www.ijbar.org
ISSN 2249-3352 (P) 2278-0505 (E)
Cosmos Impact Factor-5.86

FASTTRACK CNN FRAMEWORK FOR LIVE VIDEO OBJECT ANALYTICS

Ch. Anil[1*], Kodakandla Suhas[2], Gunti Karthik[2], Meghana Maskula[2], J. Janaki Ram[2]
[1]Assistant Professor, [2]UG Student, [1,2]Department of Computer Science and Engineering
Vaagdevi College of Engineering (UGC - Autonomus), Bollikunta, Warangal, Telangana.
[*]Corresponding author: Ch. Anil (anil_ch@vaagdevi.edu.in)

ABSTRACT

Real-time vehicle counting and classification has become indispensable for modern traffic management, surveillance, and infrastructure optimization. This system combines a pretrained YOLOv4 convolutional neural network and the OpenCV library to detect, classify, and count vehicles in live video streams. Each frame undergoes background subtraction, YOLO inference, and non-maximum suppression to refine detections. The model outputs bounding boxes and labels for multiple vehicle categories, including cars, trucks, buses, and motorcycles. Continuous tracking and aggregation yield accurate count metrics and type-specific traffic statistics, updated at over thirty frames per second on standard GPU hardware. Robust performance under diverse environmental conditions such as low light, occlusion, and traffic ensures reliable operation in urban and highway scenarios. Results feed into an interactive real-time dashboard for visualization, historical trend analysis, and congestion modeling. Detection confidence thresholds and anchor box optimization ensure precise localization. The system logs vehicle counts for weekly traffic pattern analysis, enabling authorities to identify peak hours and optimize signal timings. Automating detection and counting eliminates the need for fixed sensors and manual calibration, reducing costs. By integrating deep learning with efficient image processing, this scalable solution supports smart city initiatives, enhances road safety, mitigates congestion, and informs data-driven transportation planning across municipalities.

Keywords: YOLOv4, Vehicle Detection, Real-time Traffic Monitoring, OpenCV, Vehicle Classification, Smart City, Traffic Analytics.

1. INTRODUCTION

Real-time vehicle counting and classification systems harness advanced computer vision and machine learning techniques to automatically detect, track, and categorize vehicles from live video streams, providing continuous traffic metrics without intrusive road sensors. Early traffic monitoring relied on inductive loops and piezoelectric sensors installed beneath road surfaces, which delivered point measurements but required costly installation and maintenance. Vision-based methods emerged in the early 2000s, using background subtraction and contour analysis to distinguish vehicles from the road background, achieving over 90 % accuracy in highway scenarios. The advent of convolutional neural networks (CNNs) around 2015 enabled more robust detection and classification under occlusion and variable lighting, with architectures like Faster R-CNN and SSD delivering real-time performance on GPUs. YOLOv4, introduced in 2020, further pushed the frontier, achieving 65 FPS and 43.5 % AP on COCO by incorporating CSP connections and Mosaic augmentation. In India, where over 300 million registered vehicles traverse a 66-lakh km road network, manual traffic surveys and fixed sensors cover only a fraction of critical intersections. Congestion in major metros such as Delhi and Bengaluru incur annual economic losses exceeding ₹1.5 lakh crore due to delays and fuel waste. Deploying camera-based, non-intrusive systems enables continuous, type-specific traffic counts—cars, trucks, buses, motorcycles—and supports dynamic signal control, tolling, and infrastructure planning. By integrating background subtraction, YOLO-based detection, and line-crossing algorithms, modern implementations achieve over 30 FPS on standard GPUs, ensuring scalability for smart-city initiatives. This introduction frames the evolution, technical foundations, and pressing need for real-time vehicle counting and classification in rapidly urbanizing environments.

2. LITERATURE SURVEY

Alpatov et al. [1] considered road situation analysis tasks for traffic control and ensuring safety. The following image processing algorithms are proposed: vehicle detection and counting algorithm, road marking detection algorithm. The algorithms are designed to process images obtained from a stationary camera. The developed vehicle detection and counting algorithm was implemented and tested also on an embedded platform of smart cameras. Song et al. [2] proposed a vision-based vehicle detection and counting system. A new high-definition highway vehicle dataset with a total of 57,290 annotated instances in 11,129 images is published in this study. Compared with the existing public datasets, the proposed dataset contains annotated tiny objects in the image, which provided the complete data foundation for vehicle detection based on deep learning. Neupane et al. [3] created a training dataset of nearly 30,000 samples from existing cameras with seven classes of vehicles. To tackle P2, this trained and applied transfer learning-based fine-tuning on several state-of-the-art YOLO (You Only Look Once) networks. For P3, this work proposed a multi-vehicle tracking algorithm that obtains the per-lane count, classification, and speed of vehicles in real time.

