An empirical performance analysis for on road object detection from Traffic videos: An image pre-processing perspective

Shubham Garg, Saurabh Pandey, Sarthak Kaushal, Anuradha Dhull*, Yogita Gigras

Department of Computer Science and Engineering, The NorthCap University, Gurugram, India
*Corresponding Author E-mail Id: *anuradha@ncuindia.edu*

Abstract: This paper presents an empirical performance analysis of various object detection algorithms for the identification of on road objects from the perspective of different pre-processing techniques. The analysis is done on some real time traffic videos data captured from CCTV camera. The pre-processing techniques considered for analysis are background subtraction, denoising and smoothing methods. For background subtraction, two popular algorithms named Temporal Median Filtering and Canny Edge Detection are being utilized. The Temporal Median Filtering outputs the mask of the objects of interest by eliminating the background and Canny Edge Detection draws the outline of the objects of interest. Gaussian Blur technique is used for image smoothing and noise removal from traffic video. These transformations are applied on test videos of on road vehicle traffic collected manually and the results are obtained using state of the art methods. Apart from this analysis, a new training-testing model for quality object detection under different light conditions (day light, night and low vision) have also been proposed. The main idea here is to feed the detectors with images having less but meaningful features like object boundaries. This paper also presents a comparative analysis on state-of-the-art object detection algorithms such as YOLO and Mask-RCNN using transfer learning concept. Moreover, this paper also compares the performance of a segmentation model when fed with the data labelled with bounding boxes instead of the pixel level segmentation.

Keywords: YOLO, Mask-RCNN, Traffic Surveillance, Image pre-processing

I. INTRODUCTION

With a rapid growth in the number of vehicles on road, traffic monitoring has become a challenging issue. Traffic regulation is becoming more and more challenging because of the growing traffic. Proper analysis and planning are required to handle this large traffic data efficiently and effectively. Modern surveillance systems built using Computer Vision techniques are more configurable and robust as the data collected using these systems helps in detection and classification of the vehicles in the best possible way [1]. Object detection has applications in many fields such as video monitoring, unusual behaviour detection, vehicle detection, accident detection etc. The quality of videos captured plays an important role in better identification of objects.

In this paper, the real time traffic videos have been used for analysis to enable effective use of background subtraction techniques by successfully estimating the background as we would not encounter a scenario in the dataset with a different background. The idea behind background subtraction is to show the detector some specific and meaningful features only. For instance, with Canny Edge Detection, showing the detector only the object boundaries. It will help the trained model to generalise and detect objects by looking at their boundaries only.

The state-of-the art algorithms like YOLO and Mask R-CNN use different techniques to detect the objects in an image. Also, the input dataset needs to be labelled differently based on the algorithm.

Mask R-CNN being a segmentation model inputs the data segmented at pixel level whereas the annotations for YOLO take the form of bounding boxes. A comparative analysis to find out to what extent pixel level segmentation performs better than labelling the objects with bounding boxes is required. A comparison of detection accuracy is evaluated by fine tuning the models using these algorithms on the dataset. As pixel level segmentation is a very time-consuming task, this paper presents an approach to use bounding box data annotations with segmentation models like Mask R-CNN thereby saving a lot of time.

Rest of the paper is divided into the following subsections. Section 2 presents the literature review; Section 3 is about the proposed analysis along with detailed description over the real video dataset captured for analysis and presents a new research methodology proposed for object detection for improvised results under different light conditions. Section 4 illustrates the results. Section 5 concludes the findings and discussion about the future scope.

II. LITERATURE REVIEW

In the past decade, advancements in computer vision have been consistent and ground-breaking. Various solutions to deal with the problem of traffic are in use or are being developed. Companies like Tesla [2] are using object detection for many solutions. However, some of these solutions are not public. The solution proposed in this paper uses input from three cameras placed at different locations to capture a wider view. Background subtraction is a go to technique for scenarios having a single camera for input stream. Implementing such a technique is difficult and time consuming [3].

Object detection, traditionally, involves machine learning algorithms which would extract features and input them into classifier such as SVM based classifiers, Bayesian classifiers, Decision trees, etc [4]. In these classifier-based algorithms, the feature extraction is done by Scale Invariant feature transform (SIFT) proposed by Zhao et al. (2012) [5] and histogram of oriented gradient (HOG) proposed by Dalal et al. (2005) [6]. But these methods lack in certain areas and results in low accuracy of edge feature extraction is not significantly high.

Traditional approaches use detectors based on SVMs which are Haar cascade classifiers or sliding window methods. The deep learning models such as deep convolutional neural networks have better accuracy than simple machine learning classifiers. Deep learning technology brought a revolution in object detection because of its powerful feature extraction ability. Ross Girshick et al. (2014) [7] proposed the RCNN model which extracted subregions from input and a convolution neural network (CNN) was used to extract sub-regional features. Lou et al. (2007) [8] proposed a technique called non-maximal suppression, which finds the optimality of a region in an image. Ross Girshick et al. (2015) [9] proposed Faster R-CNN, which was an extension of RCNN which had increased speed of detection and as well as increased accuracy. Faster R-CNN combined the region classification with the bounding box regression as a multitask problem into a single network.

In 2016, Redmond et al. proposed YOLO [10] which is another convolutional network-based algorithm that is faster than Faster R-CNN because it does not use the region proposal stage and directly applies CNN on the image directly and classifies the region containing the object. YOLO lacks in detecting small objects or when objects are clustered. These disadvantages were overcome by Redmond et al. in YOLO v2 (2017) [11] which was an improved version of YOLO. It uses Darknet 19 architecture containing 19 convolutional layers, 5 max pooling layers and a SoftMax layer for classification objects. It significantly outperformed YOLO v1 in detecting small objects. Further, Redmond et al. proposed YOLO v3 (2018) [12] which improved the speed and accuracy of YOLO v2 by using more Convolutional layers as it uses a Darknet 53 architecture.

Latest approaches include Shafiee et al.'s (2017) [13] Fast YOLO [13] and Liu et al.'s (2016) [14] SSD (Single Shot Multibox Detector) which are state of the art methods for object detection. SSD is slower than Fast YOLO but if accuracy is important then SSD is the more modest of the two.

Liu et al. (2019) [15] used multi labelled classification to improve object detection which used multi-label classification as an additional task to improve object detection accuracy. They used a real time object detection technique based on YOLO, used pre-processing to remove background and then used Fast YOLO [13] for detection and took inspiration from GoogleLeNet [16] to change the starting layers of YOLO for better speed.

