2022 10th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO)
Amity University, Noida, India. Oct 13-14, 2022

2022 10th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO) | 978-1-6654-7433-7/22/$31.00 ©2022 IEEE | DOI: 10.1109/ICRITO56286.2022.9965169

A Novel approach for Detection and Counting of Vehicles using CNN architectures

S.Venkatesh,
Department of CSE
Gokaraju Rangaraju Institute Of Engineering and Technology
Hyderabad,Telangana, India
email:sirusanivenkatesh143@gmail.com

Dr. B.Sankara Babu
Professor
Department of CSE
Gokaraju Rangaraju Institute of Engineering and Technology
Hyderbad,Telangana,India
email:bsankarababu81@gmail.com

***Abstract*— Recent years the usage of automobiles significantly increased in urban regions. As the usage increased, the traffic jams also became a major issue in day-to-day life. Even governments tried to increase the infrastructure and failed mitigating the traffic issues. In this research, we are proposing a Deep learning approach to find out the density of the vehicle flow in a fully traffic regions. In order to do this project, we are proposing VGG16 a CNN architecture for vehicle recognition (classification) and proposing Yolo for detection of vehicles. Further we employed the OpenCV techniques for 1) Adaptive background subtraction 2) image segmentation 3) vehicle counting 4) vehicle tracking purpose. The primary objectives of this proposed system is: 1) the ability to count vehicles 2) its efficiency, as it produces accurate results. For this we are using a Traffic and Congestions (TRANCOS) dataset which consists of 1244 images of different vehicles, for training and validation purpose and also we gathered an open source traffic surveillance video for live tracking of vehicles and counting. The proposed system achieved 95.77% model accuracy and 98% of ROC curve for both bike and car classes. Further we validated our model using Accuracy, precision , Recall , F1 metrics and achieved 96% accuracy.**

Keywords—VGG16, Yolo, Convolutional Neural networks (CNN), OpenCV, Vehicle Detection, Vehicle Tracking, Vehicle Counting.

I. INTRODUCTION

A key task of intelligent transportation systems is vehicle counting in traffic video sequences. For traffic management and control, having the exact number of vehicles on the road is crucial. According to the number of vehicles on the road, a general idea of the traffic conditions can be obtained. This includes the intensity of road traffic, lane occupancy, and congestion levels. Automated route planning can rely on this information for early incident detection, traffic congestion prevention, and road congestion prevention. The early intelligent transportation systems used special sensors to determine whether there were vehicles. Magnetic loops, microwaves, and ultrasound detectors were used to identify vehicles. In addition, these methods have a high total cost, a small detection range, and a limited level of accuracy. In recent years, many methods for counting vehicles using vision have been developed thanks to the rapid development of digital video processing [1]. Vision-based vehicle counting systems are more flexible, can detect a wider range of vehicles, are simple to install and maintain, and are cheaper than other traditional methods. There are three types of methods for counting vehicles video-based: frame difference methods, optical flow methods, and background subtraction methods. A-frame difference is compared between successive frames for comparing moving objects with their backgrounds. Basically, these methods detect just parts of moving objects, but they're fast and simple. It is often difficult to satisfy the real-time requirements of optical flow methods because these methods require the calculation of optical information of the whole image. The background subtraction method consists of building a background model and subtracting from each frame the background image that this model represents to obtain the foreground objects. It is quite common practice for vehicle counting to using background subtraction, due to its effectiveness. In most of these methods, however, an area or a large portion of the frame is taken into account when creating the background model. Furthermore, it usually tracks the vehicle after detecting the vehicle. Due to the complexity of these methods, they might not fit for use in low-end computers.

Object detection and classification, as well as vehicle detection and counting, have been greatly impacted by deep learning in the last few years [2]. While traditional methods can't change the appearance of vehicles or decrease the problem of occlusion, deep learning can solve those issues. However, speed and accuracy still struggle to be balanced in deep learning object detection. As for the fact that it remains a difficult task to achieve real-time analysis on a limited computing platform.

In this paper, a new approach for vehicle counting based on convolutional neural networks (CNN) is proposed. The proposed method can achieve real-time processing on a limited computing platform. this paper tackles the task of online movement-specific vehicle counting. Specifically, online movement-specific vehicle counting requires to not only count the total vehicle number for each MOI, but also record the timestamp when each vehicle moves out of the ROI and stream out the counting result in a limited time. Our proposed approach mainly follows the detection pipeline, in which we choose YOLO [3] as the baseline methods for vehicle detection and multi-object tracking, respectively. We proposed Euclidean distance tracker algorithm to track the vehicles movement.

