# The Analysis of Deep Learning Based Vehicle Classification, Tracking and Speed Estimation System

Arkadeepto Majumder [1], Chaitanya Anand [2], Ardhendu Kundu [3]
Department of Computer Science and Engineering, Institute of Engineering & Management
Salt Lake, Kolkata - 700011, India
[1] – arkadeepto@gmail.com, [2] – anandchaitanya12@gmail.com, [3] – ardhendu.kundu@iem.edu.in

***Abstract*—Traffic congestion and accidents are emerging as significant social problems due to technological advancements and urban population growth. While computer vision is valuable for detecting traffic incidents, its performance is often hindered by environmental and technological constraints, resulting in inefficiencies. To improve traffic safety and efficiency, deep learning algorithms like Convolutional Neural Networks (CNNs) and object detection models such as You Only Look Once (YOLO) and Faster Region based Convolutional Neural Network (R-CNN) are increasingly utilized. A system using object tracking techniques calculates and displays the speed of moving vehicles in real-time using models like YOLO-NAS and Deep SORT. The project, implemented with Python and OpenCV, analyses video streams to detect vehicles and calculate their speeds, employing non-maximum suppression and Kalman Filters to enhance tracking accuracy.**

## I. Introduction

As the urban population increases exponentially coupled with rapid technological advancements, there is extreme traffic congestion in many densely populated cities with ineffective transportation systems due to the substantial rise in the number of vehicles and the capacity of the current transportation systems [1]. This situation is not one that can be solved by simply building more highways and subways, as not only the resources for such infrastructural development are scarce, but at the same time, these steps have been proven to be moot in solving the problem of traffic management [2].

An efficient substitute for addressing traffic congestion is a traffic management and monitoring system, which is a completely automated system for gathering traffic information such as the count, class and speed of vehicles. The collection of these data is important for such traffic management system to perform its functions which is to perform analysis on the traffic to better utilize the roadway systems and facilitate the safety of transportation [3]. The goal of such automated traffic surveillance system is to remove the need for human labour for simple vision tasks that can be performed efficiently by computer systems.

Currently, along with various data collection and processing technologies in the field of Internet of Things, a lot of traffic monitoring cameras have been placed at congestion prone areas of intersections, highways, etc [4]. However, most of the monitoring systems still work in the tradition way of collecting raw video data and transmitting and storing it. This video data is then analysed manually that presents a significant problem due to the large volume of the data that need to be processed and analysed [5]. These kinds of data can be automatically extracted from video footage using a variety of machine learning methods, which can decrease the inefficiency.

With great progress in the field of machine learning and artificial intelligence (AI), various methods were proposed to tackle the problem of vehicle tracking and classification. The traditional vehicle detection and classification models such as: i) Scale Invariant Feature Transform (SIFT) for feature matching and extraction; ii) Gaussian Mixture model for moving vehicle detection; iii) License Plate extraction method; iv) Histogram of Oriented Gradient (HOG) and Support Vector Machine (SVM) [6]. Most contemporary tracking and classification algorithms rely on deep learning models and techniques, such as You Only Look Once (YOLO), Convolutional Neural Networks (CNNs), and Region-based Convolutional Neural Networks (RCNNs), etc. This paper aims at analysing the existing methods and models used to achieve the goal of tracking and classification of vehicles.

## II. Literature Review

### A. *Traditional Methods (GMMs, SVMs)*

Tarun Kumar et al. propose an elementary but efficient approach to detect and provide an estimate of the speed of vehicles using a single point CCTV camera using image processing. MySQL is utilized as

