Performance Analysis of Deep Learning YOLO Models for South Asian Regional Vehicle Recognition

Minar Mahmud Rafi, Siddharth Chakma, Asif Mahmud, Raj Xavier Rozario, Rukon Uddin Munna, Md. Abrar Abedin Wohra, Rakibul Haque Joy, Khan Raqib Mahmud, Bijan Paul*

Department of Computer Science and Engineering
University of Liberal Arts Bangladesh
Dhaka, Bangladesh

***Abstract*—For years, humans have pondered the possibility of combining human and machine intelligence. The purpose of this research is to recognize vehicles from various media and while there are multiple models associated with this, there has not been enough testing and training of models when it comes to vehicle datasets that originate from countries like Bangladesh, India, etc. Our focus was to assimilate the largest dataset of vehicles exclusive to South Asia in addition to the more common universal vehicles and apply it to track and recognize these vehicles, even in motion. To develop this, we increased the class variations and quantity of the data. We trained different versions of the YOLOv5 model with our dataset to properly measure the degree of accuracy between the models in detecting the more unique vehicles. If vehicle detection and tracking are adopted and implemented in live traffic camera feeds, the information can be used to create smart traffic systems that can regulate congestion and routing by identifying and separating fast and slow-moving vehicles on the road. The comparison between the three different YOLOv5 models led to an analysis that indicates that the large variant of the YOLOv5 architecture outperforms the rest**.

***Keywords*—You Only Look Once (YOLOv5); vehicle detection; neural network; deep learning; vehicle tracking**

I. Introduction

Advancements in automobile manufacturing have given rise to more affordable cars which has resulted in over five million registered vehicles [1] coasting through the roads of Bangladesh. Road infrastructures in this country were not designed to hold the growing number of vehicles which presents grave environmental and health concerns. Given the circumstance, congestion is inevitable, and this significantly contributes to the rising air and noise pollution levels in the city. To circumvent this obstacle, restless drivers resort to maneuvering chaotically without any regard to traffic rules and are thus responsible for most of the road fatality cases in the country. One of the biggest obstacles is that traditional methods of prevention such as traffic lights and pedestrian crossings are not sufficient because they are generally ignored.

To address this issue, an intuitive system is needed to observe traffic patterns and direct different vehicles into proper lanes. Most vehicles in South Asia are very different than those in the western world as they differ drastically in shapes, sizes, and colors. This is a major challenge the algorithm will face [2] as it needs to differentiate between these vehicles to identify them individually. The height and angle at which these vehicles are posed and captured also factor into this problem. Datasets that include traditionally used South Asian vehicles are scarce and do not contain the required amount of data which presents a separate challenge. Due to the erratic nature of traffic in South Asian countries, different CNN models that are usually tested in other environments have not been applied enough to see how they perform in the tumultuous streets of cities like Dhaka. A major challenge of our research is that we have had data scarcity, particularly for South Asian vehicles.

Machine learning has progressed enough to make use of traffic cameras [3] to track vehicles and their patterns. Additionally, using Neural Network-based Object Detection can produce valuable tracking and surveillance data that could be essential to coming up with a solution to the traffic problem. Further applications in the division of slow- and fast-moving vehicles and the identification of missing vehicles can also be pursued through Deep Learning. Smart traffic systems [4] can utilize these applications to reduce mishaps while also improving the flow of traffic. Autonomously driven cars [5] can also employ the previously mentioned applications to avoid different vehicles, clogged roads, and potential accidents while on the road.

However, one of the key difficulties in using machine learning algorithms is the requirement of a vast amount of data to train a model. In this research, we develop a sizable vehicle dataset from scratch and train a model to accurately recognize them. The intention was to set our work apart from conventional vehicle detection systems. Our research is distinctive in that we have curated a dataset consisting of 21 classes of vehicles commonly available worldwide and those that are only seen in South Asian regions. Unique vehicles like rickshaws, human haulers, three-wheelers, etc. all vary in build and proportion. The collected images are put through a lengthy process of cleaning, augmenting, and finally labeling through bounding box annotations. To address the data scarcity issue, we used different augmentation techniques to balance the dataset. This is done to ensure we have enough data for accurate testing and training. We chose a well-known object detection algorithm called YOLOv5 (You Only Look Once) [6] to use in our model for training and we compared the