Further, the paper is presented as follows: Section II describes the brief about the previous experimentations proposed by different authors. Section III shows the proposed system. Section IV describes the methodology followed. Section -V describes the results and analysis of the

Authorized licensed use limited to: G Narayanamma Institute of Technology & Science. Downloaded on May 01,2023 at 09:45:05 UTC from IEEE Xplore. Restrictions apply.

proposed model. Section VI describes the conclusion and Section VII shows the citations of reference articles. c

II. Literature Review

In this section, we are providing a brief about the various methods proposed by different researchers.

Boris A. Alpatov et al. proposed a system for analyzing traffic management tasks using self-developed vehicle detection and counting algorithm which requires the definition of special regions of interest (the sensors) in the image. The counting of the vehicle is carried out using Incremental approach of sensor detecting entry and exit of each lane. Further, they used cameras data and detected vehicles using OpenCV technique. Also, they used sensors installed on each road lane for more accuracy. Where each sensor is divided into two zones (usually the entry zone and exit zone) to determine the movement direction of a passing vehicle. Authors used integral vector Radon transform (IVRT) for detecting straight lanes on the road to find out how many vehicles passed on each road. In this experimentation they conducted 5 test sequences and achieved 99.69% accuracy with 0.61% of false alarm rate which is very negligible for the count of 657 vehicles [4].

Ravula Arun Kumar et al. Proposed a method that uses blob tracking and adaptive background subtraction to count the number of vehicles that pass through certain roads and highways [5]. Adaptive background subtraction, image segmentation, vehicle counting, and vehicle tracks make up the four stages of the proposed system. The first step is to apply a background subtraction technique by using the virtual detector and secondly, use techniques for foreground masking, to find contours, to analyze motion, and to detect the edge. For calculating the area and detecting the vehicle, a Kalman filter is used along with the region of interest. A number of open computer vision techniques are used to remove objects and remove noise from video frames through threshold calculation and blob analysis, hole filling, and morphological operations. Three, identifying the vehicles from the video sequence captured from static cameras using contour methods and counting the number of vehicles. As mentioned in the methods above, the vehicle would begin counting as soon as it reached the region of interest. Based on experimental results, the proposed method is approximately 97.25 percent accurate [5].

Ghoreyshi et al., proposed two different network mechanisms to detect vehicles. there are many similarities between the images of the vehicles used for training, testing, and design. In order to create the first network, the ResNet network is combined with Single Shot Detection (SSD). The method of feature extraction is based on ResNet, while localization is done with SSD. The training and testing of this proposed system is carried out on, 150,000 images of 115 vehicles and achieved 95% accuracy [6].

Reza Safabakhsh et al., proposed a method to detect vehicles. classification and counting vehicles authors extracted 2 features: the covariance derived from the greyscale correlation matrix of the vehicle image and length of the vehicle in the associated spatial-time image. Using random forests, vehicles are categorized into three groups based on their size: small (cars), medium (vans), and large (trucks). In order to detect the vehicles in the video frames, the authors used Active basis models (ABMs) and checked the reflection symmetry to detect them. There is no special procedure for counting sequential frames, but once the sequence has been selected, the vehicles are counted and they are then given to the step of classification. In order to train an RF and further categorize vehicles, the vehicle length in the STI image along with the correlation value pulled out of the GLCM matrix is used. To improve algorithm performance and run-time, GPU programming and texture-based features are proposed to enhance the classification part. A dataset comprising seven video streams was used to evaluate the proposed method and the following accuracies were obtained: Noon-90.5%, Afternoon-48.7%, Sundown-68.2%, Night-66.8%, Traffic Jam-63.8% in sunny days, while in rainy days Daytime-51.6% and Night-46.6% detection and counting accuracy was achieved [7].

Nilakorn Seenouvong proposed a video-based algorithm using OpenCV order to find objects in the foreground of a video sequence, a method based on background subtraction. They used to include hole filling, thresholding, adaptive morphology, and others. last step in the process is to count and record the vehicles within a virtual detection zone. Pixels outside of the ROI are deleted when the ROI is detected. Based on the intensity of the pixels in the foreground and background in the different images, the thresholding operation is applied. This proposed method achieved a median error of 0.0686 for this task while these two methods achieved a median error of 0.0957 and 0.2290, respectively

III. Proposed System

The proposed approach is implemented on a Window operating system the configuration of PC is as follows: Intel Core i7 processor, NVIDIA GTX 3060Ti GPU and 16 Gb of RAM. For classification VGG19 architecture is chosen, along with OPENCV library used for video and image processing in real-time and yolo algorithm is used for detection process.

