quality – 720p
Types of objects – Cars and trucks

Fig. 42 shows the images of multiple object tracking for surveillance video, it contains cars and trucks. The vehicle tracker trained on surveillance video using YOLOV3 and deep learning methods. The vehicle tracking process was successfully carried out by testing trained vehicle detector on test data set video. Algorithm divided the video into frames with a rate of 30fps and performed object detection in first frame. In the latter frames, the particular detected image was tracked using its centroid position. Objects are tracked in different frames at different intervals of time as shown in Fig. 43.

Fig. 42. Multiple Object Tracking Snippet.



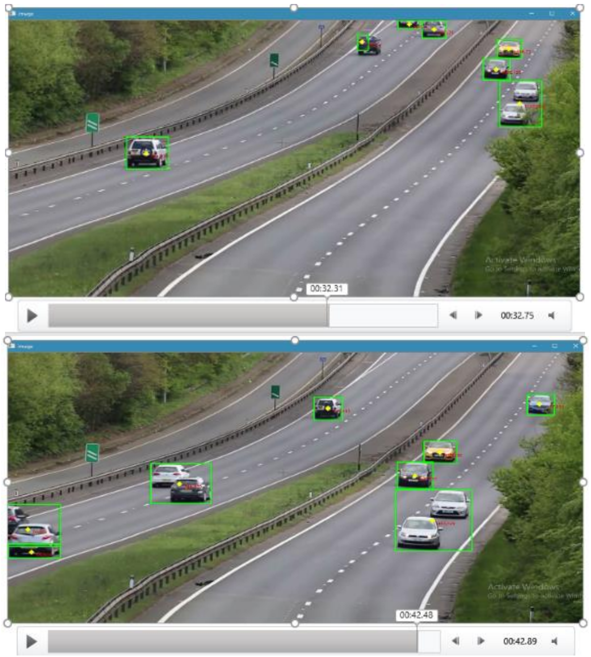


Fig. 43. Images of Multiple Object Tracking at different Frames.

VI. Conclusions

The inclusion of Artificial Intelligence to solve Computer vision tasks has outperformed the image processing approaches of handling the tasks. The CNN model trained to on road vehicle dataset for single object detection, achieved a validation accuracy of 95.7 % for auto, 95.5% for car and 96 % for heavy vehicles for day images. The high validation accuracy is because of huge amount of data on which it is trained from each class. Performance metrics are tabulated for day, evening and NIR images. Multiple object detection is implemented using YOLOv3 for KITTI and COCO dataset. Performance metrics is tabulated for YOLOv3 on considered classes of images. Higher the precession value of class greater will be mAP value. The mAP value depends on image chosen for calculation. IoU of 0.5 is ideal for detection and tracking. mAP values can be enhanced by increasing true positive values. Results of performance metrics is totally dependent on image data set used. Further objects are detected in video based on region of interest. The performance measures measured such as speed and color of vehicle, type of vehicle, direction of vehicle movement and the number of vehicles in ROI. Multiple object tracking is implemented for traffic surveillance video using YOLOv3 and OpenCV. Multiple objects are detected and tracked on different frames of a video. Further training the models on powerful GPUs and by increasing the number of images evaluate the models on other datasets and modify the design if required to make the model more robust and suitable for real-time applications.

References

[1] V. D. Nguyen et all., “Learning Framework for Robust Obstacle Detection, Recognition, and Tracking”, IEEE Transactions on Intelligent Transportation Systems, vol. 18, no. 6, pp. 1633-1646, June 2017.

[2] Zahraa Kain et all, “Detecting Abnormal Events in University Areas”, 2018 International conference on Computer and Applications(ICCA),pp. 260-264, 2018.

[3] P. Wang et all., “Detection of unwanted traffic congestion based on existing surveillance system using in freeway via a CNN-architecture trafficnet”, IEEE Conference on Industrial Electronics and Applications (ICIEA), Wuhan, 2018, pp. 1134-1139.

[4] Q. Mu, Y. Wei, Y. Liu and Z. Li, “The Research of Target Tracking Algorithm Based on an Improved PCANet”, 10th International Conference on Intelligent Human-Machine Systems and Cybernetics (IHMSC), Hangzhou, 2018, pp. 195-199.

[5] H. C. Baykara et all., “Real-Time Detection, Tracking and Classification of Multiple Moving Objects in UAV Videos”, 29th IEEE International Conference on Tools with Artificial Intelligence (ICTAI), Boston, MA, 2017, pp. 945-950.

[6] W. Wang, M. Shi and W. Li, “Object Tracking with Shallow Convolution Feature”, 9th International Conference on Intelligent Human-Machine Systems and Cybernetics (IHMSC), Hangzhou, 2017, pp. 97-100.

[7] K. Muhammad et all., “Convolutional Neural Networks Based Fire Detection in Surveillance Videos”, IEEE Access, vol. 6, pp. 18174-18183, 2018.

[8] D. E. Hernandez et all., “Cell Tracking with Deep Learning and the Viterbi Algorithm”, International Conference on Manipulation, Automation and Robotics at Small Scales (MARSS), Nagoya, 2018, pp. 1-6.

[9] X. Qian et all., “An object tracking method using deep learning and adaptive particle filter for night fusion image”, 2017 International Conference on Progress in Informatics and Computing (PIC), Nanjing, 2017, pp. 138-142.

[10] Y. Yoon et all., “Online Multi-Object Tracking Using Selective Deep Appearance Matching”, IEEE International Conference on Consumer Electronics - Asia (ICCE-Asia), Jeju, 2018, pp. 206-212.

[11] H. S. Bharadwaj, S. Biswas and K. R. Ramakrishnan. “A large scale dataset for classification of vehicles in urban traffic scenes”, Proceedings of the Tenth Indian Conference on Computer Vision, Graphics and Image Processing,ACM,2016.

[12] Mohana et.al., “Performance Evaluattion of background modeling methods for object Detection and Tracking”,International Conference on Inventive systems and Control (ICISC).

[13] G. Chandan et.al., “Real Time Object Detection and Tracking Using Deep Learning and OpenCV”, International Conference on Inventive Research in Computing Applications (ICIRCA), 2018.

[14] Mohana et.al., “Elegant and efficient algorithms for real time object detection, counting and classification for video surveillance applications from single fixed camera” International Conference on Circuits, Controls, Communications and Computing (I4C),2016.

[15] Mohana et.al., “Simulation of Object Detection Algorithms for Video Survillance Applications”, 2nd International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud),2018.

[16] A. Raghunandan et. al., "Object Detection Algorithms for Video Surveillance Applications," International Conference on Communication and Signal Processing (ICCSP), 2018.

[17] A. Mangawati et al., “Object Tracking Algorithms for Video Surveillance Applications,” 2018 International Conference on Communication and Signal Processing (ICCSP), 2018.

[18] Mohana, et al., “Design and Implementation of Object Detection, Tracking, Counting and Classification Algorithms using Artificial Intelligence for Automated Video Surveillance Applications” Advanced Computing and Communication Society (ACCS)- 24th annual International Conference on Advanced Computing and Communications (ADCOM-2018), IIITB, Bangalore.