*Corresponding Author.

performance of different architectures of YOLOv5 models: Small, Medium, and Large. Previous versions [7] were an option but the new update presented a more efficient and time-saving alternative. YOLOv5's hyperparameters are tweaked to accurately detect objects in real-time through bounding box coordinates of objects from the carefully labeled data it has been trained with. This model will be able to recognize different native vehicles which can then be used in systems to reduce the traffic problem that plagues South Asian cities.

II. Literature Review

The subject of object detection has attracted attention from various independent researchers over the years. Different detection systems were used over the years for identifying objects. LIDAR (Light Detection and Ranging) was used in the form of sensors attached to both vehicles and certain points of the road to detect oncoming vehicles [8]. Other non-intrusive methods like ASFF (Adaptive Spatial Feature Fusion) [9] and Radar Sensors [10] were also used. The interest in this field and its innovations date back to the 1970s [11].

Object detection [12] by the camera has become more prevalent in recent years and more accurate and cost-effective than other sensors. To accurately identify vehicles, real-time detection speed and high accuracy are required for a quick response to fast-moving vehicles and to get a reduced latency. While most algorithms repurpose classifiers by taking images at multiple scales and locations and applying the algorithm to perform detection [13], YOLO applies a neural network that dissects an image into different parts and can predict bounding box regions based on predicted probabilities [14]. YOLO detects objects using a single inference which makes it faster than its peers, SSD (Single Shot Detector) and Faster R-CNN (Region-based Convolutional Neural Network) [15].

Research by Liu & Zhang [16] aimed to improve the standard YOLOv3 model by training it to adapt to actual traffic conditions and applying a scale prediction layer to improve the detection accuracy of large vehicles. They use the k-means++ algorithm to improve the efficiency of the anchor box dimension clustering as shown in Fig. 1. The resulting F-YOLOv3 algorithm clocked in at 91.12% on the mAP (Mean Average Precision) accuracy scale beating out Faster R-CNN at 90.01% and base YOLOv3 at 78.68%. The recognition performance of large vehicles is poor when compared to small vehicles because of their contrasting characteristics.

Fig. 1. Anchor Box Dimension Clustering [16].

Redmon [17] suggested the integration of classification and localization into a single convolutional neural network which would improve the speed at the cost of precision. While it achieves a combined accuracy of 75.0%, the model has difficulty in detecting smaller grouped objects due to the spatial constraints imposed by YOLO and objects in different aspect ratios. Chandan suggested a different approach [18] where he opted to use OpenCV to detect objects in a python environment with the assistance of the Single Shot Detector algorithm. This algorithm used optical flow and background subtraction to achieve an optimal accuracy in detecting standard vehicle classes and this was a great basis for comparison with the earlier versions of YOLO. A detection system for localized mobile environments like roads and railways was made by Chen [19]. Using the same COCO dataset, they compared YOLOv3, and the Single Shot Detector mentioned above to find their efficiency and applicability in traffic. It was found that YOLOv3 had attained an 85% score over SSD's 79.5% in terms of mAP at high resolution. A more recent comparison of YOLOv4 with SSD and Faster R-CNN was conducted by Kim [20] for real-time vehicle detection. After evaluating the different models, it was observed that YOLOv4 performed at 98.1% precision while Faster R-CNN and SSD performed at 93.4% and 90.5% respectively.

Phillips suggested a system [21] for distance estimation between vehicles in traffic to avoid collision by mounting a monocular camera to a vehicle dashboard. It is fitted with systems for object tracking and detection and is modular enough to switch out systems for other uses. Errors in estimation increase as the distance increases. Wang has used edge Detection technology [22] to demonstrate the detection of objects such as vehicles by their outer edge lines. The edge detection technique must remove noises from an image background using a higher threshold before identification of the vehicle in question can begin. This changes how we can detect vehicles from a certain height and makes detection possible using image-capturing objects like Drones.