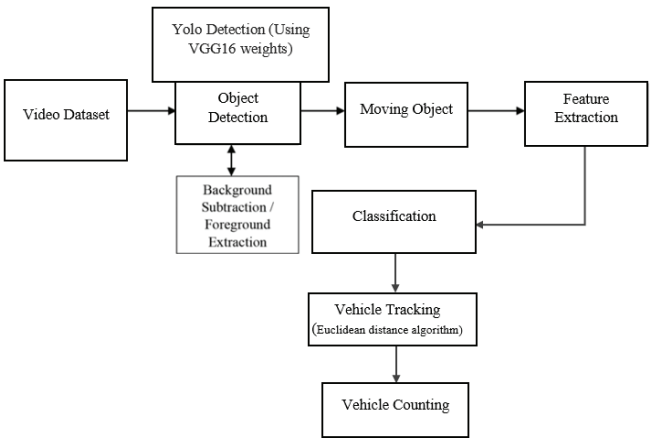

Fig. 1. Proposed Flowchart

1) VGG16 Architecture

K Simonyan and colleagues from the University of Oxford have proposed a convolutional neural network model called VGG16 in the paper "Very Deep Convolutional Networks for Large-Scale Image Recognition" [8]. The model achieved an accuracy of 92.7% in the top-5 test in the ImageNet dataset, which comprises of over 14 million images belonging to 1000 class categories. VGG16 architecture is one of the famous models presented at ILSVRC-2014. Below is an illustration of VGG16. COV1 receives a RGB image of 224 x 224 pixels as input. We have used convolutional layers on the image, in which the filters

Authorized licensed use limited to: G Narayanamma Institute of Technology & Science. Downloaded on May 01,2023 at 09:45:05 UTC from IEEE Xplore. Restrictions apply.

were used with a relatively small receptive field: 3*3 (that's the minimum size to be able to discern up/down, left/right, center). It also uses convolution filters in one configuration, which are linear transformations of the input channels (then non-linear ones). One pixel is fixed as the convolution stride; two pixels is fixed as the spacial padding. After convolution, the layer input is padded by 1 pixel for the 3*3 convolutional layers so that the spatial resolution is maintained. Five layers of maximum-pooling are used for spatial pooling, each following a specific convolutional layer (not all conv. layers are max-pooled) as shown in fig.2. A maximum-pooling calculation is performed over a 2×2 pixel window, with stride 2.

Following a stack of convolutional layers (each with varying depths depending on architecture), three fully connected layers (FC) are added. The first two FC layers each have 4096 channels, while the third FC layer performs ILSVRC classification with 1000 ways (one for each class). Last but not least, the soft-max layer provides the final treatment as shown in fig.2. It is the same for all networks in terms of the way the fully connected layers are configured [9].

In addition, all lattices have been equipped with rectification non-linearity (ReLU). The fact that none of the networks (with the exception of one) contains Local Response Normalization (LRN) leads to increased cell memory consumption and computation time. According to the results for the ILSVRC dataset, LRN does not improve the performance [9].

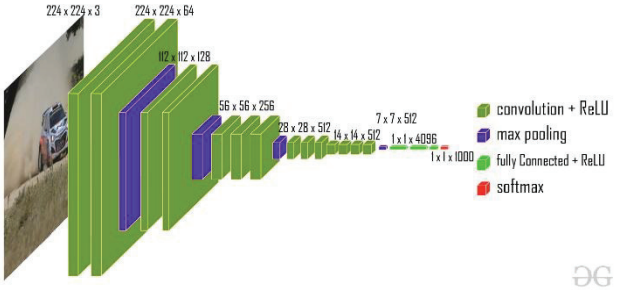

Fig. 2. VGG16 Architecture [10]

B. Vehicle Detection Tracking and Counting:

1) Vehicle Detection

This study proposes a method for detecting the type of vehicle. To directly detect vehicle location and class, the deep object detection network was used in conjunction with YOLO[1]. The option of using the 3-tiny version [1] of the YOLO algorithm was selected due to its lightweight and compact nature and required the highest level of accuracy and speed. Equation 1 summarizes the results from experiments 1 and 2 [11].