Lin et al. [4] presented a real-time traffic monitoring system based on a virtual detection zone, Gaussian mixture model (GMM), and YOLO to increase the vehicle counting and classification efficiency. GMM and a virtual detection zone are used for vehicle counting, and YOLO is used to classify vehicles. Moreover, the distance and time traveled by a vehicle are used to estimate the speed of the vehicle. In this study, the Montevideo Audio and Video Dataset (MAVD), the GARM Road-Traffic Monitoring data set (GRAM-RTM), and our collection data sets are used to verify the proposed method. Chauhan et al. [5] used the state-of-the-art Convolutional Neural Network (CNN) based object detection models and train them for multiple vehicle classes using data from Delhi roads. This work gets upto 75% MAP on an 80-20 train-test split using 5562 video frames from four different locations. As robust network connectivity is scarce in developing regions for continuous video transmissions from the road to cloud servers, this work also evaluated the latency, energy and hardware cost of embedded implementations of our CNN model-based inferences. Arinaldi et al. [6] presented a traffic video analysis system based on computer vision techniques. The system is designed to automatically gather important statistics for policy makers and regulators in an automated fashion. These statistics include vehicle counting, vehicle type classification, estimation of vehicle speed from video and lane usage monitoring. The core of such system is the detection and classification of vehicles in traffic videos. This work implemented two models for this purpose, first is a MoG + SVM system and the second is based on Faster RCNN, a recently popular deep learning architecture for detection of objects in images.

Gomaa et al. [7] presented an efficient real-time approach for the detection and counting of moving vehicles based on YOLOv2 and features point motion analysis. The work is based on synchronous vehicle features detection and tracking to achieve accurate counting results. The proposed strategy works in two phases; the first one is vehicle detection and the second is the counting of moving vehicles. For initial object detection, this work has utilized state-of-the-art faster deep learning object detection algorithm YOLOv2 before refining them using K-means clustering and KLT tracker. Then an efficient approach is introduced using temporal information of the detection and tracking feature points between the framesets to assign each vehicle label with their corresponding trajectories and truly counted it. Oltean et al. [8] proposed an approach for real time vehicle counting by using Tiny YOLO for detection and fast motion estimation for tracking. This application is running in Ubuntu with GPU processing, and the next step is to test it on low-budget devices, as Jetson Nano. Experimental results showed that this approach achieved high accuracy at real time speed (33.5 FPS) on real traffic videos. Pico et al. [9] proposed the implementation of a low-cost system to identify and classify vehicles using an Embedded ARM based platform (ODROID XU-4) with Ubuntu operating system. The algorithms used are based on the Open-source library (Intel OpenCV) and implemented in Python programming language. The experimentation carried out proved that the efficiency of the algorithm implemented was 95.35%, but it can be improved by increasing the training sample.

Tituana et al. [10] reviewed different previous works developed in this area and identifies the technological methods and tools used in those works; in addition, this work also presented the trends in this area. The most relevant articles were reviewed, and the results were summarized in tables and figures. Trends in the used methods are discussed in each section of the present work. Khan et al. [11] aimed of this work is that a cost-effective vision-based vehicle counting and classification system that is mainly implemented in OpenCV utilising Python programming and some methods of image processing. Balid et al. [12] reported on the development and implementation of a novel smart wireless sensor for traffic monitoring. Computationally efficient and reliable algorithms for vehicle detection, speed and length estimation, classification, and time-synchronization were fully developed, integrated, and evaluated. Comprehensive system evaluation and extensive data analysis were performed to tune and validate the system for a reliable and robust operation.

Jahan et al. [13] presented convolutional neural network for classifying four types of common vehicle in our country. Vehicle classification plays a vital role of various application such as surveillance security system, traffic control system. This work addressed these issues and fixed an aim to find a solution to reduce road accident due to traffic related cases. To classify the vehicle, this work used two methods feature extraction and classification. These two methods can straight forwardly be performed by convolutional neural network. Butt et al. [14] proposed a convolutional neural network-based vehicle classification system to improve robustness of vehicle classification in real-time applications. This work presented a vehicle dataset comprising of 10,000 images categorized into six-common vehicle classes considering adverse illuminous conditions to achieve robustness in real-time vehicle classification systems. Initially, pretrained AlexNet, GoogleNet, Inception-v3, VGG, and ResNet are fine-tuned on self-constructed vehicle dataset to evaluate their performance in terms of accuracy and convergence. Based on better performance, ResNet architecture is further improved by adding a new classification block in the network. Gonzalez et al. [15] showed a vision-based system to detect, track, count and classify moving vehicles, on any kind of road. The data acquisition system consists of a HD-RGB camera placed on the road, while the information processing is performed by clustering and classification algorithms. The system obtained an efficiency score over the 95 percent in test cases, as well, the correct classification of 85 percent of the test objects.