Sokalski [23] produced another alternative that combines edge detection with color identification to differentiate between artificial and natural objects. The only drawback is the process of extracting the nine features from various channels of each color in the image which are used to define the edges. An identification approach by thickness estimation and edge detection was put forward by Kanistras [24] where angle vectors of an elevated image would be determined in its edge guide. These vectors are constantly changing by determining the standard deviation of slope vectors therefore pre-defining edges to detect vehicles.

Different datasets and their use in creating a large diverse dataset in the training of algorithms are discussed by Xiao & Kang [25]. This paper has influenced how we approached diversifying and enriching our dataset to obtain satisfactory test results. The paper also provides tips for being efficient in collecting and labeling datasets properly. The importance of data augmentation is made clear by Zoph [26] in his discussions about how different augmentation strategies such as rotating, shearing, equalizing, changing colors, etc. can not only expand the dataset but also increase accuracy up to 6% but at the cost of data loss during training.

A YOLO-based traffic counting system developed by Lin [27] was employing three different pieces. The detector generates the bounding boxes of the vehicles, the buffer stores the vehicle coordinates, and the counter is responsible for counting the vehicles. Images/videos are put through the detector where it passes through filters and then the YOLO algorithm. Data access is built from the frame number input and output, previous and current array in buffer, and vehicle counting algorithm in counter. Checkpoints are also added for validation of detection whereas the overall accuracy is determined by using a video that has a different height and angular perspective. The counting accuracy seen during the day was around 95% but dropped to unfavorable rates at night due to factors such as headlight exposure, dim streetlights, etc. which was later improved upon by implementing night vision technology. An alternate YOLO method created by Tao [28] removes the last two layers of the connected system and adds a pooling layer. This is faster than its peers and the addition of a pre-processing procedure for night images enhances detection in darkness by removing highlights and modifying contrast and brightness. This new optimized O-YOLO algorithm executes an accuracy of 66% on a standard VOC dataset and 80.1% on a custom-curated dataset. Corovic [29] implemented YOLO to detect objects in real-time traffic and pre-trained the algorithm to detect them in five categories. These were cars, trucks, civilians, signs, and lights. Tests were conducted to prove that YOLO was suitable for real-time detection and in different weather. They deducted detection could be improved in place of obstructions by incorporating datasets that contain weather conditions into the training. After training of 120 epochs was conducted, the accuracy had a steady increase from 18.98% to 46.60% but failed to climb to higher rates due to many occluded objects in the dataset. Salarpour [30] realized an algorithm to track multiple vehicles using the Kalman filter and background subtraction. A region-based algorithm is then combined with the filter to track and predict the region of the vehicle in the continuing frame while also using its color and size to get an accurate result of 96%. This method helps detect issues such as occlusion and clutter with minimal loss of accuracy.

Occlusion makes it hard for vehicles to be distinguished for detection and therefore a procedure to reduce dense occlusion from surveillance cameras was put forward by Phan [31]. This is also a combination of background subtraction and detection but mixed with occlusion detection where each occluded vehicle is extracted from images based on their features. The method improves the accuracy of detection in occluded images at higher angles proven by its 85% accuracy score during high traffic. To address the problem of detecting vehicles at varying scales and distances, Lu [32] produced a modified version of the Region Proposal Network (RPN) which is tailored to be scale aware during detection. This system has two different sub-networks to detect large and small case proposals and inputs through two separate XGBoost (Extreme Gradient Boosting) algorithms to create final predictions. Both algorithms boasted scores of 64.1% and 84.8% respectively in terms of precision.

The most consistently accurate model out of the many commented on above is the YOLOv4 algorithm. It offers a staggering 98.1% mean average precision when applied over a vehicle recognition system. YOLOv5, which currently lacks substantial research documentation, has data that exhibits improved accuracy and speeds over its previous iteration [33] which will be nothing but beneficial to our training.

To train a varied dataset such as this, we needed a sustainable system powerful enough to process and execute all the data. The model we used in conjunction with neural networks was built to enhance every aspect of its previous build and therefore the dataset was processed much quicker than expected. The dataset itself is a mixture of both common and unique vehicles found here in the South of Asia but the rarity in variety of some of these vehicles was a complication. 21 assorted classes were selected, collected, processed, and augmented to create a robust dataset for this research.