$$y_{prediction}^{1,2} = \{[class1: prob1], [class1: prob1], \dots [classn: probn]\} \quad (1)$$

$$y_{prediction}^{3} = \{[prob1, class, x_{min}, y_{min}, x_{max}, y_{max}], [\dots.]\} \quad (2)$$

A convolutional VGG network output is represented by Equation 1, where each class array represents a proposed vehicle class and probe is the probability of accuracy. The output results of the YOLO network are represented in arrays in Equation 2. x_{min}, y_{min}, x_{max}, y_{max} values represent the coordinates of the detected location. Probe values represent the probability that the class will be detected, and class values represent the proposed class. We have largely drawn inspiration from the philosophy of VGG networks for our network's basic and simple principles. A typical convolutional layer has three filters and follows two design rules: (i) The layer has the same number of filters regardless of the output property's size. For each layer, the number of filters is doubled if the size of the feature map is halved. Using stride 2 as a parameter in the convolutional layer, sampling was conducted directly. In addition to that, at the very end of the network we have a soft Max layer proportional to the number of classes that are available. It is been demonstrated that the Yolo [3] network divides a given image into a Grid or S*S network with 416*416*3 dimensions as shown in fig.3. A matrix of 30*S*S is the result of this network. In the input S*S network, one cell is equal to one S*S matrix layer in the output S×S network. The output 30*S*S algorithm contains the cell coordinates and probability calculations per cell. Figure 2 illustrates this in detail. To train the waste function, the 30*S*S output is given with the Ground Truth and this is used as a training mechanism. In order to remove the weak boxes and to show only the right boxes in the total, the 30* S*S output is entered into an algorithm known as Non-Maximum Suppression.

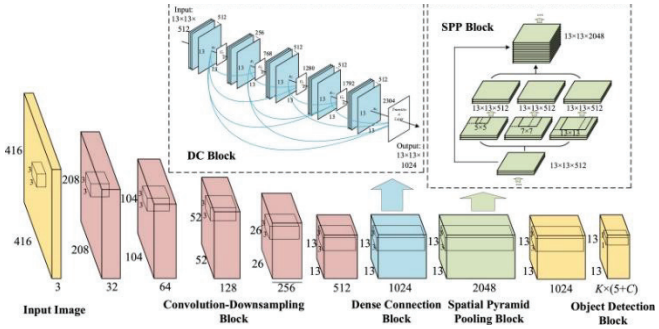

Fig. 3. Yolo architecture [6]

2) Vehicle Tracking

The act of tracking consists of identifying if an object is the same one that was in the previous frame, as well as locking onto the movement of that object. The process involves:

a) Motion detection: Most of the time from a static camera. Commonly used with surveillance systems. This process is often pixel-by-pixel (because of speed constraints).

b) Object localization: Highlights a specific region of an image. Reduces the amount of data. Most of the times there will only be interest points that can be used later to help solve the correspondence problem.

c) Motion segmentation: Each moving object in the image is segmented into its own region.

d) Three-dimensional shape from motion: A well-integrated way of combining motion and form, it is also called structure from motion.

e) Object tracking: Sometimes, only a few features are tracked, for example, the corner points.

Authorized licensed use limited to: G Narayanamma Institute of Technology & Science. Downloaded on May 01,2023 at 09:45:05 UTC from IEEE Xplore. Restrictions apply.

In order to track the vehicles we used the Euclidean distance algorithm between the optical flow estimate of the i th vehicle's center position (It,i) and the observed vehicle position at time t(θt) allows us to determine whether the observed position belongs to a previously existing vehicle, a newly arriving vehicle, or a missed vehicle that has occluded previous frames. The distance between two points is denoted by δ(It,i, θt). Depending on the vehicle's position in the imaging geometry, the resolution varies. As the displacement is higher for closer pixels, those regions have a higher threshold value.

The objective of optical flow calculation is to determine how the windowed image regions centered at locations u and v are similar, i.e., v = u+d in the frame at time t+1, when u = [ux uy] T at time t. A pixel motion vector is called the optical flow vector at u if d = [dx dy] T. The residual function is defined as the minimum of the mean square of d.

$$\varepsilon(d_x, d_y) = \sum_{x=u_x-w_x}^{u_x+w_x} \sum_{y=u_y-w_y}^{u_y+w_y} (I_t(x,y) - I_{t+1}(x+d_x, y+d_y) \quad (3)$$

For optical flow calculations, wx and wy represent the region of interest. Kalman filter updates for matched vehicles' Kalman filters with observed vehicle position for each match with optical flow estimate.