Fig. 1 shows the number of papers hosted on ArXiv related to object detection using image segmentation and bounding boxes as annotations. It shows an increasing trend in the number of papers published per year and there is almost equal popularity of both the methods for object labelling, however the use of bounding boxes for annotation is slightly more.

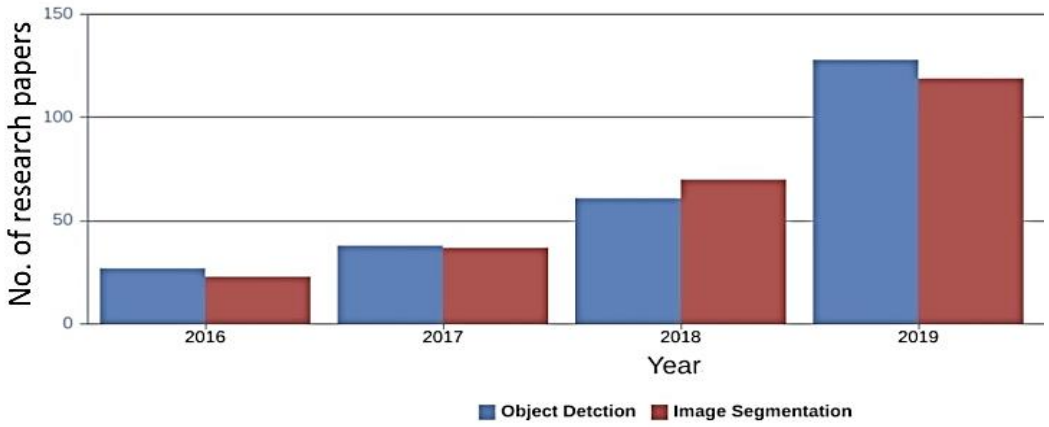

Figure 1. Papers hosted on ArXiv using bounding boxes and segmentation for object detection.

While most of the research is focused on improving the detection algorithms and architecture of the neural networks, image pre-processing is a comparatively less studied area. Machine learning techniques have the potential of un-revealing many hidden facts in different types of data and found giving better performance in comparison with several state of art methods proposed in different area of science and technology [17-23].

The input data is processed using Gaussian Blur and background subtraction techniques and detector models are evaluated on this data to see if detection accuracy improves. Also, as the data recorded with CCTVs have objects at arbitrary angles, bounding box annotation may not be the best choice as a bounding box encloses additional area around the object.

The following three angles of analysis have been discussed throughout the paper:

1. A comparative analysis of object detection accuracy on the real video dataset using different pre-processing techniques such background subtraction and Gaussian Blur. Moreover, two different techniques for background subtraction i.e., Temporal Median Filtering and Canny Edge Detection have been used to evaluate the detection models.

2. Performance evaluation and accuracy comparison of various state of the art object detection models (YOLO and Mask R-CNN) has been done on this new dataset, first one using the boundary coordinates of an object for annotations and another with data segmented at pixel level.

3. Demonstrating the effectiveness of transfer learning by comparing the accuracy of object detection methods on this real video dataset by pre-trained COCO weights and fine-tuned weights.

III. PROPOSED WORK

The real traffic video data has been taken from the CCTV cameras installed outside The NorthCap University, Gurugram (Haryana) entrance gate. These cameras are installed at different angles and positions. The orientation of cameras plays a major role in accurate identification of on road objects. It is difficult to label these objects using bounding box technique. This is because the objects are no longer perpendicular to the frame and are inclined at an angle. As a result, a lot of redundant area is also marked when the object is labelled using bounding boxes. Apart from camera alignment issues, the other concern is about segmentation models. At present, segmentation models are complex and take more time to train. Thus, it became important to the test the accuracy of different labelling techniques on any given video surveillance dataset. This new real traffic video dataset used for object identification consists of five classes namely person, bicycle, bike, car and auto.

The training data involves images from nighttime and from different weather conditions like rainy weather with low lighting conditions and the sunny weather. Fig. 2 shows the sample images taken from this new dataset.

Figure 2. Images used for training from different cameras and various lighting conditions.

The following algorithms/models have been utilized to demonstrate the performance improvement in traffic object detection in combination with some pre-processing techniques:

A. YOLO V3

This algorithm for object detection uses bounding boxes for object localization on the input image and then from these boxes the convolutional features are extracted. This approach takes an image as input and divides it into N x N grid and for each cell the probability distribution of an object class is simultaneously calculated to predict a corresponding confidence score with its bounding box.

The YOLO algorithm computes confidence score and gives the class of the object around the bounding box with the accuracy of the object. If an object is present in the bounding area, then the confidence value will be equal to the intersection over the union of the score of predicted classes. But if there is no object detected in the bounding box, the confidence value will be zero. In order to improve the results, a pre-trained model trained on COCO dataset have been used and fine-tuning of the final layers is done on our own dataset. Fig. 3 shows an overview of the training and prediction process in the YOLO algorithm.