3. PROPOSED SYSTEM

The research work begins with the acquisition of real-time video streams as the primary input source, typically sourced from surveillance cameras or traffic monitoring systems. The first module, Background Subtraction and ROI, plays a crucial role in isolating moving objects, i.e., vehicles, from the stationary background.

This is achieved using advanced algorithms to create a Region of Interest (ROI) that narrows down the area for subsequent analysis, reducing computational load and minimizing false positives. The heart of the system lies in the Vehicle Detection and Tracking module, which employs the YOLO (You Only Look Once) deep learning model. YOLO excels at real-time object detection and tracking, enabling the system to identify vehicles within the defined ROI and track their movements across frames, allowing for continuous monitoring. The next step involves the Vehicle Classification module, powered by the Darknet framework, which classifies the detected vehicles into specific categories, such as cars, trucks, or motorcycles. Following this, the Counting and Analytics module quantifies vehicle movements, including counting, speed measurement, and other relevant analytics, providing valuable data for traffic management and research purposes. Finally, the system generates a video output that overlays processed information onto the original video feed, offering a user-friendly visual representation of the vehicle counting and classification results, making it a versatile tool for various applications in traffic monitoring, urban planning, and transportation research.

The detailed operation illustrated as follows:

Step 1: This is the starting point of system. Acquire real-time video streams as input data for the system. These video streams could come from surveillance cameras, traffic cameras, or any source capturing vehicle movements.

Step 2: Background subtraction is a crucial step for isolating moving objects (vehicles) from the stationary background. In this module, use algorithms and techniques to detect the background and create a Region of Interest (ROI) where vehicle detection will occur. This step helps reduce noise and focus on the relevant area.

Step 3: Use YOLO (You Only Look Once), a deep learning-based object detection model, for vehicle detection. YOLO can efficiently detect and locate objects in real-time video frames. It identifies the vehicles within the defined ROI and can track their movements across frames, allowing us to follow vehicles as they move through the video.

Step 4: After detecting and tracking vehicles, further analyze and classify them using the Darknet framework. Darknet is a neural network framework well-suited for classification tasks. It can classify vehicles into different categories such as cars, trucks, motorcycles, or any other relevant classes.

Step 5: In this step, count and analyze the detected and classified vehicles. We can track the number of vehicles passing through specific points or regions of interest, calculate vehicle speed, and gather other relevant analytics data. This information can be useful for traffic management, surveillance, or research purposes.

Step 6: Finally, the system provides video output with the processed information overlaid on the original video feed. This output can include counted vehicles, their classifications, and any other relevant data. It allows users to visualize and interpret the results of the vehicle counting and classification system.

3.1 Vehicle Detection and Tracking

After applying foreground extraction module, proper contours are acquired, Features of these contours such as centroid. Aspect ratio, area, size and solidity are extracted and are used for the classification of the vehicles. This module consists of three steps, background subtraction, image enhancement and foreground extraction. Background is subtracted so that foreground objects are visible. This is done usually by static pixels of static objects to binary 0. After background subtraction image enhancement techniques such as noise filtering, dilation and erosion are used to get proper contours of the foreground objects. The result obtained from this module is the foreground.

Region of Interest selection: In the very first frame of the video, ROI is defined by drawing a close line on the image. The goal is to recognize that ROI in a later frame, but that ROI is not a salient vehicle. It is just a part of a vehicle, and it can deform, rotate, translate and even not be fully in the frame.

Vehicle Detection: Active strategy to choose a search window for vehicle detection using an image context was proposed GMM framework to capture the vehicle by sequential actions with top-down attention. It has achieved satisfactory performance on vehicle detection benchmark, by sequentially refining the bounding boxes. Proposed a sequential search strategy to detect visual vehicles in images, where the detection model was trained by proposed a deep RL framework to select a proper action to capture a vehicle in an image.

Vehicle Counting: In this module detected vehicles will be counted and these counted results will be updated frequents based on vehicle detection, results will be printed streaming video using OpenCV.