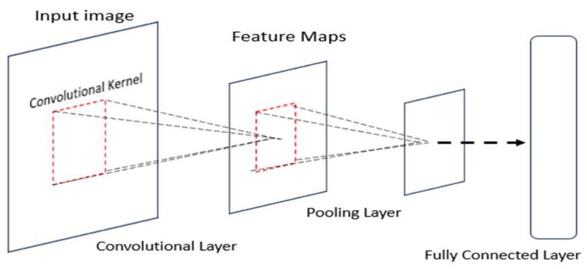

*Figure 1 Basic Architecture of CNN*

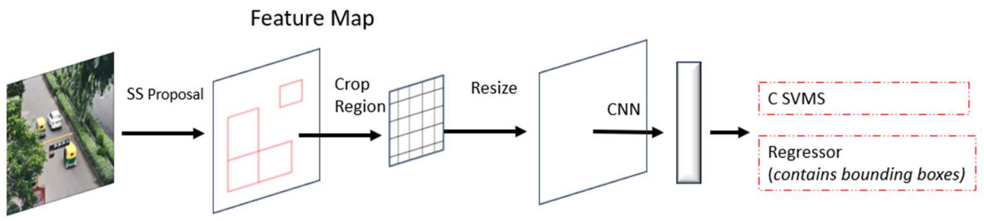

*Figure 2 Simplified Architecture of RCNN*

the database, and OpenCV and JavaCV are employed to implement the suggested strategy. Although the calibration of camera is strict to suit this model of detection, this flaw can be easily handled through modern tools and models. Regardless, the innovation of this approach lies in the selection of Region of Interest for vehicle detection. The accuracy of detection through this proposed method was found to be 87.7% with the maximum accuracy found to be 98.3% [7].

The goal of Chen et al.'s work was to efficiently extract an image of a car from video [8]. To lessen the detrimental effects of outside elements such shadows, camera vibration, changes in illumination, etc. on vehicle classification, the authors apply the background Gaussian Mixture Model (GMM) and the shadow removal technique [9]. Vehicle tracking is accomplished with the Kalman filter, while vehicle classification is accomplished with SVM [10]. Five categories were used to classify the vehicles: cars, vans, buses, motorcyclists, and unknown vehicles. For these vehicle categories, the categorization accuracy was 94.6%

### B. *Deep Learning Algorithms (CNNs, RCNNs)*

Minglan Sheng et al. suggested a approach to use convolutional neural networks (CNNs) for feature extraction and classification [6]. CNNs have been shown to be effective in identifying vehicle types from different angles and scenes. The architecture a basic CNN model is shown in fig. 1. However, there are still challenges in accurately identifying vehicles in real traffic situations, some of which are low resolution of images, ambiguity due to uneven aspect ratios, and biased or imbalanced training datasets. Researchers have employed techniques such as the use of depth learning for vehicle positioning and classification to tackle these challenges. These traditional methods, however, do produce results with satisfactory accuracy, especially individually. Therefore, there is a need for further research to improve the accuracy and robustness of vehicle detection and classification methods.

At the time they were first invented, RCNNs were computationally expensive. However, with the development of new techniques like Fast R-CNN, their cost has substantially decreased. [11]. A simple architecture of a RCNN is shown in fig. 2. A fully-convolutional network (FCN) can be created by combining the Fast R-CNNs and RCNNs using an alternate training procedure that alternates between fine-tuning for the region proposal task and then fine-tuning for object identification, with the proposals remaining fixed. This method is proposed by Shaoqing Ren et al. [12]. This scheme is used to build the unified convolutional network which was tested against the benchmarks put forth by PASCAL VOC and it resulted in improvement in region proposal quality13].

The fact that different elements of a vehicle are treated similarly is one factor contributing to the lower accuracy of image processing and feature

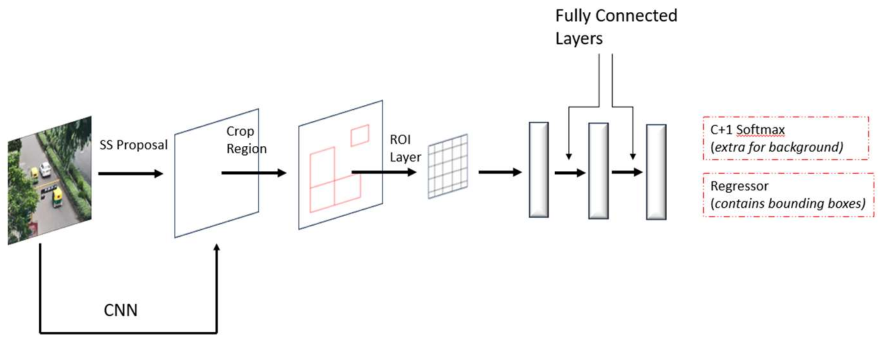
Fully Connected Layers
SS Proposal
Crop Region
ROI Layer
C+1 Softmax (extra for background)
Regressor (contains bounding boxes)
CNN