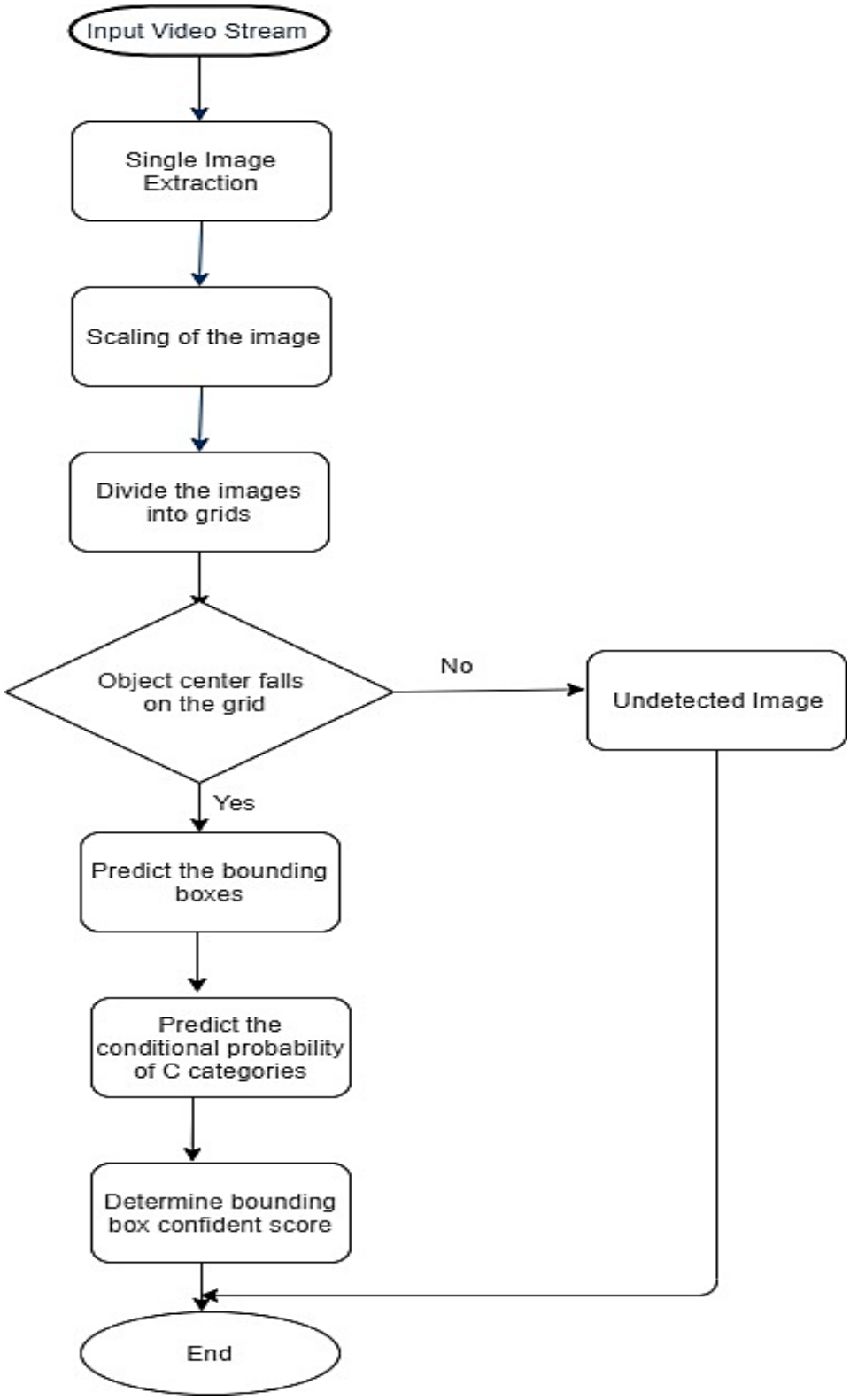

Figure 3. Generic layout of the training and predictions process in YOLO algorithm

B. *YOLO-tiny V3*

The YOLO-tiny algorithm has the same backbone as YOLOv3 but only uses one detection layer for prediction of bounding boxes (YOLO v3 uses three of these detection layers for prediction of bounding boxes). Since it uses only one detection layer its inference speed is more than YOLO v3. But due to this it has a lower accuracy and lower floating-point operations per seconds (FLOPS) needed for inference.

C. *YOLO V4*

YOLO v4 uses CSP Darknet 53 as backbone which is inspired from Cross Stage Partial Networks (CSPNet) [25]. CSPNets propose a way of reducing computation of gradients by 20% with an equivalent or higher accuracy on the MS COCO object detection dataset, by integrating feature maps from the start and end of a network stage. YOLO v4 [26] uses a head, the same as YOLO v3 involving three detection layers. The detection layers comprise CNN layers which compute the location objects at different scales. Table 1 shows the difference between three famous YOLO variants.

D. *MASK R-CNN*

Masked RCNN is unlike other object detection algorithms which uses bounding box strategy, instead it uses image segmentation for object detection. Masked RCNN is an extension of the Faster R-CNN algorithm that uses an output model for predicting the mask of the detected objects. The process of object detection using masked-rcnn is divided into two stages. In the first stage, we get the area in which the objects are present using Region Proposal Network (RPN).

Table 1. Comparison between different YOLO models

Parameters	YOLO v4	YOLO v3	YOLO-Tiny
Architectural Differences	It uses CSPDarknet53 to extract features	Uses darknet53 to extract features	Uses darknet53 to extract features but uses only one YOLO block.
Accuracy On imagenet dataset @ mAP 50	65.7%	65.4%	33.1%
Computation/forward pass FPS	128.5 BFlops 34 FPS	100.5 BFlops 32 FPS	5.6 BFlops 345 FPS

In the next stage, the objects are classified, and bounding boxes and segmentation masks are created around the objects. The different layers used in the model detect different features of the objects such as edges, etc. and handle the complex features such as person, car, etc. In the last section, a mask is generated covering all the pixels that the object occupies. Fig. 4 shows an overview of the training and prediction process in Mask R-CNN.

The following pre-processing techniques have been used to show the analysis perspective:

E. *Gaussian Blur*

Gaussian Blur performs smoothing on an image by applying a Gaussian function. It is a widely used technique in graphics software, to reduce distortion or noise from the images. In one dimension, the Gaussian function is:

$$G(x) = \frac{1}{\sqrt{2\pi\sigma}} e^{-\frac{x^2}{2\sigma^2}} \qquad (1)$$

Where G is the Gaussian function, σ is the standard deviation of the distribution and x represents the 1-D *x* component kernel.

F. Canny Edge Detection

Canny edge detection is a multi-stage algorithm that uses a series of steps to detect or sketch a wide range of edges in the images. These different stages of the algorithm comprise noise removal using Gaussian filter, gradient calculation, Non-maximum suppression, and edge tracking using Hysteresis.

The noise or distortion is removed from the input image using a 5 x 5 Gaussian filter. It reduces the amount of noise present in the image, and we get the output as a smoothened image. In the next step, it calculates the derivative of Gaussian to compute the intensity of the gradients. The equations used to calculate edge gradient (G) and (θ) the direction for each pixel are as follows:

$$Edge\ Gradient\ (G) = \sqrt{G_x^2 + G_y^2} \qquad (2)$$

$$Angle\ (\theta) = tan^{-1}\left(\frac{G_y}{G_x}\right) \qquad (3)$$

where, Angle (θ): Direction of each pixel,

Edge Gradient (G): Magnitude of the edge gradient,

G_x: Edge gradient along x-axis, and

G_y: Edge gradient along y-axis.

After computing the gradient magnitude and direction, an effort is made to remove the unwanted pixels from the image in which the edge is not present. In this step, each pixel is scanned to check if it is a local maximum in its neighbourhood in the direction of the gradient. In this last step, edge pixels are kept or discarded using hysteresis thresholding on the gradient magnitude [28-29].