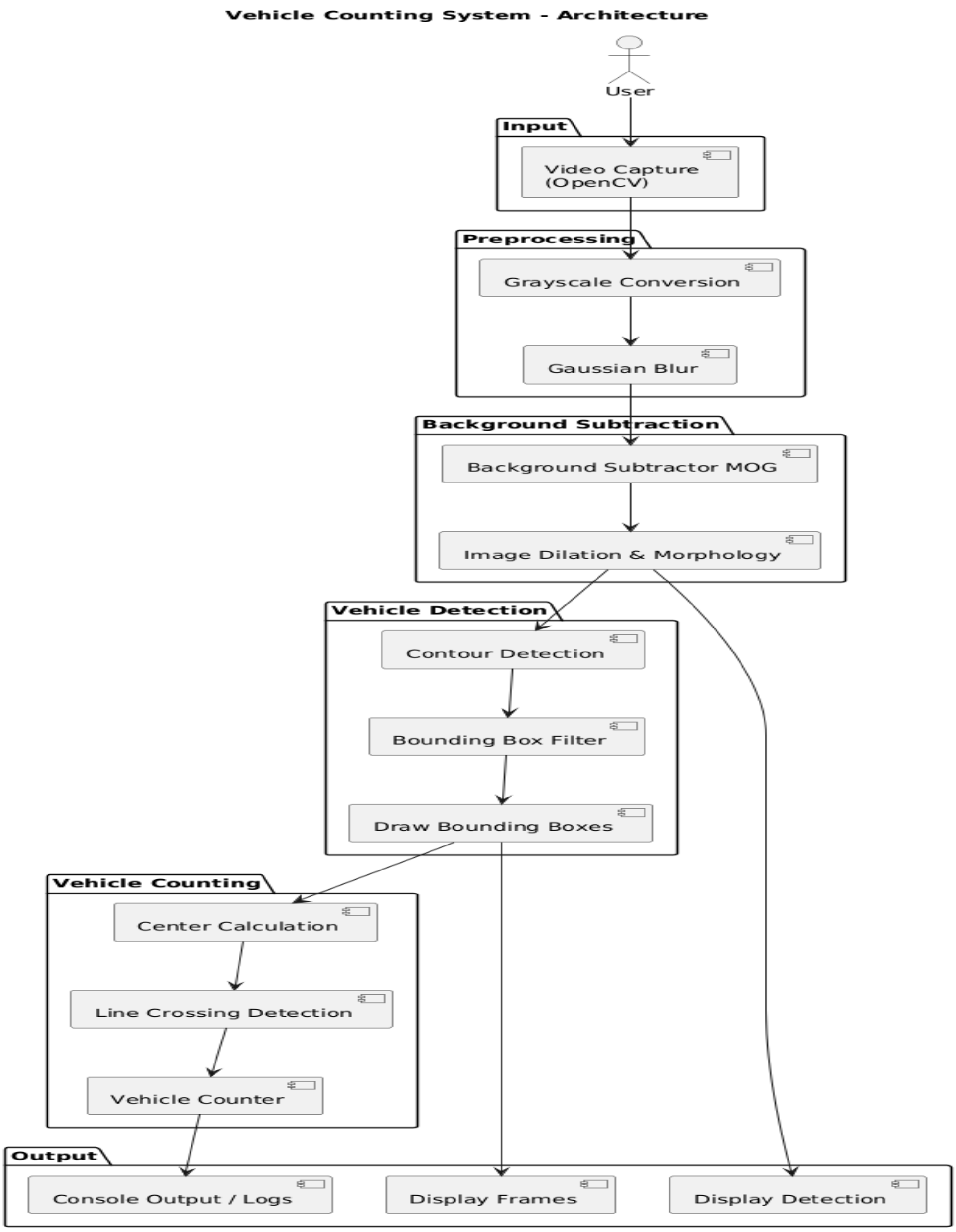


Fig. 1: Block diagram of proposed system.

3.2 YOLO-V3 Model

Object detection is a phenomenon in computer vision that involves the detection of various objects in digital images or videos. Some of the objects detected include people, cars, chairs, stones, buildings, and animals. This phenomenon seeks to answer two basic questions:

What is the object? This question seeks to identify the object in a specific image.

Where is it? This question seeks to establish the exact location of the object within the image.

Object detection consists of various approaches such as fast R-CNN, Retina-Net, and Single-Shot MultiBox Detector (SSD). Although these approaches have solved the challenges of data limitation and modeling in object detection, they are not able to detect objects in a single algorithm run. YOLO

www.ijbar.org
ISSN 2249-3352 (P) 2278-0505 (E)
Cosmos Impact Factor-5.86

algorithm has gained popularity because of its superior performance over the aforementioned object detection techniques.

YOLO Definition: YOLO is an abbreviation for the term 'You Only Look Once'. This is an algorithm that detects and recognizes various objects in a picture (in real-time). Object detection in YOLO is done as a regression problem and provides the class probabilities of the detected images.