3) Vehicle Tracking:

In order to conduct the use of contour finding methods, as shown in figure it is necessary to calculate the area for each vehicle. If a vehicle is counted, the value 1 will be displayed. As more vehicles are detected, the count will be increased. This value will be set to 0 if no vehicle is detected, and the other frames will continue to be detected. Figure shows that once the vehicle is counted as 1, the area would appear on the output screen. Fig. shows how the area of each vehicle is calculated once the vehicle has crossed the red line and once the vehicle has passed the pink line the vehicle's frame is differentiated to count the vehicles. The vehicle would be counted when it crosses the pink line. In addition to this, the system will work in both bi-directional and one-directional modes, and if no vehicle has been detected, the binary image would be displayed as 0.

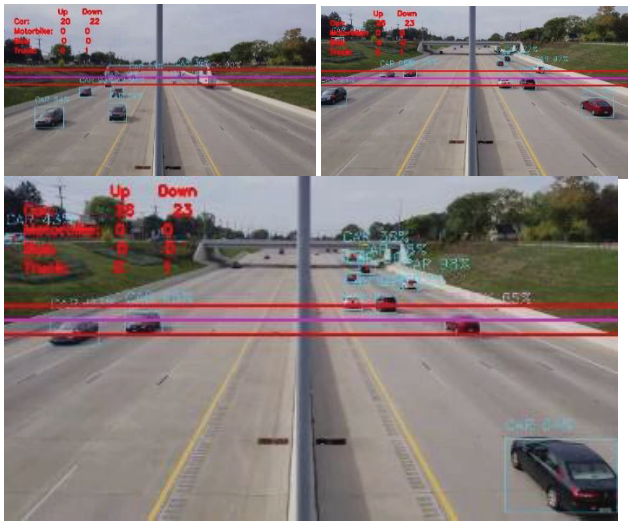

Fig. 4. Vehicle Tracking, Counting and Detection of both sides of the traffic

IV. Implementation and Results

In this section, we are going to explain the implementation and results of the experiment conducted are provided in detail. Initially we trained the VGG16 model using a Mendeley vehicle image dataset [12]. In this process we used a data augmentation technique to multiply the images with a rotation range of 15 and flipped the images. Instead of training the whole model we opted a transfer learning process using ImageNet weights under the proposed model. Later, we added flatten, dense and dropout layers on top of the architecture. In this model we used an adam optimizer to perform the gradient descent algorithm with the learning rate of 0.001. We trained the proposed model using 20 epochs with a batch size of 64 which means for every iteration the architecture takes 64 images as input and train the model and continues the process for next 20 iterations. In order to avoid overfitting, we chose the early stopping function as epochs. Iterative methods such as gradient descent are used when training a learner. Using such methods, the learner is upgraded with each iteration to match the training data better. We achieved 95.76% of validation accuracy and 95.24% of training accuracy at 19th epoch. The accuracy graphs are shown below in the fig 5 and 6.