G. Temporal Median Filtering

For estimating the background of an image and efficiently removing it from the frame, we use the Median Filtering frame differencing method [30]. We take a number of images and estimate the background by taking the median of these images. So, the background is estimated from the median of pixel values in the input image stack.

The estimated background is used as the base image and subtracted from all the input image frames. To the resulting frame, we apply a certain threshold to keep only the regions of interest, i.e., the foreground.

$$B(x, y, t) = median\{I(x, y, t - i)\} \qquad (4)$$

$$|I(x, y, t) - B(x, y, t)| > Th \qquad (5)$$

where i ∈ {0, . . ., n − 1} and I is the image and B is the background at time t.

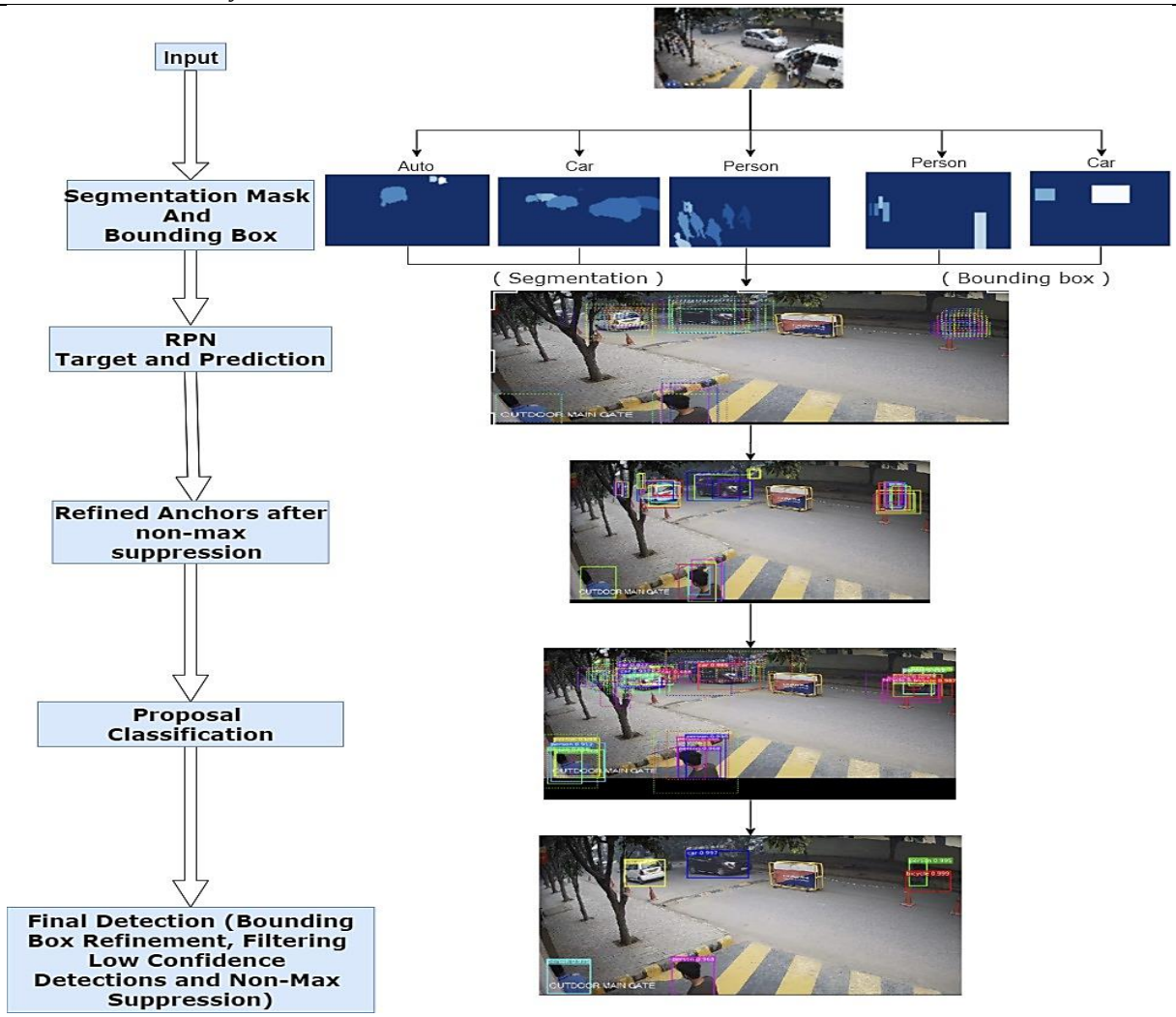

Figure 4. Overview of the training and predictions process in Mask R-CNN algorithm

Assuming the median background to be a frame a time t, the difference of the input/current frame and the background gives an image shown in Fig. 5 (a, b). Only after applying a threshold value, we can filter the required foreground from the pixel difference image. As shown in Fig. 5, an input image is subtracted from the estimated median background, resulting in the image similar to Fig. 5 (c). Applying the threshold value results in image Fig. 5 (d) with only the regions of interest i.e., the foreground.

Figure 5 (a, b. c and d). The process of background subtraction in an image using frame differencing.

The following strategy have been employed to improve the object detection accuracy. The step-by-step description is as follows:

- The first step is to extract the frames from input video taken from CCTV footage. A software named KVMS, provided by the CCTV manufacturer is used to extract images from the video stream at 25fps.

- In the next step, these extracted images are pre-processed. The images extracted in the last step are first loaded, as the images are of different sizes, so we established a base size i.e., width and height to 220 pixels for all the images before further pre-processing.

- For pre-processing the dataset, smoothing is performed on these resized images to remove unwanted noise using Gaussian Blur. For background subtraction, we have used two techniques namely Canny Edge Detection and Frame Differencing. With Canny Edge Detection, we just get the outline of the objects in the frame as shown in Fig. 7, whereas the Frame Differencing method outputs the complete mask of the foreground objects as shown in Fig. 6.

- The idea behind using these two techniques is to compare how the detectors perform with this transformation of the data wherein in one case just the outline of the objects is present and, in another case, the complete mask is given (Fig. 8). But in both of these cases, the background has been removed.