*Figure 3 Architecture of Fast-RCNN*

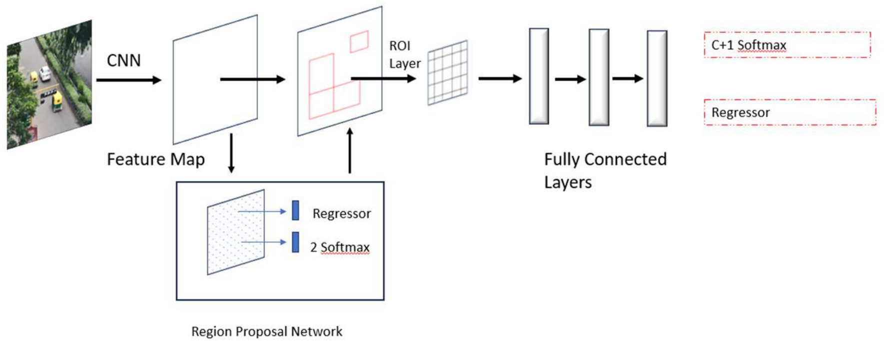

*Figure 4 Simplified Architecture of Faster-RCNN*

extraction using machine learning algorithms. Zhao et al. attempt to address the detection issue in spite of the image of an automobile having missing portions. [14]. They conducted tests to categorize cars into five groups: coaches, SUVs, trucks, vans, and sedans. They were 97.9% accurate in their classifications.

### *C. Modern Tools and Models (TensorFlow, Deep SORT, YOLO)*

Modern object detection models can be broadly classified on the basis of the number of passes that are made in order to detect objects in the frame. There are one-pass models and two-pass models. Two pass models are more computationally expensive than their one-pass counterparts, but they provide higher accuracy. Modern two-pass models include R-CNN based models such as fast and faster R-CNN. Modern one pass models include YOLO, YOLOv2 and SSD.

Girshick et al. proposed Fast R-CNN to address the limitations of RCNN [15]. For the entire image, Fast R-CNN calculated a feature map from which fixed length region features were extracted. Furthermore, ROI Pooling layer was used to extract region features as shown in fig. 3 [16], [17]. Fast R-CNN outperformed R-CNN and SPP-net in terms of detection accuracy and had faster training and inference times, with the provision of end-to-end optimisation for feature extraction, bounding box prediction and region classification [18].

There are already many detector variations based on Faster R-CNN for various applications [19]–[22]. Despite advancements in learning-based detectors, generating object proposals still depended on older methods like Selective Search or Edge Boxes. These strategies relied on low-level visual cues and were not suited to data-driven improvements. [23], [24]. To solve this problem, Faster R-CNN introduced a new approach called the Region Proposal Network (RPN), as shown in fig. 4 [12]. The RPN can be trained using supervised learning methods. An n×n sliding window is used by the RPN to generate a feature vector for every point. This feature vector is input into two simultaneous branches: one for bounding box regression to fine-tune the object's position, and another for object classification to determine whether the proposition is an object. Bounding box localization and final object categorization are then performed using the outputs from these branches.

YOLO (You Only Look Once) is a real-time detector created by Redmon et al. YOLO partitioned the image into a set number of grid cells (e.g., 5 × 5 grid) and treated object detection as a regression issue [25]. A prediction was created for each cell that included the following details: whether or not the box contained an object, the object's bounding box's width and height, its coordinates, and the class of the object as depicted in fig. 5 [26]. However, the algorithm could only detect up to two objects in a single box and the

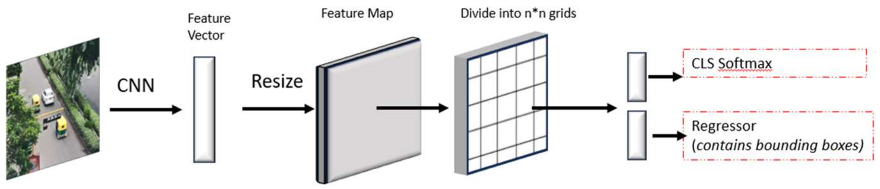

*Figure 5 Simplified Architecture of YOLO*

prediction was hindered due to change in aspect ratios.