YOLO algorithm employs convolutional neural networks (CNN) to detect objects in real-time. As the name suggests, the algorithm requires only a single forward propagation through a neural network to detect objects. This means that prediction in the entire image is done in a single algorithm run. CNN is used to predict various class probabilities and bounding boxes simultaneously. The YOLO algorithm consists of various variants. Some of the common ones include tiny YOLO and YOLOv3.

Importance of YOLO: YOLO algorithm is important because of the following reasons:

- Speed: This algorithm improves the speed of detection because it can predict objects in real-time.
- High accuracy: YOLO is a predictive technique that provides accurate results with minimal background errors.
- Learning capabilities: The algorithm has excellent learning capabilities that enable it to learn the representations of objects and apply them in object detection.

YOLO algorithm working: YOLO algorithm works using the following three techniques:

- Residual blocks
- Bounding box regression
- Intersection Over Union (IOU)

Residual blocks: First, the image is divided into various grids. Each grid has a dimension of S x S. The following Figure 2 shows how an input image is divided into grids. In the Figure 2, there are many grid cells of equal dimension. Every grid cell will detect objects that appear within them. For example, if an object center appears within a certain grid cell, then this cell will be responsible for detecting it.

Figure 2. Example of residual blocks.

Bounding box regression: A bounding box is an outline that highlights an object in an image. Every bounding box in the image consists of the following attributes:

- Width (bw)

- Height (bh)
- Class (for example, person, car, traffic light, etc.)- This is represented by the letter c.
- Bounding box center (bx,by)

The following Figure 3 shows an example of a bounding box. The bounding box has been represented by a yellow outline. YOLO uses a single bounding box regression to predict the height, width, center, and class of objects. In the image above, represents the probability of an object appearing in the bounding box.

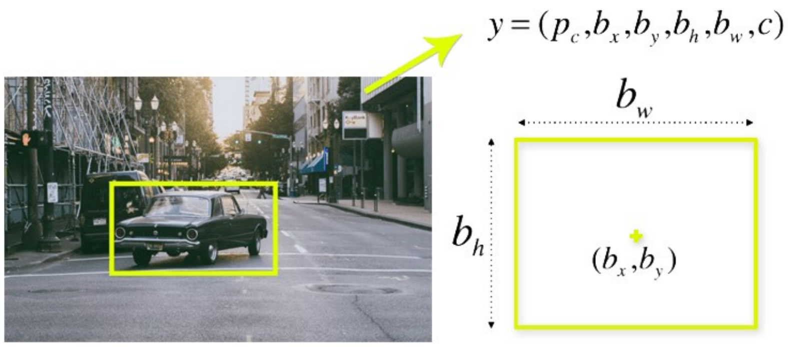


Figure 3. Bounding box regression

Intersection over union (IOU): Intersection over union (IOU) is a phenomenon in object detection that describes how boxes overlap. YOLO uses IOU to provide an output box that surrounds the objects perfectly. Each grid cell is responsible for predicting the bounding boxes and their confidence scores. The IOU is equal to 1 if the predicted bounding box is the same as the real box. This mechanism eliminates bounding boxes that are not equal to the real box.

Combination of the three techniques: The following image shows how the three techniques are applied to produce the final detection results.

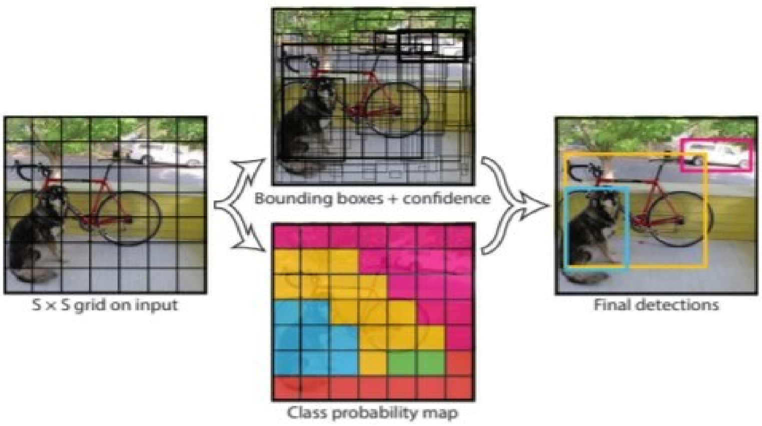
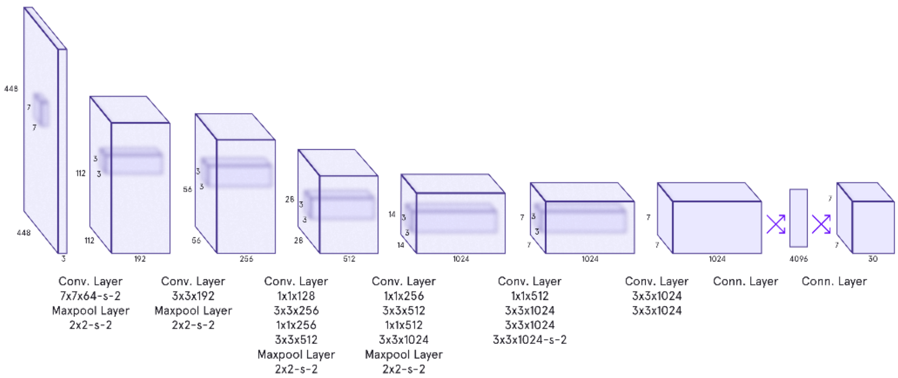