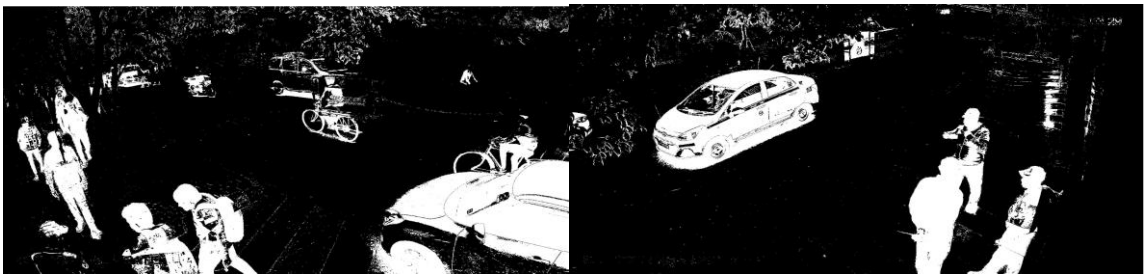

Figure 6. Images after background subtraction using frame differencing.

Figure 7. Images after background subtraction using Canny Edge detection

Figure 8. a. Image using Gaussian Blur b.) Mask generated by Gaussian Blur

The output image after pre-processing is fed as input to the annotation step for further processing. Annotation/Labelling was done using two techniques, instance segmentation and bounding boxes. VGG Image Annotator (VIA) Tool [26] is used for segmentation and LabelImg tool [27] for drawing bounding boxes. Fig 9 shows the annotation done using image segmentation and bounding box using proposed approach.

Figure 9. a. Annotation using image segmentation b.) Annotation using bounding box

As we compare the accuracy for both YOLO and Mask-RCNN algorithms, we save some effort by using the same annotations for Mask-RCNN we use for the YOLO algorithm. YOLO takes in as input the width, height and x-y coordinates of the center point, for an object in an image, and these values are relative to the height and width of the image. Mask RCNN on the other hand takes in input a set of x-y coordinates and generates a polygon mask around the object. We take the properties of an object in YOLO annotations format and convert it to the x-y coordinates for a polygon, using the equations below. This eliminates the need to separate labels for both the algorithms.

Annotations for YOLO algorithm is in the following format: (x_center, y_center, width, height)
All these values are relative to the width and height of the image. To convert these values into absolute terms, we multiply these relative values by the height or width of the image and round them to the nearest integer:

$$absolute_{value} = relative_{value} * image_{width}$$

So, for all these values, we transform,

$$abs_x = relative_{x_{center}} * image_{width}$$
$$abs_y = relative_{y_{center}} * image_{height}$$
$$abs_{width} = relative_{width} * image_{width}$$
$$abs_{height} = relative_{height} * image_{height}$$

We now have the object bounding box values in absolute terms. We shift the centre coordinates to point to one the vertices by subtracting the one half of image height and width from y and x coordinates respectively:

$$x = abs_x - \frac{width}{2}$$
$$y = abs_y - \frac{height}{2}$$

The x and y coordinates point to a vertex of the object's bounding box. Now we map all the remaining vertices.

$$all_{x_{coordinates}} = (x, x + image_{width}, x + image_{width}, x)$$

$$all_{y_{coordinates}} = (y, y + image_{height}, y + image_{height}, y)$$

Finally, a polygon is generated from these coordinates, which is then used to generate a mask which is fed into the Mask R-CNN algorithm.

General layout of the proposed strategy is as follow:

The dataset is trained on the YOLO and Mask-RCNN algorithms. For the YOLO algorithm, the input data is the one with bounding box annotations and for Mask RCNN, we train the model on data with both segmentation and bounding box annotations separately. Fig. 10 shows the overview of the proposed strategy.

Once the data is labelled, the dataset is split into a training, validation and test set. The models are trained on the training set and are further validated with the help of validation set to ensure that the detector model is not overfitting. On successful completion of training and validating the model, it was tested on the test dataset to find the accuracy of the model. The detector models are trained separately for each scenario, i.e., with the original data and the pre-processed data.

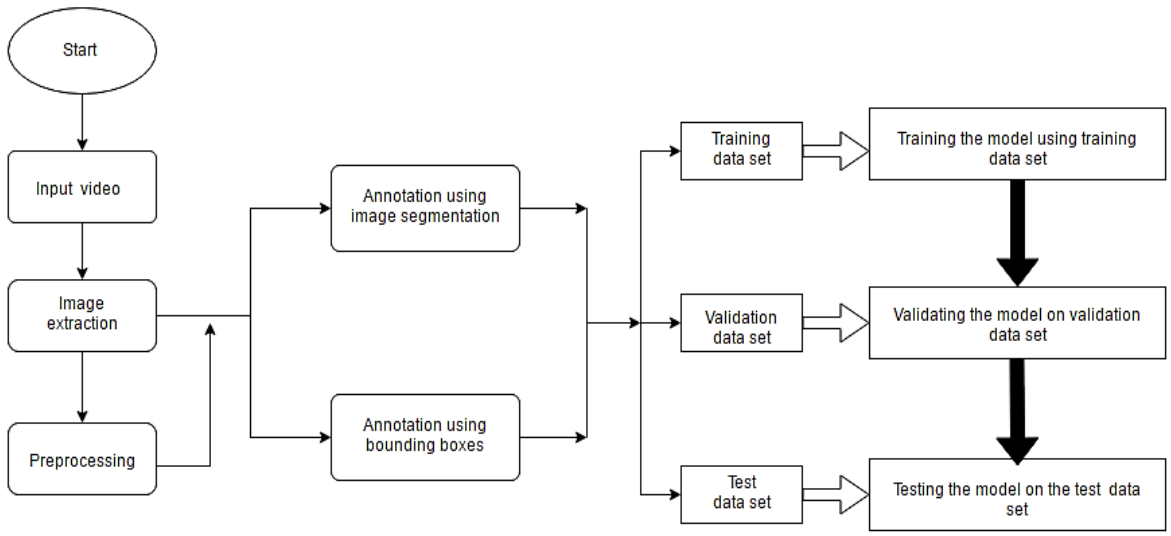

Figure 10. General layout for training and testing process in proposed method

IV. RESULT ANALYSIS

For processing the raw dataset, we use a machine with 16GB RAM, Nvidia 940MX Video Card with 4GB memory and Intel i7 processor. And for training the detectors on the dataset, we use Google Colaboratory. Table 2 shows the mAP values for detection algorithms.