Moving away from variations of the convolutional neural networks (CNNs), Muhammad Azhad bin Zuraimi and Fadhlan Hafizhelmi propose a vehicle detection and tracking system using TensorFlow library with Deep SORT algorithm based on YOLOv4 model to perform Real time traffic surveillance. The proposed model had an accuracy of 82.08 [27].

Cheng-Jian Lin et al. have presented a real-time traffic monitoring system that addresses the challenges of accurate vehicle detection and classification in traffic flows, particularly dealing with total occlusions [28]. They employed convolutional neural networks, specifically utilizing YOLO algorithm. Their solution improved vehicle counting and classification efficiency by integrating a virtual detection zone, Gaussian mixture model (GMM), and YOLO. The method also estimated vehicle speed. The highest classification accuracy achieved with YOLOv4 was 98.91% in MAVD and 99.5% in GRAM-RTM datasets. The method also showed strong performance in different environmental conditions with average absolute percentage error for vehicle speed estimation being approximately 7.6.

## III. Objective

Our system utilizes computer vision and object tracking techniques to calculate and display speed of moving vehicles. The system receives a video feed as input, and it uses a deep learning model for object detection to identify cars in each frame.

There are several object detection models employed, including YOLO, Fast R-CNN, and R-CNN. After training on the COCO Annotated Vehicle dataset, their accuracy metrics are evaluated. The weights and configuration files of the pretrained models are loaded into OpenCV, which is used to detect objects in real-time using the trained models. Outputs in different formats are provided by each model, so bounding boxes and confidence scores are obtained through careful parsing. In the current implementation, the transition has been made from the YOLOv4 model to YOLO-NAS (Neural Architecture Search), which offers a more streamlined output and parsing system [30].

Furthermore, the research focuses on implementing Deep SORT for vehicle tracking using a Kalman Filter to forecast and update each vehicle's status across a frame—including its location, speed, and acceleration. This approach aims to enhance tracking accuracy by refining predictions based on observed detections

Our future research aims to determine whether congestion is caused by a slow-moving vehicle through:

- Velocity of Buses: The speed of buses will be determined by analysing their movement across frames. Congestion may be indicated by a significant drop in speed compared to vehicles in earlier time frames.
- Surrounding Vehicle Density: The density of traffic near the bus will be analysed. A traffic jam can be signalled by high vehicle density with slow movement.

## IV. Implementation of Proposed Model

The project was implemented using a Python script that utilizes the OpenCV library and the YOLO object detection framework to analyse a video stream, identify vehicles, and calculate their speeds based on their movement between two predefined lines in the video frame. Configuration files and pre-trained weights are fed into the YOLO model, and the COCO dataset is used for class labels. The video stream is obtained from a .avi file , and the output is saved to an MP4 file with obtained results.

The script processes each frame of the video, detects vehicles using YOLO, and draws bounding boxes around them. Two horizontal lines are drawn on the video frame to represent positions where vehicle speeds will be calculated. The script tracks the time when each vehicle crosses the first line, and when it crosses the second line, it calculates the vehicle's speed based on the known real-life distance between the two lines. The calculated speed is then displayed on the video frame along with the bounding box and label for each detected vehicle.

Additionally, the deployment of a Gaussian Mixture Model for image segmentation of frames in sample videos can be tested using the scikit-learn library as implemented in fig. 7.

In order to filter out overlapping bounding boxes during the frame-by-frame processing, the script employs multi-object detection and non-maximum suppression. The calculated speeds and visual annotations are included in the output video. The processing continues until the user presses 'q' to exit the video playback. This script is useful for traffic monitoring applications, providing a practical example of computer vision and object detection in a real-world context.

After detection, our project utilizes an object tracking program. This algorithm assigns a unique identifier to each vehicle across the sequence of video frames, enabling individual vehicle tracking as they move within the frame. Deep SORT (Deep Simple Online Realtime Tracking) extracts features from each detected bounding box using a separate CNN to distinguish between objects of the same type [31], [32]. We have updated our model from YOLOV4 to YOLO-NAS and applied Deep SORT as shown in fig. 8. Deep SORT maintains a state using a Kalman Filter for each object, which typically includes the object's location, velocity, acceleration, and other dynamic parameters. Based on the object's prior estimated state, the Kalman Filter predicts the object's position and velocity which constitute its new state [33]. With each new input and after associating each detection with a track, the Kalman Filter updates its state with the new measurement. This process refines predictions by integrating actual observed detections, thereby enhancing tracker accuracy [34]

## V. Results

The following results have been obtained after using 3 different models to detect and segment objects in a sample video whose screen capture is shown in fig. 8. Faster-RCNN algorithm using the VGG16 model, YOLOv4 and YOLO-NAS have been used.