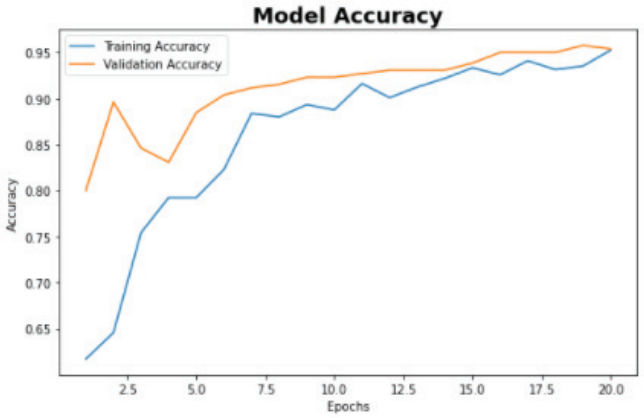

Fig. 5. VGG16 Model accuracy curve

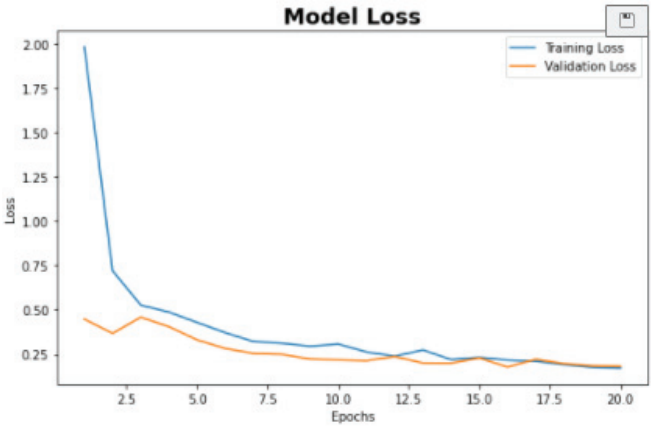

Fig. 6. VGG16 model loss curve

Further, evaluation of the proposed model we used different metrics. There is a model for evaluating the performance of a classification model that is referred to as the Confusion Matrix, that is an N x N matrix where N refers to the target classification list. In this matrix, the actual target values are compared to the predictions of the learning model. The below fig.7 shows the confusion matrix obtained. The

Authorized licensed use limited to: G Narayanamma Institute of Technology & Science. Downloaded on May 01,2023 at 09:45:05 UTC from IEEE Xplore. Restrictions apply.

confusion matrix depicts the misclassified images as purple color which is 3 misclassified images out of 131 validated images and remaining 128 classified correctly represented in yellow color. The predicted misclassified images are shown below in fig.8.

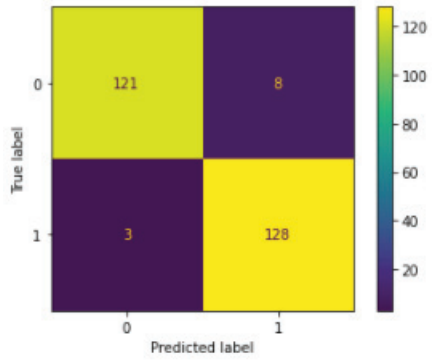

Fig. 7. Confusion matrix

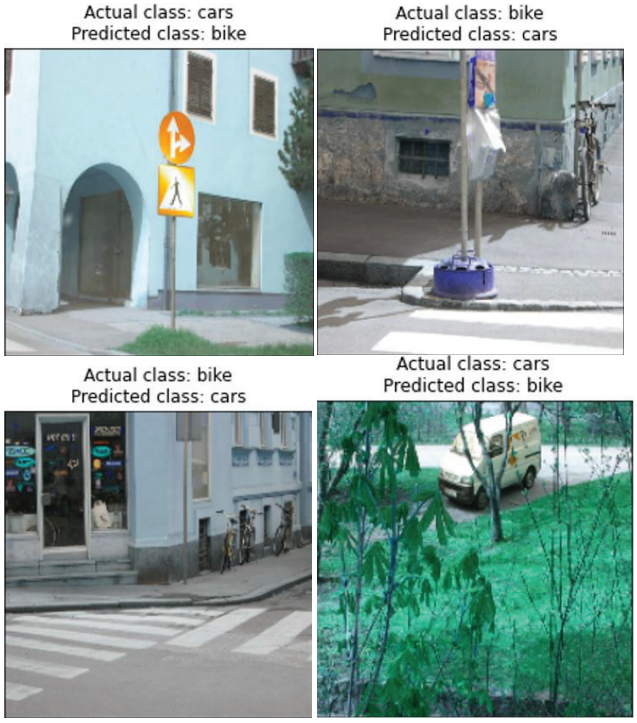

Fig. 8. Misclassified images

In this evaluation we are intended to use the four combinations of TP, FP, TN and FN to plot the ROC curve which can be plotted Precision VS Recall or True positive rate (TPR) vs False positive rate (FPR). The properties precision and recall are actual positives cases that are correctly identified.

$$Precision = \frac{TP}{TP + FP}$$

$$Recall = \frac{TP}{TP + FN}$$

$$TPR = \frac{TP}{TP + FN}$$

$$FPR = \frac{FP}{FP + TN}$$

A ROC curve is produced by plotting a true positive rate along the y-axis and a false positive rate along the x-axis. By examining the precision and recall for three different algorithms with ROC curves, we compared the performance and quantified the results. We obtained a precision and recall metrics of 96%. Also, the ROC curve represents the accuracy of TPR and FPR which is 98% of both bike and car classes. The roc curves are shown in the fig. 9 & 10. Also we measured the Accuracy and F1-Score of the present classification model and obtained 95.77% accuracy and 96% F1 metrics.