Table 2. mAP values for detection algorithms

No. of Iterations	YOLO-v3	YOLO-v4	YOLO-Tiny	Mask R-CNN	Mask R-CNN
	Bounding Box	Bounding Box	Bounding Box	Segmentation	Segmentation
1000	55.94	72.95	46.16	63.18	36.06
2000	66.52	75.58	59.96	77.47	10.21
3000	69.30	75.08	49.25	74.00	24.35
4000	79.38	70.33	68.00	72.16	20.66
5000	77.02	78.00	59.90	75.34	17.01
6000	76.33	76.06	57.22	85.18	30.42
7000	77.68	76.31	57.15	75.20	41.11
8000	78.38	74.01	61.92	66.18	54.48
9000	70.26	63.54	63.43	75.47	42.18
10000	72.40	69.18	60.02	77.67	45.80

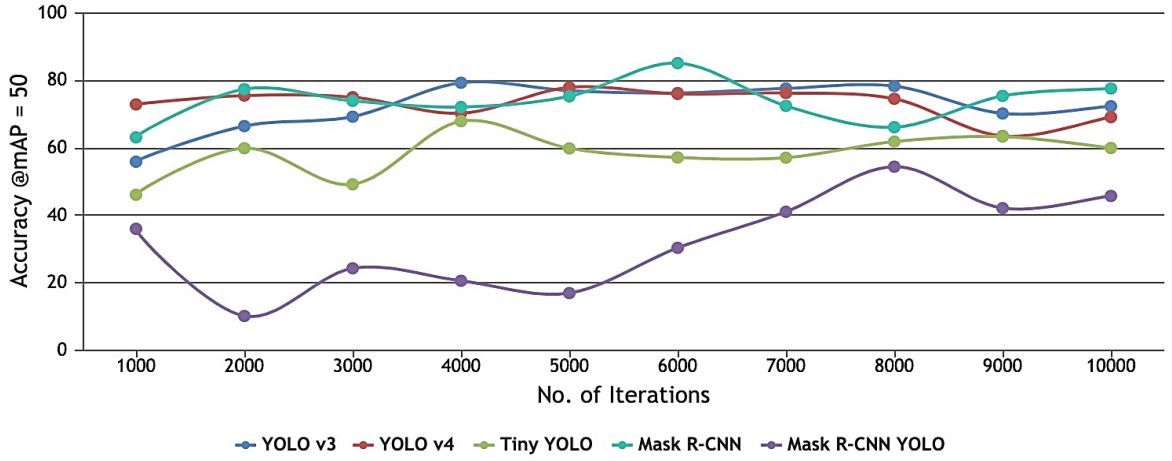

Figure 11. Comparison of mAP values for detection algorithms

The above results have been computed on the original dataset. From Table 2 and Fig. 11. we can see that,

- Mask R-CNN outperforms all other algorithms by 5% at mAP = 50
- YOLO V3 and V4 show similar results on our dataset, with V3 performing better by a small margin of 1.3%
- Mask R-CNN on YOLO annotations, performs poorly on the dataset with max accuracy of 54%. When trained on data with bounding box annotations, Mask R-CNN seems to struggle with localisation of the object mask and thus loses accuracy. So, it might not be the best case to train this model on bounding box data.
- Although Mask R-CNN is better in terms of accuracy on our dataset, training time for Mask-RCNN is much more than algorithms like YOLO. So, if slight accuracy gains and pixel level detections are not a priority, algorithms like YOLO are still a better choice than segmentation models.

Table 3 shows the mAP values for YOLO-Tiny and Mask R-CNN on different pre-processing methods.

Table 3. mAP values for YOLO-Tiny and Mask RCNN on different pre-processing methods

No. of Iterations	YOLO-Tiny Raw	YOLO-Tiny Canny	YOLO-Tiny Median Filtering	YOLO-Tiny Gaussian Blur	Mask R-CNN Raw	Mask R-CNN Canny	Mask R-CNN Median Filtering	Mask R-CNN Gaussian Blur
1000	46.16	20.71	33.07	35.00	63.18	11.9	38.59	75.23
2000	59.96	48.27	39.88	58.10	77.47	13.94	66.38	79.83
3000	49.25	31.38	44.78	61.16	74.00	12.21	45.74	68.59
4000	68.00	43.52	36.27	56.72	72.16	16.00	57.40	83.44
5000	59.90	45.45	49.81	58.68	75.34	13.20	50.05	76.53
6000	57.22	40.83	35.16	59.12	85.18	9.02	71.90	77.38
7000	57.15	36.67	36.0	58.14	75.20	9.46	71.06	67.74
8000	61.92	37.00	37.46	59.96	66.18	11.23	62.98	81.27
9000	63.43	46.58	60.21	58.27	75.47	15.90	65.18	76.37
10000	60.02	44.91	42.27	59.67	77.67	16.40	56.22	70.51

In case of YOLO-tiny and Mask RCNN, we train the dataset on both original data and pre-processed data. Fig. 12 and Fig. 13 shows the observed results.

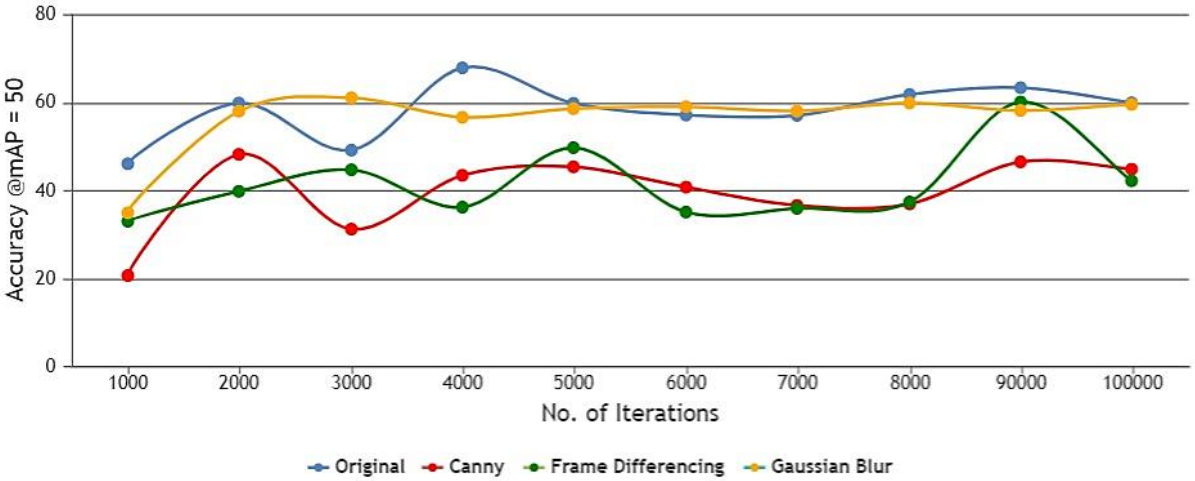

Figure 12. Accuracy in case of YOLO-Tiny algorithm with differently processed datasets