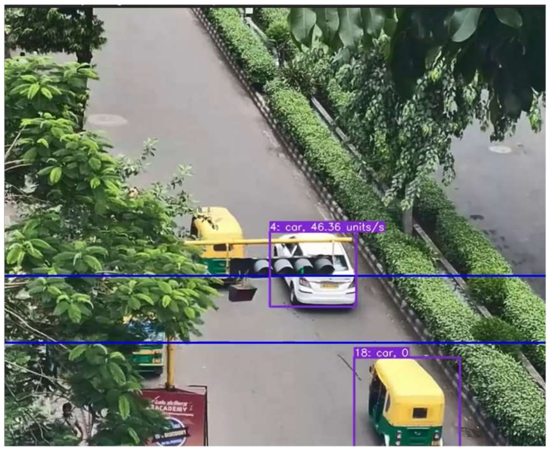

*Figure 6 Combined Result of YOLO NAS and Deep SORT*

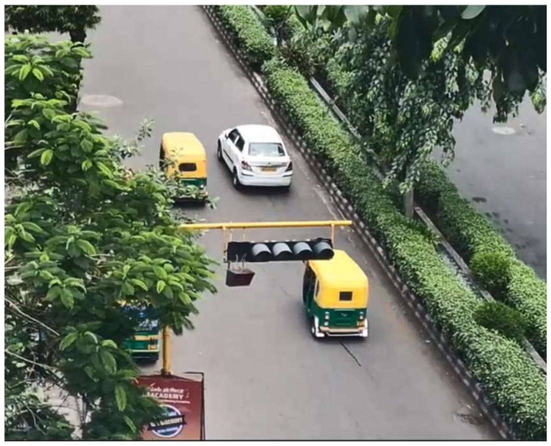

*Figure 8 Sample screen capture used for comparison*

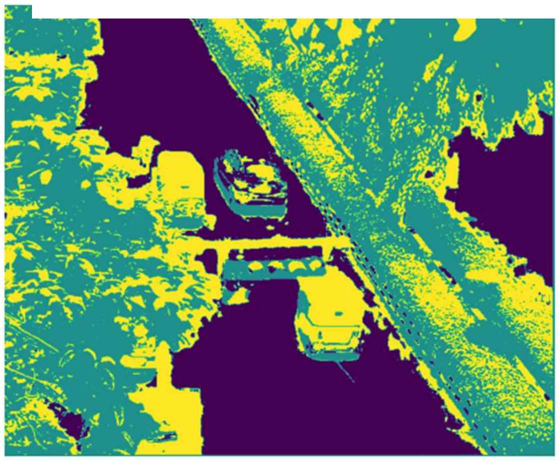

*Figure 7 Result of Gaussian Mixture Algorithm*

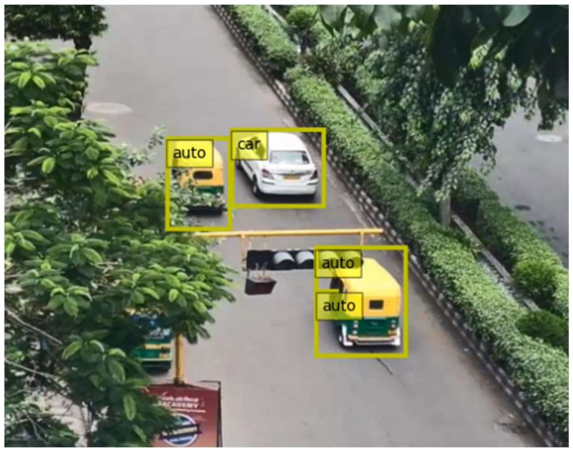

*Figure 9 Result of FASTER-RCNN*

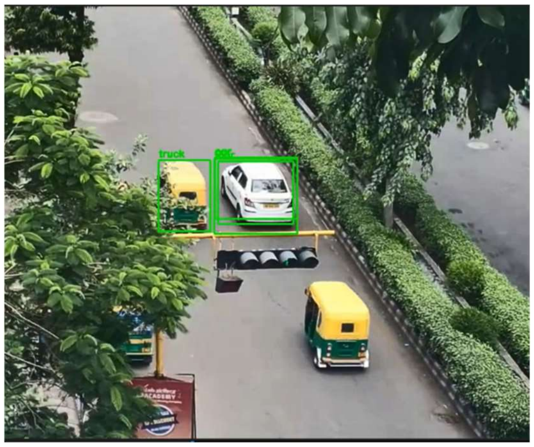

*Figure 10 Result of YOLOv4*

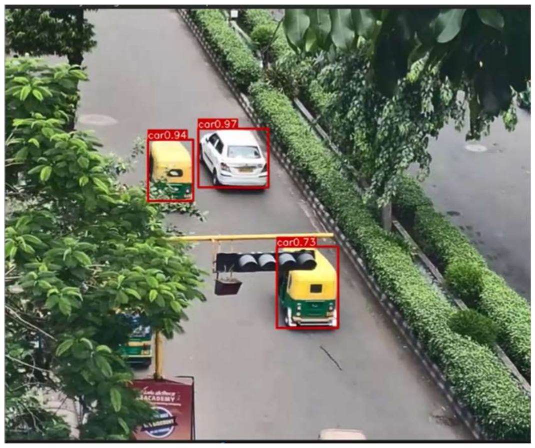

*Figure 11 Result of YOLO-NAS*

| Model | Latency (ms) | mAP 0.5:0.95 | reference |
|---|---|---|---|
| Faster-RCNN (VGG16) | 27 | 0.73 | [11] |
| YOLOv4 | 8.7 | 0.435 | [32] |
| YOLOv8 | 7.1 | 0.509 | [34] |
| YOLO-NAS | 5.8 | 0.523 | [33] |

Table 1: Comparison of different models based on latency and mAP 0.5:0.95

Faster-RCNN:

The faster-RCNN model, being a two-pass algorithm has been more accurate than its counterparts. The prediction of the model is shown in fig. 9. However, it has been significantly slower in obtaining the results compared the other two models as they are one-pass algorithms.

YOLOv4:

YOLOv4 model obtained results with lower latency compared to faster-RCNN model, but it fell short of the latest YOLO-NAS model [35]. Additionally, it struggled with tracking of objects through continuous frames, evidence of which can be seen in the fig. 10.

YOLO-NAS:

This model provided results with the lowest latency and has been successful in tracking objects through individual frames [36]. Along with detection, segmentation and tracking, calculation of speed of vehicles has been performed with the help of this model. The following result has been obtained when the same video was given as input to the model described in fig. 11.

The evaluation metrics of the pretrained models were obtained from their respective papers and repositories and compiled in table I.