Figure 4. Combination of three modules.

First, the image is divided into grid cells. Each grid cell forecasts B bounding boxes and provides their confidence scores. The cells predict the class probabilities to establish the class of each object. For example, we can notice at least three classes of objects: a car, a dog, and a bicycle. All the predictions are made simultaneously using a single convolutional neural network. Intersection over union ensures that the predicted bounding boxes are equal to the real boxes of the objects. This phenomenon eliminates unnecessary bounding boxes that do not meet the characteristics of the objects (like height and width). The final detection will consist of unique bounding boxes that fit the objects perfectly. For example, the car is surrounded by the pink bounding box while the bicycle is surrounded by the yellow bounding box. The dog has been highlighted using the blue bounding box. The YOLO algorithm takes an image as input and then uses a simple deep convolutional neural network to detect objects in the image. The architecture of the CNN model that forms the backbone of YOLO is shown below.



The Architecture. Our detection network has 24 convolutional layers followed by 2 fully connected layers. Alternating 1x1 convolutional layers reduce the features space from preceding layers. We pretrain the convolutional layers on the ImageNet classification task at half the resolution (224x224 input image) and then double the resolution for detection.

YOLO Layers.

The first 20 convolution layers of the model are pre-trained using ImageNet by plugging in a temporary average pooling and fully connected layer. Then, this pre-trained model is converted to perform detection since previous research showcased that adding convolution and connected layers to a pre-trained network improves performance. YOLO's final fully connected layer predicts both class probabilities and bounding box coordinates. YOLO divides an input image into an S × S grid. If the center of an object falls into a grid cell, that grid cell is responsible for detecting that object. Each grid cell predicts B bounding boxes and confidence scores for those boxes. These confidence scores reflect how confident the model is that the box contains an object and how accurate it thinks the predicted box is.

YOLO predicts multiple bounding boxes per grid cell. At training time, we only want one bounding box predictor to be responsible for each object. YOLO assigns one predictor to be "responsible" for predicting an object based on which prediction has the highest current IOU with the ground truth. This leads to specialization between the bounding box predictors. Each predictor gets better at forecasting certain sizes, aspect ratios, or classes of objects, improving the overall recall score. One key technique used in the YOLO models is non-maximum suppression (NMS**)**. NMS is a post-processing step that is used to improve the accuracy and efficiency of object detection. In object detection, it is common for multiple bounding boxes to be generated for a single object in an image. These bounding boxes may overlap or be located at different positions, but they all represent the same object. NMS is used to identify and remove redundant or incorrect bounding boxes and to output a single bounding box for each object in the image.

4. RESULTS AND DISCUSSION

Figure 5 represents a single frame from a video. In this frame: The object detection model has identified a car in the frame, and it has assigned a high confidence score to this classification, indicating a 98% accuracy in identifying it as a car. Figure 6 represents the frame from another video. In this frame, there are multiple objects detected: a laptop, book, dining table and a cup. The object detection model has identified the objects in the frame, and it has assigned a high confidence score to this classification. Figure 7 represents yet another frame from the same video. In this frame, there are multiple objects detected, including persons, cars, and buses. The object detection model has identified persons in the frame with a high confidence score, indicating a 94% accuracy in identifying them as persons. The model has also detected cars in the frame, and it is highly confident in this classification, indicating a 96% accuracy in identifying them as cars. Furthermore, the model has detected buses in the frame with an extremely high confidence score, indicating a 99% accuracy in identifying them as buses.