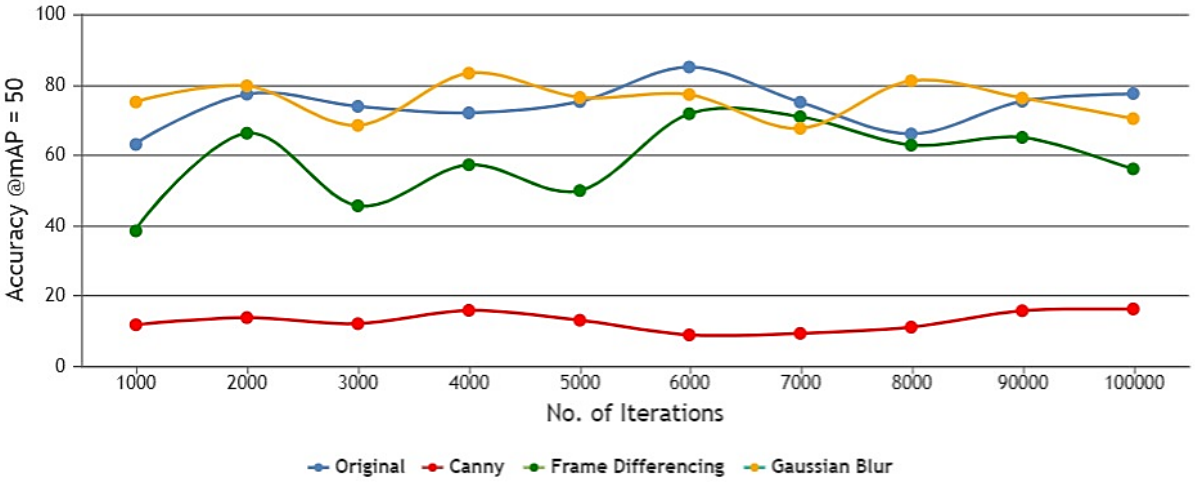

Figure 13. Accuracy in case of Mask R-CNN algorithm with differently processed datasets

- The detection algorithms perform better on original dataset and perform similarly well on data pre-processed using Gaussian Blur technique
- Pre-processing techniques like Canny Edge Detection and Frame Differencing do not improve the detection accuracy whereas data processed using Gaussian Blur performs much better as compared to these.
- Pre-processing may lead to a performance decrease in some cases, as there is significant loss in accuracy for the Mask RCNN model and a relatively small accuracy loss for YOLO algorithm when the pre-processed data is fed.

Table 4 demonstrate the performance difference between pre-trained and fine-tuned weights.

Table 4. Comparison between pre-trained and fine-tuned weights

Annotation	Network	Fine-Tuned Weighs With Preprocessing	Fine-Tuned Weighs Without Preprocessing	Pre-trained Accuracy (mAP)
Bounding Box	YOLO-Tiny	68.00%	61.16%	0.13%
Segmentation	Mask R-CNN	65.60%	85.18%	0.32%

From Table 4, it can be observed that transfer learning not only increases the accuracy but also reduces training time. It is quite interesting to note that weights trained on benchmark datasets are not able to detect objects in our dataset. But fine-tuning these weights by training on our dataset, results in a good accuracy score. Fig. 14 presents the object detection scenario generated using proposed strategy.

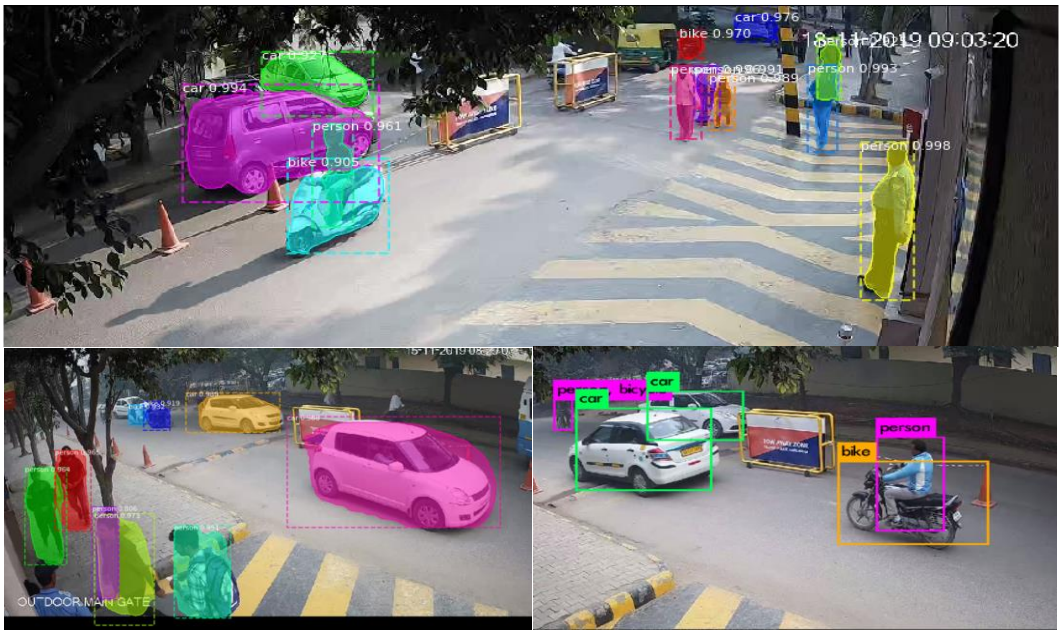

Figure 14. Prediction results for (a) Mask RCNN with pixel level segmentation, (b) Mask RCNN with bounding box annotations and (c) YOLO-Tiny with bounding boxes

The detailed result analysis presented in this section proves that pre-processing techniques when used correctly have the potential of providing better object detection from on road traffic video.

V. CONCLUSION AND FUTURE WORK

While the YOLO algorithm performs good in most of the scenarios related to the detection of on road traffic objects, there can be certain cases that might benefit from the added complexity of segmentation models like Mask R-CNN. These segmentation models can be used for precision-based traffic control systems, where the critical matter is not only to detect and classify an object but also to know the exact position and the region in which any object is lying.

In the proposed analysis, results are better in raw images as compared to pre-processing scenarios. It can be observed that out of certain pre-processing techniques, techniques which induce colour variations to the dataset perform poorly than others. This shows that the algorithms which are color and background sensitive have image features playing a significant role in detection accuracy as training on images with dark or no background shows poor results in comparison to the ones having both color and background. As deep learning models require more data to capture variation precisely in the dataset, therefore single band images lack the complexity otherwise present in RGB images.