## VI. Discussion

Over the last decade, vehicle classification systems have seen significant advancement. Recent developments in wireless communication, machine learning, and sensor technologies have dramatically increased classification accuracy at a significantly lower cost.

Machine learning techniques are becoming increasingly important in automobile classification systems. However, in order to train and develop a successful classification model, a sizable amount of data must be gathered in order to attain high classification accuracy. The latency and accuracy of faster-RCNN, YOLOv4, YOLOv8, and YOLO-NAS were compared, with YOLO-NAS found to offer the better accuracy with shorter latency. Thus the YOLONAS model was complemented with DeepSORT to track the vehicles and measure their speed [38].

The classification model training procedure involves manual labelling, which takes a lot of time and work. Several challenges were encountered, including the high computational power requirements for training custom models, difficulties in measuring model accuracy with our functions, and traditional vehicle tracking issues. These challenges necessitated the research and employment of the Deep SORT algorithm.

The advent of self-driving cars have led to opening of a whole new avenue with regards to the management of traffic systems [29]. Autonomous Traffic Management (ATM) is a popular research subject in Intelligent Transportation Systems (ITS) that aims to minimize traffic congestion through a well-connected infrastructure [39]. Autonomous vehicles and their interconnected surroundings must effectively navigate through dynamically shifting traffic scenarios and road conditions [40]. Vehicle congestion detection and traffic management can be facilitated by our project by comparing drop in speeds of buses and other large vehicles and analysing the change in density surrounding the vehicle.