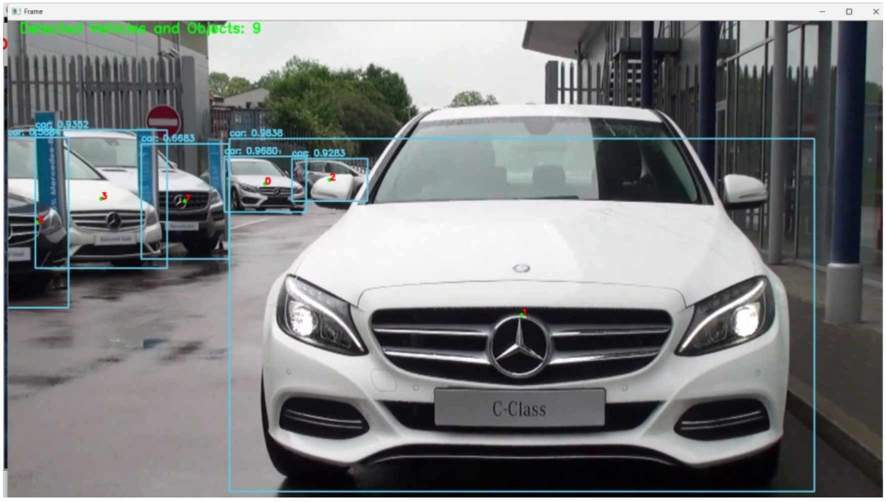


Figure 5: Frame with cars vehicle classification.

Figure 6: video frame with classification of both person and motor bike.

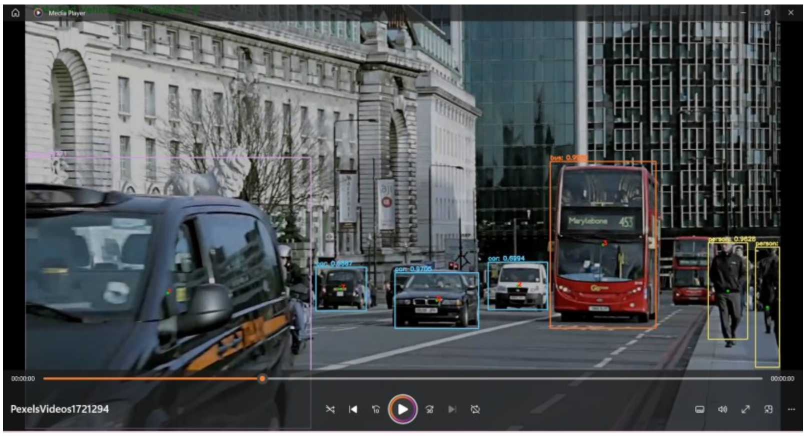


Figure 7: video frame with classification of persons, cars & buses with an accuracy of 94% ,96% & 99%.

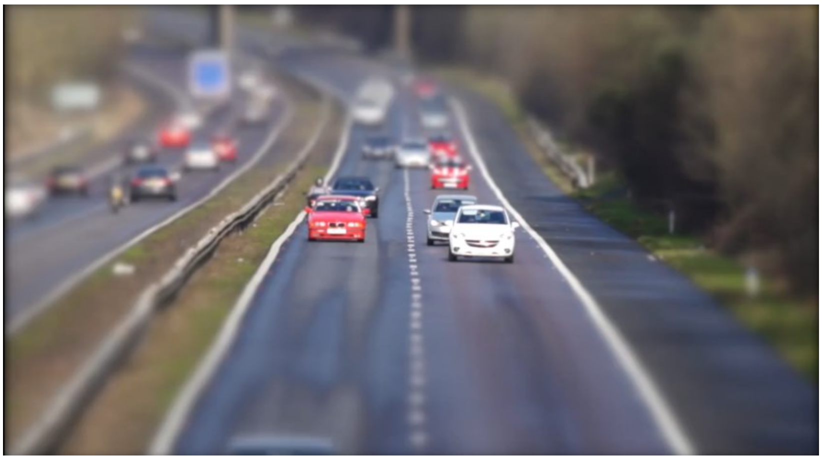


Figure 8: Shows the input video given to the system.

Figure 9: Shows the result given in the output screen and objects are classifier.

5. CONCLUSION

The proposed solution is implemented on python, using the OpenCV bindings. The traffic camera footages from variety of sources are in implementation. A simple interface is developed for the user to select the region of interest to be analyzed and then image processing techniques are applied to calculate vehicle count and classified the vehicles using machine learning algorithms. From experiments it is apparent that CC method outperforms than BoF and SVM method in all results and gives more close classification results to the ground truth values.

REFERENCES

[1] Alpatov, Boris & Babayan, Pavel & Ershov, Maksim. (2018). Vehicle detection and counting system for real-time traffic surveillance. 1-4. 10.1109/MECO.2018.8406017.