III. Architecture

The method of detecting items in an image as shown in Fig. 2 and calculating their location using bounding boxes is known as object detection. The classification of images is concerned with determining whether an object exists in each image based on calculative likelihood. Images have characteristics like distinct edges that an object recognition method must extract. Convolutional Neural Networks, Auto Encoder techniques, and others, can be used to automate this procedure. The most effective object identification strategy is one that assures that all objects of vivid size are given a bounding box to be recognized, as well as having high computational capabilities allowing for faster processing. Both YOLO and SSD promise good outcomes, but there is a speed/accuracy trade-off.

A. Proposed Methodology

In the South Asian region, we began gathering images of various vehicle items. The dataset was sorted and categorized after the image collections were completed to prepare it for the machine learning model. This dataset was then split at random using the standard splitting technique, with 80% of the data going to the training set for training and 20% going to the validation set for validation.

Fig. 2. Detection Process during a Training Phase.

This is done to assess the model's correctness while avoiding over-fitting. The prediction model was evaluated using the set developed for the validation procedure, whereas the training set provided the algorithm. We were given numerous metrics and statistics to evaluate because of the outcome. The whole process is outlined in Fig. 3.

The YOLO model is part of the Fully Convolutional Neural Networks (FCN) family that allows for the most optimally achievable outcome and real-time object recognition in every end-to-end model. YOLOv5 as shown in Fig. 4 contains a variety of internal models, each with its own set of complexity and architectures. The largest and most sophisticated network is YOLOv5l, which is followed by YOLOv5m and YOLOv5s.

Each of these models was trained to see how the various architectures influence the overall model's speed and accuracy. The three different and crucial aspects of YOLOv5's architecture can be outlined.

Backbone - The backbone is primarily responsible for resolving gradient information and incorporating changes into feature maps, hence lowering parameter numbers and the overall model's FLOPS (Floating Point Operation per Second).

Neck - The neck improves the data flow within the model. It includes a feature pyramid network with an upgraded bottom-up approach that can expose new low-level complicated features, as well as localization signals in lower layers that can improve localization accuracy. Through interconnection provided by Adaptive Feature Pooling, the feature grids, and levels produce useful data on all levels.

Head - The head contains algorithms for creating feature maps of many sizes with prediction procedures that the model employs to recognize objects of many sizes. Because vehicles might have components and add-ons of various shapes and sizes, this technique is critical for vehicle recognition. Each of these items can be easily detected using the multi-scale detection feature. When the training on the model is started in YOLOv5, the procedure begins.

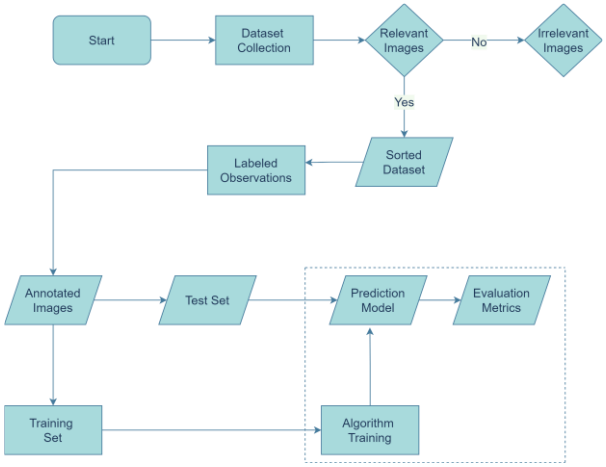

Fig. 3. Block Diagram of Vehicle Detection & Tracking Process.