Although the pretrained weights on COCO dataset are trained on a huge number of images, this paper shows how they can fail to detect similar objects in an entirely different and complex setting. But transfer learning makes it possible to train algorithms on new dataset while using the pretrained weights as their starting point resulting in improved accuracy.

While segmentation models do a better job at this than other algorithms like YOLO, there is still a scope of improvement. Due to its complex nature, Mask R-CNN takes more time to train than other detection models like YOLO, so improvements in model architecture can reduce complexity thereby helping decrease the training time.

One significant advantage that background subtraction provides is that it can help train a more robust classifier. We plan to keep this a part of future study to train the background subtraction weights as the base weights to train better detector algorithms.

Conflict of interest: The authors declare that they have no conflict of interest.

Ethical statement: The authors declare that they have followed ethical responsibilities.

REFERENCES

[1] Luvizon, D. C., Nassu, B. T., & Minetto, R. (2016). A video-based system for vehicle speed measurement in urban roadways. *IEEE Transactions on Intelligent Transportation Systems*, *18*(6), 1393-1404.

[2] Ćorović, A., Ilić, V., Durić, S., Marijan, M., & Pavković, B. (2018, November). The Real-Time Detection of Traffic Participants Using YOLO Algorithm. In *2018 26th Telecommunications Forum (TELFOR)* (pp. 1-4). IEEE.

[3] Brahme, Y. B., & Kulkarni, P. S. (2011, October). An implementation of moving object detection, tracking and counting objects for traffic surveillance system. In *2011 International Conference on Computational Intelligence and Communication Networks* (pp. 143-148). IEEE.

[4] Ramík, D. M., Sabourin, C., Moreno, R., & Madani, K. (2014). A machine learning based intelligent vision system for autonomous object detection and recognition. *Applied intelligence*, *40*(2), 358-375.

[5] Zhao, W. L., & Ngo, C. W. (2012). Flip-invariant SIFT for copy and object detection. *IEEE Transactions on Image Processing*, *22*(3), 980-991.

[6] Dalal, N., & Triggs, B. (2005, June). Histograms of oriented gradients for human detection.

[7] Girshick, R., Donahue, J., Darrell, T., & Malik, J. (2014). Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 580-587).

[8] Lou, X., & Cui, B. (2007). Boundedness and exponential stability for nonautonomous RCNNs with distributed delays. *Computers & Mathematics with Applications*, *54*(4), 589-598.

[9] Ren, S., He, K., Girshick, R., & Sun, J. (2015). Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems* (pp. 91-99).

[10] Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 779-788).

[11] Redmon, J., & Farhadi, A. (2017). YOLO9000: better, faster, stronger. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 7263-7271).

[12] Redmon, J., & Farhadi, A. (2018). Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767*.

[13] Shafiee, M. J., Chywl, B., Li, F., & Wong, A. (2017). Fast YOLO: A fast you only look once system for real-time embedded object detection in video. *arXiv preprint arXiv:1709.05943*.

[14] Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C. Y., & Berg, A. C. (2016, October). Ssd: Single shot multibox detector. In *European conference on computer vision* (pp. 21-37). Springer, Cham.

[15] Gong, T., Liu, B., Chu, Q., & Yu, N. (2019). Using Multi-label Classification to Improve Object Detection. *Neurocomputing*.

[16] Luo, Z., Branchaud-Charron, F., Lemaire, C., Konrad, J., Li, S., Mishra, A., ... & Jodoin, P. M. (2018). MIO-TCD: A new benchmark dataset for vehicle classification and localization. *IEEE Transactions on Image Processing*, *27*(10), 5129-5141.

[17] Kaur, H., Sharma, H., & Manu, G. (2014). Multi-hop Routing SEP (MR-SEP) for clustering in wireless sensor network. *International Journal of Engineering Technology, Management and Applied Sciences*, *2*(3), 54-65.

[18] Sharma, S., Bansal, R. K., & Bansal, S. (2017). Heterogeneity-aware Energy-efficient Clustering (HEC) Technique for WSNs. *TIIS*, *11*(4), 1866-1888.

[19] Sharma, S., Bansal, R. K., & Bansal, S. (2016). Energy efficient clustering techniques for heterogeneous wireless sensor networks: A survey. *J. Mobile Comput. Commun. Mobile Netw.*, *3*, 1-9.

[20] Ahuja, R., Chug, A., Kohli, S., Gupta, S., & Ahuja, P. (2019). The impact of features extraction on the sentiment analysis. *Procedia Computer Science*, *152*, 341-348.

[21] Ahuja, R., & Banga, A. (2019). Mental stress detection in university students using machine learning algorithms. *Procedia Computer Science*, *152*, 349-353.

[22] Jain, D., & Singh, V. (2018). An efficient hybrid feature selection model for dimensionality reduction. *Procedia Computer Science*, *132*, 333-341.

[23] Jain, D., & Singh, V. (2019). A two-phase hybrid approach using feature selection and adaptive SVM for chronic disease classification. *International Journal of Computers and Applications*, 1-13.

[24] Jain, D., & Singh, V. (2018). Feature selection and classification systems for chronic disease prediction: A review. *Egyptian Informatics Journal*, *19*(3), 179-189.

[25] Abhishek Dutta and Andrew Zisserman. 2019. The VIA Annotation Software for Images, Audio and Video. In Proceedings of the 27th ACM International Conference on Multimedia (MM '19), October 21–25, 2019, Nice, France. ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3343031.3350535.

[26] Tzutalin. LabelImg. Git code (2015). https://github.com/tzutalin/labelImg

[27] Wang, C. Y., Liao, H. Y. M., Yeh, I. H., Wu, Y. H., Chen, P. Y., & Hsieh, J. W. (2019). CSPNet: A New Backbone that can Enhance Learning Capability of CNN. arXiv preprint arXiv:1911.11929.

[28] Bochkovskiy, A., Wang, C. Y., & Liao, H. Y. M. (2020). YOLOv4: Optimal Speed and Accuracy of Object Detection. arXiv preprint arXiv:2004.10934.

[29] Canny, J. (1986). A computational approach to edge detection. *IEEE Transactions on pattern analysis and machine intelligence*, (6), 679-698.

[30] Tamersoy, B. (September 29, 2009). "Background Subtraction – Lecture Notes" (PDF). University of Texas at Austin