## VII. Conclusion

A summary of contemporary traffic surveillance systems has been provided, with an emphasis on the essential features of vehicle tracking, detection, and speed computation systems. By categorizing the vehicle classification systems into 3 categories based on the tools used by authors in implementing the system, mainly the traditional methods (SVMs, GMMs), the Deep Learning Techniques (CNNs, RCNNs) and the modern methods that provide an inbuilt set of tools to aid in classification and detection such as TensorFlow, Colab, YOLO, etc. The project has been implemented using a Python script that utilizes the OpenCV library and the YOLO object detection framework with deep SORT to analyze a video stream, identify vehicles, and calculate their speeds based on their movement between two predefined lines in the video frame. Future study directions and some of the drawbacks of these suggested solutions were explored.

## References

[1] M. Won, T. Park, and S. H. Son, “Toward mitigating phantom jam using vehicle-to-vehicle communication,” *IEEE Trans. Intell. Transp*, vol. 18, pp. 1313–1324, May 2017.

[2] M. T. Masood, “Transportation problems in developing countries pakistan: a case-in-point,” *International Journal of Business and Management*, vol. 6, no. 11, p. 256, 2011.

[3] A. Arinaldi, J. A. Pradana, and A. A. Gurusinga, “Detection and classification of vehicles for traffic video analytics,” in *Procedia Computer Science*, vol. 144, 2018, pp. 259–268.

[4] H. Lu, Z. Sun, and W. Qu, “: Big data and its applications in urban intelligent transportation system,” *J. Transp. Syst. Eng. Inf. Technol*, vol. 15, no. 5, 2015.

[5] R. Ravish and S. R. Swamy, “Intelligent traffic management: A review of challenges, solutions, and future perspectives,” *Transport and Telecommunication Journal*, vol. 22, no. 2, pp. 163–182, 2021.

[6] M. Sheng, C. Liu, Q. Zhang, L. Lou, and Y. Zheng, “Vehicle detection and classification using convolutional neural networks,” in *2018 IEEE 7th Data Driven Control and Learning Systems Conference (DDCLS)*, 2018.

[7] T. Kumar and D. S. Kushwaha, “An efficient approach for detection and speed estimation of moving vehicles,” in *Procedia Computer Science*, vol. 89, 2016, pp. 726–731.

[8] Z. Chen, T. Ellis, and S. A. Velastin, “Vehicle detection tracking and classification in urban traffic,” in *Proc. 15th International IEEE Conference on Intelligent Transportation Systems*, Sep. 2012, pp. 951–956.

[9] Z. Chen, N. Pears, M. Freeman, and J. Austin, “Background subtraction in video using recursive mixture models spatiotemporal filtering and shadow removal,” in *Proc. International Symposium on Visual Computing*, 2009, pp. 1141– 1150.

[10] M. Won, “Intelligent traffic monitoring systems for vehicle classification: A survey,” *IEEE Access*, vol. 8, pp. 73340–73358, 2020.

[11] R. Girshick, “Fast r-cnn,” *arXiv:1504.08083*, 2015.

[12] S. Ren and et al., “Faster r-cnn: Towards real-time object detection with region proposal networks,” *Advances in Neural Information Processing Systems*, vol. 28, 2015.

[13] M. Everingham, L. V. Gool, C. K. I. Williams, J. Winn, and A. Zisserman, “The pascal visual object classes challenge 2007 (voc2007) results,” 2007.

[14] D. Zhao, Y. Chen, and L. Lv, “Deep reinforcement learning with visual attention for vehicle classification,” *IEEE Transactions on Cognitive and Developmental Systems*, vol. 9, no. 4, pp. 356–367, Dec. 2017.

[15] R. Girshick, “Fast r-cnn,” in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1440–1448.

[16] J. Kleban, X. Xie, and W.-Y. Ma, “Spatial pyramid mining for logo detection in natural scenes,” in *2008 IEEE International Conference on Multimedia and Expo*. IEEE, 2008, pp. 1077–1080.

[17] K. He, X. Zhang, S. Ren, and J. Sun, “Spatial pyramid pooling in deep convolutional networks for visual recognition,” *IEEE transactions on pattern analysis and machine intelligence*, vol. 37, no. 9, pp. 1904–1916, 2015.

[18] X. Wu, D. Sahoo, and S. C. Hoi, “Recent advances in deep learning for object detection,” *Neurocomputing*, vol. 396, pp. 39–64, 2020.