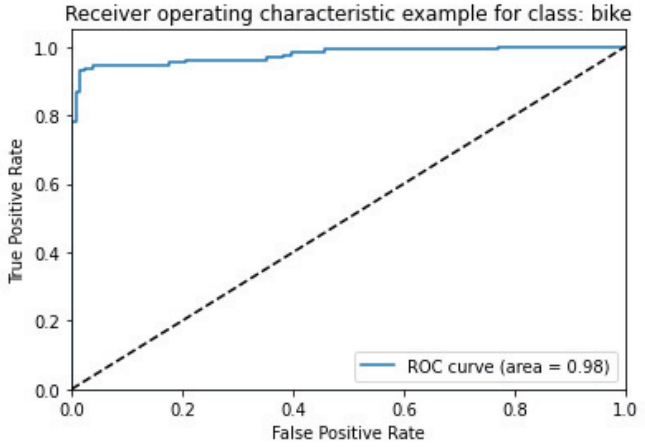

Fig. 9. ROC curve for bike class

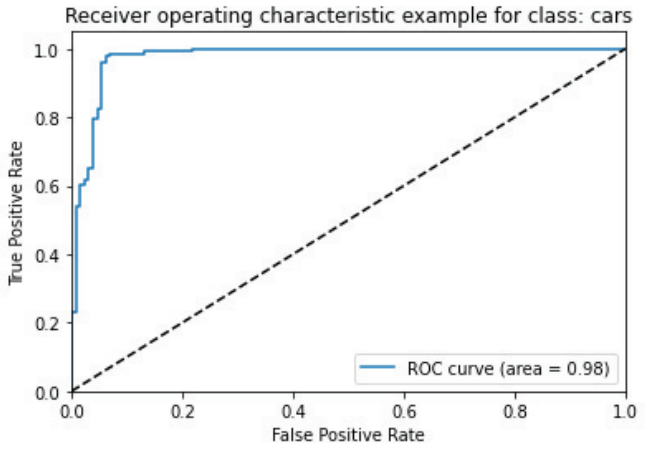

Fig. 10. ROC curve for car class

Further, the saved VGG16 model weights are fed to a Yolo detection architecture to detect the vehicles. In the below fig 11 shows the average accuracy of each vehicle obtained from all the pixels in the frame. Also, we checked the traffic congestion to see how the traffic changes over a period of time represented in the graph below fig. 12

S.no	*Accuracy type*	*Value*
1	Precision	96%
2	Recall	96%
3	TPR	93.79%
4	FPR	97.78%

Authorized licensed use limited to: G Narayanamma Institute of Technology & Science. Downloaded on May 01,2023 at 09:45:05 UTC from IEEE Xplore. Restrictions apply.