[2] Song, H., Liang, H., Li, H. et al. Vision-based vehicle detection and counting system using deep learning in highway scenes. Eur. Transp. Res. Rev. 11, 51 (2019). https://doi.org/10.1186/s12544-019-0390-4.

[3] Neupane, Bipul et al. “Real-Time Vehicle Classification and Tracking Using a Transfer Learning-Improved Deep Learning Network.” Sensors (Basel, Switzerland) vol. 22,10 3813. 18 May. 2022, doi:10.3390/s22103813.

[4] C. J Lin, Shiou-Yun Jeng, Hong-Wei Lioa, "A Real-Time Vehicle Counting, Speed Estimation, and Classification System Based on Virtual Detection Zone and YOLO", Mathematical Problems in Engineering, vol. 2021, Article ID 1577614, 10 pages, 2021. https://doi.org/10.1155/2021/1577614.

[5] M. S. Chauhan, A. Singh, M. Khemka, A. Prateek, and Rijurekha Sen. 2019. Embedded CNN based vehicle classification and counting in non-laned road traffic. In Proceedings of the Tenth International Conference on Information and Communication Technologies and Development (ICTD '19). Association for Computing Machinery, New York, NY, USA, Article 5, 1–11. https://doi.org/10.1145/3287098.3287118.

[6] A. Arinaldi, J. A. Pradana, A. A. Gurusinga, “Detection and classification of vehicles for traffic video analytics”, Procedia Computer Science, Volume 144, 2018, Pages 259-268, ISSN 1877-0509, https://doi.org/10.1016/j.procs.2018.10.527.

[7] Gomaa, A., Minematsu, T., Abdelwahab, M.M. et al. Faster CNN-based vehicle detection and counting strategy for fixed camera scenes. Multimed Tools Appl 81, 25443–25471 (2022). https://doi.org/10.1007/s11042-022-12370-9.

[8] G. Oltean, C. Florea, R. Orghidan and V. Oltean, "Towards Real Time Vehicle Counting using YOLO-Tiny and Fast Motion Estimation," 2019 IEEE 25th International Symposium for Design and Technology in Electronic Packaging (SIITME), 2019, pp. 240-243, doi: 10.1109/SIITME47687.2019.8990708.

[9] L. C. Pico and D. S. Benítez, "A Low-Cost Real-Time Embedded Vehicle Counting and Classification System for Traffic Management Applications," 2018 IEEE Colombian Conference on Communications and Computing (COLCOM), 2018, pp. 1-6, doi: 10.1109/ColComCon.2018.8466734.

[10] D. E. V. Tituana, S. G. Yoo and R. O. Andrade, "Vehicle Counting using Computer Vision: A Survey," 2022 IEEE 7th International conference for Convergence in Technology (I2CT), 2022, pp. 1-8, doi: 10.1109/I2CT54291.2022.9824432.

[11] A. Khan, A., Sabeenian, R.S., Janani, A.S., Akash, P. (2022). Vehicle Classification and Counting from Surveillance Camera Using Computer Vision. In: Suma, V., Baig, Z., K.

Shanmugam, S., Lorenz, P. (eds) Inventive Systems and Control. Lecture Notes in Networks and Systems, vol 436. Springer, Singapore. https://doi.org/10.1007/978-981-19-1012-8_31.

[12] W. Balid, H. Tafish and H. H. Refai, "Intelligent Vehicle Counting and Classification Sensor for Real-Time Traffic Surveillance," in IEEE Transactions on Intelligent Transportation Systems, vol. 19, no. 6, pp. 1784-1794, June 2018, doi: 10.1109/TITS.2017.2741507.

[13] N. Jahan, S. Islam and M. F. A. Foysal, "Real-Time Vehicle Classification Using CNN," 2020 11th International Conference on Computing, Communication and Networking Technologies (ICCCNT), 2020, pp. 1-6, doi: 10.1109/ICCCNT49239.2020.9225623.

[14] M. A. Butt, A. M. Khattak, S. Shafique, B. Hayat, S. Abid, Ki-Il Kim, M. W. Ayub, A. Sajid, A. Adnan, "Convolutional Neural Network Based Vehicle Classification in Adverse Illuminous Conditions for Intelligent Transportation Systems", Complexity, vol. 2021, Article ID 6644861, 11 pages, 2021. https://doi.org/10.1155/2021/6644861.

[15] P. Gonzalez, Raul & Nuño-Maganda, Marco Aurelio. (2014). Computer vision based real-time vehicle tracking and classification system. Midwest Symposium on Circuits and Systems. 679-682. 10.1109/MWSCAS.2014.6908506.