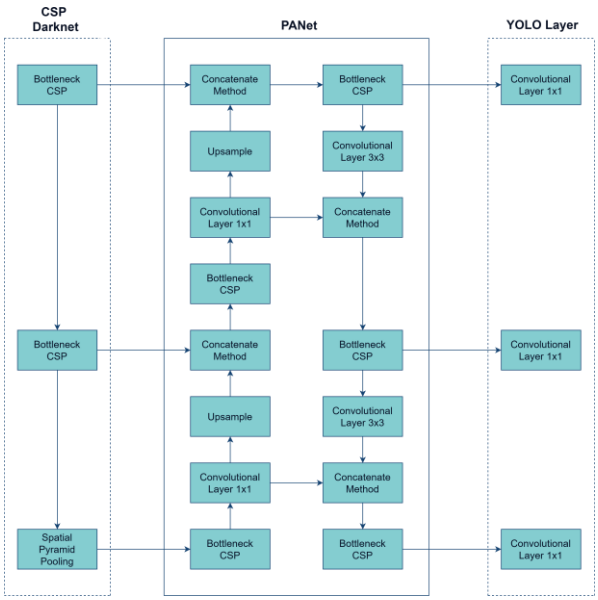

Fig. 4. YOLOv5 Architecture

The data is fed into the Sparked module, which extracts the features, which are subsequently transmitted to the PANet (Path Aggregation Network) [34] module to be fused.

It is all gathered in the YOLO layer, which is then processed to produce important analytic data like class, location, score, and size.

B. Data Annotation Format in YOLOv5

Each image was annotated in the form of an a.txt file, with each line of the content record depicting a bounding box. In Fig. 5, for example, there are five items (car, three-wheeler, motorbike, minivan, and bus).

Fig. 5. Bounding Boxes For Data Annotation.

IV. Design and Implementation

On the vehicle dataset, three distinct YOLOv5 architectures – small, medium, and large were implemented and trained. Table I lists the dataset specifications that have been defined. The total number of bounding box instances and their respective classes are displayed in Fig. 7.

This dataset needed to be carefully sliced for training. There was no exact way to divide the dataset, so we employed the traditional slicing technique. We separated 80% of the data for training and a small 20% for the validation process which is

required to reduce over-fitting. A random selection was made from the main dataset to create the test set which was sufficient for the calculation of the accuracy of the model.

The first dataset contains 11,808 images of different classes of vehicles, out of which 9,447 images were used for training and 2,361 images were used for validation. After splitting the dataset and annotating the images, the training set and validation set were fed into the machine learning model. The training set was used for the algorithm training process and the validation set was used for the prediction model which provided us with the different evaluation metrics and statistics.

The instances of some of the classes are higher than others and this makes the dataset non-uniform in nature. For example, the 'Car' class contains the highest number of appearances in our dataset with a total of 10,926 times whereas classes like 'garbage van' and 'Police car' appear less in the dataset.

A. Dataset Preparation

We used a custom data collection with about 21 types of automobiles in South Asian territory for our research. Most of the images as shown in Fig. 6 were gathered from real-time data acquired by users, social networking pages, blogs, and other online sources, and the improper images were filtered out of the dataset.

Filtering the data collection is intensive, and as with any model preparation, it is required to increase the amount of relevant data that our model can extract from the dataset. Because there are images containing items that aren't supposed to be there, the dataset is full of noise.

B. Dataset Pre-processing and Augmentation

When it comes to improving the model's performance, pre-processing the dataset is essential. It is a necessary step toward improving the quality of data and the amount of useful information the model can derive from it. It is also critical to generate a balanced dataset to improve the accuracy of an existing model. Before the dataset was supplied to the model for training, the images with their pixels had to be reshaped and resized. The images were resized using the numpy.reshape() method, and the pixels were replaced with pixel/255 using vector scalar division.

Fig. 8 shows how the different augmentations were carried out and as a result, multiple different versions of the image were created. Balancing this dataset was a complex task as different categories of images had massive disparities in numbers.

Fig. 6. Sample Images from the Dataset.