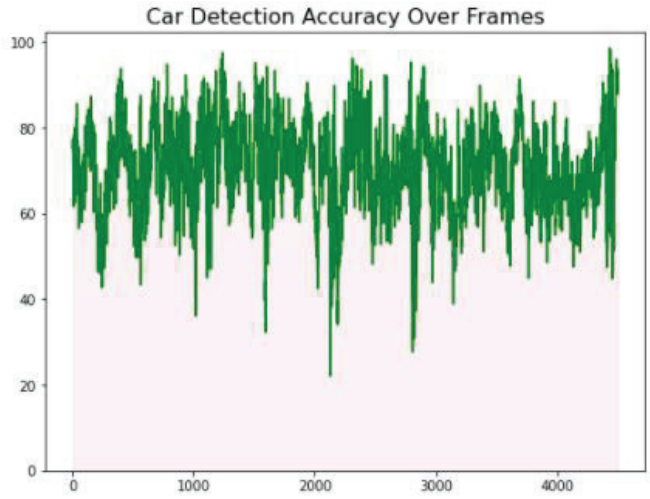

Fig. 11. Vehicle Detection accuracy over frames

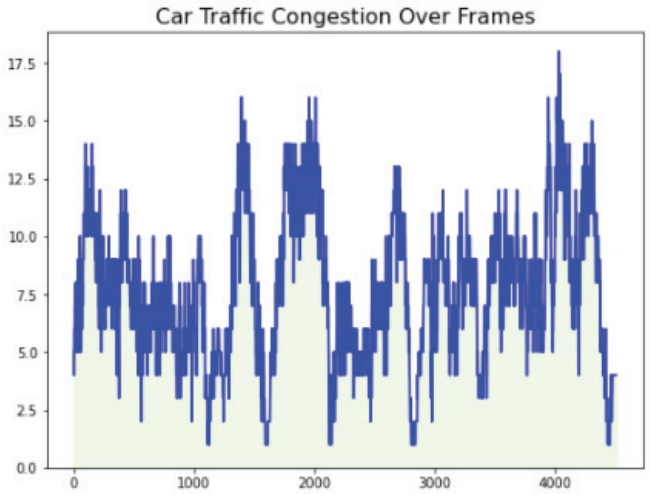

Fig. 12. Traffic congestion over frames

V. Conclusion and Future Scope

Using existing traffic cameras with low frame-rate and low-resolution videos, we present a system for tracking vehicles in real-time using a computer vision and deep learning pipeline. The system we adopted consists of two parts: the detection pipeline, and the tracking pipeline. In order to detect the object, the Yolo object detection algorithm is used. Using a low frame-rate image feed with a low resolution, we demonstrate how our system has a higher accuracy in counting vehicles than existing algorithms. Also, we demonstrate how the system has multiple kinds of adaptability in varied environments. Also, we demonstrated the accuracies of classification model and evaluation metrics to prove the reliability of this model.

References

[1] "NGUYEN, HOANH. "EFFICIENT APPROACH FOR VEHICLE COUNTING BASED ON DEEP CONVOLUTIONAL NEURAL NETWORKS." Journal of Theoretical and Applied Information Technology 97.20 (2019).".

[2] "Hanafi, Muh & Suryana, N. & Basari, Abd Samad. (2018). Deep learning for recommender system based on application domain classification perspective: A review. Journal of Theoretical and Applied Information Technology. 96. 4513-4529.".

[3] "Redmon, Joseph, et al. "You only look once: Unified, real-time object detection." Proceedings of the IEEE conference on computer vision and pattern recognition. 2016.".

[4] "B. A. Alpatov, P. V. Babayan and M. D. Ershov, "Vehicle detection and counting system for real-time traffic surveillance," 2018 7th Mediterranean Conference on Embedded Computing (MECO), 2018, pp. 1-4, doi: 10.1109/MECO.2018.8406017.".

[5] "Ravula Arun kumar, D. Sai Tharun Kumar, K. Kalyan, B. Rohan Ram Reddy "Vehicle Counting and Detection", International Journal of Innovative Technology and Exploring Engineering (IJITEE)".

[6] "A. M. Ghoreyshi, A. AkhavanPour and A. Bossaghzadeh, "Simultaneous Vehicle Detection and Classification Model based on Deep YOLO Networks," 2020 International Conference on Machine Vision and Image Processing (MVIP), 2020, pp. 1-6, doi: 10.1109/MVIP49855.".

[7] "Kamkar, Shiva & Safabakhsh, Reza. (2016). Vehicle detection, counting and classification in various conditions. IET Intelligent Transport Systems. 10. 10.1049/iet-its.2015.0157.".

[8] "Simonyan, Karen, and Andrew Zisserman. "Very deep convolutional networks for large-scale image recognition." arXiv preprint arXiv:1409.1556 (2014).".

[9] "NeuroHive," 20 11 2018. [Online]. Available: https://neurohive.io/en/popular-networks/vgg16/.

[10] pawangfg, "Geeks for Geeks," 27 02 2020. [Online]. Available: https://www.geeksforgeeks.org/vgg-16-cnn-model/.

[11] "Ghoreyshi, Amir Mohammad, Alireza AkhavanPour, and Alireza Bossaghzadeh. "Simultaneous vehicle detection and classification model based on deep YOLO networks." 2020 International Conference on Machine Vision and Image Processing (MVIP). IEEE, 2020.".

[12] "Ali, Mohsin (2022), “Vehicle images dataset for make and model recognition”, Mendeley Data, V1, doi: 10.17632/hj3vvx5946.1".

[13] "Y. Chen, B. Wu, H. Huang and C. Fan, "A Real-Time Vision System for Nighttime Vehicle Detection and Traffic Surveillance," in IEEE Transactions on Industrial Electronics, vol. 58, no. 5, pp. 2030-2044, May 2011, doi: 10.1109/TIE.2010.2055771.".

Authorized licensed use limited to: G Narayanamma Institute of Technology & Science. Downloaded on May 01,2023 at 09:45:05 UTC from IEEE Xplore. Restrictions apply.