[19] T.-Y. Lin, P. Dollar, R. Girshick, K. He, B. Hariharan, and´ S. Belongie, “Feature pyramid networks for object detection,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 2117–2125.

[20] Z. Cai and N. Vasconcelos, “Cascade r-cnn: Delving into high quality object detection,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 6154–6162.

[21] T. Kong, A. Yao, Y. Chen, and F. Sun, “Hypernet: Towards accurate region proposal generation and joint object detection,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 845–853.

[22] S. Bell, C. L. Zitnick, K. Bala, and R. Girshick, “Insideoutside net: Detecting objects in context with skip pooling and recurrent neural networks,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2874–2883.

[23] J. R. Uijlings, K. E. Van De Sande, T. Gevers, and A. W. Smeulders, “Selective search for object recognition,” *International journal of computer vision*, vol. 104, pp. 154–171, 2013.

[24] C. L. Zitnick and P. Dollar, “Edge boxes: Locating object´ proposals from edges,” in *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*. Springer, 2014, pp. 391–405.

[25] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You only look once: Unified, real-time object detection,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 779–788.

[26] J. Redmon and A. Farhadi, “Yolo9000: better, faster, stronger,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 7263–7271.

[27] M. A. B. Zuraimi and F. H. K. Zaman, “Vehicle detection and tracking using yolo and deepsort,” in *2021 IEEE 11th IEEE Symposium on Computer Applications Industrial Electronics (ISCAIE)*, 2021.

[28] T.-H. M. et al., “A real-time vehicle counting, speed estimation, and classification system based on virtual

detection zone and yolo," *Mathematical Problems in Engineering*, vol. 2021, p. 1577614, 2021.

[29] A. Mushtaq, I. U. Haq, M. U. Imtiaz, A. Khan, and O. Shafiq, "Traffic flow management of autonomous vehicles using deep reinforcement learning and smart rerouting," *IEEE Access*, vol. 9, pp. 51005–51019, 2021.

[30] J. Terven and D. Cordova-Esparza, "A comprehensive review of yolo: From yolov1 to yolov8 and beyond," *arXiv preprint arXiv:2304.00501*, 2023.

[31] X. Hou, Y. Wang, and L.-P. Chau, "Vehicle tracking using deep sort with low confidence track filtering," in *2019 16th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*. IEEE, 2019, pp. 1–6.

[32] A. Pujara and M. Bhamare, "Deepsort: real time & multiobject detection and tracking with yolo and tensorflow," in *2022 International Conference on Augmented Intelligence and Sustainable Systems (ICAISS)*. IEEE, 2022, pp. 456– 460.

[33] Y. Kim, H. Bang *et al.*, "Introduction to kalman filter and its applications," *Introduction and Implementations of the Kalman Filter*, vol. 1, pp. 1–16, 2018.

[34] G. Welch, G. Bishop *et al.*, "An introduction to the kalman filter," 1995.

[35] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "Yolov4: Optimal speed and accuracy of object detection," 2020.

[36] S. Aharon, Louis-Dupont, Ofri Masad, K. Yurkova, Lotem Fridman, Lkdci, E. Khvedchenya, R. Rubin,
N. Bagrov, B. Tymchenko, T. Keren, A. Zhilko, and Eran-Deci, "Super-gradients," 2021. [Online]. Available: https://zenodo.org/record/7789328

[37] G. Jocher, A. Chaurasia, and J. Qiu, "Ultralytics yolov8," 2023. [Online]. Available: https://github.com/ultralytics/ ultralytics

[38] A. Majumder, C. Anand, "Code repository," 2024. [Online]. Available: https://github.com/Arkadeepto2002/ Vehicle-classification-and-speed-detection

[39] S. E. Hamdani and N. Benamar, "Autonomous traffic management: Open issues and new directions," in *Proc. International Conference on Selected Topics in Mobile and Wireless Networking (MoWNeT)*, Jun. 2018, pp. 1–5.

[40] M. V. Rajasekhar and A. K. Jaswal, "Autonomous vehicles: The future of automobiles," *2015 IEEE International Transportation Electrification Conference (ITEC)*, pp. 1–6, 2015. [Online]. Available: https://api.semanticscholar.org/ CorpusID:8766010