TABLE I. DATASET SPECIFICATIONS

Attributes	Features
Image Type	RGB
Image Extension	JPG, PNG
Image Dimension	Various

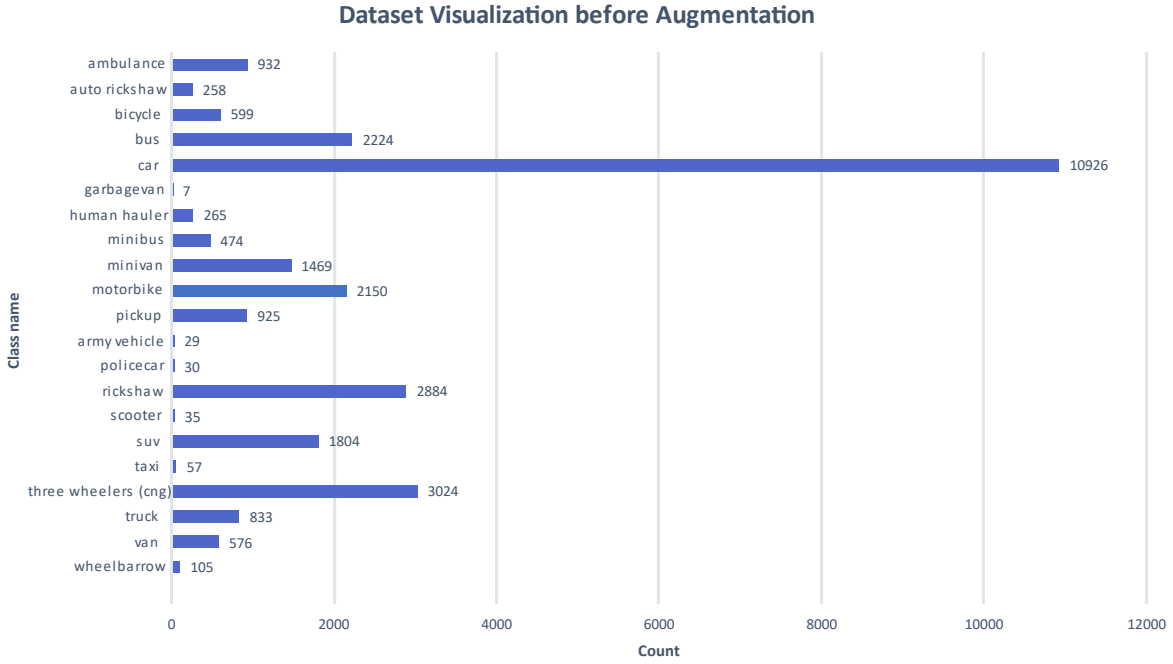

Fig. 7. Number of Bounding box Instances Per Class before Augmentation.

Fig. 8. Different Types of Image Augmentations Applied to Our Dataset.

To solve this, only two different augmentations were carried out. The Contrast, Darken and Grayscale filters were utilized on our dataset to create augmented data with distinct differences that would be easily identifiable by the machine when trained. Because our original dataset was imbalanced and non-uniform, augmentation was used to increase the number of instances of classes in the dataset, particularly those that appeared less frequently like ambulances, bicycles, etc., thus improving the model's accuracy and performance by making the dataset more balanced.

After augmentation, a relative balance within the dataset was achieved which can be seen in Fig. 9 which shows the number of bounding box instances of different classes after augmentation was applied to it compared to Fig. 7 above which was before augmentation.

C. Model Training

YOLOv5 has multiple architectures such as YOLOv5s, YOLOv5m, and YOLOv5l in order of complexity and depth. We have implemented and juxtaposed each of these models on our dataset to determine the one which fits best. Each of the models was trained for 100 epochs which took around 12 hours per model. The model training process is illustrated in Fig. 10. The network is trained using a collection of training data, and it then learns to predict the target values. We have also improved the accuracy of the dataset for the YOLOv5l, YOLOv5m, and YOLOv5s designs.

An appropriate dataset is required to train a deep learning network. A train-test split needs to be made depending on the available data. During the training phase, validation losses are tracked, and non-constant values are generated after several epochs. Otherwise, the model will be adjusted for hyper-parameters, and the validation loss value will be kept as low as possible.

The model with the largest validation loss is saved for testing on the real data. When a model obtains high precision and recall rates for new datasets, or when it demonstrates improved performance after training on an enriched dataset, it is said to perform satisfactorily.

D. Dataset Demonstration

For model training, the entire vehicle dataset is provided. Table II shows the number of images evaluated for implementation, the number of images used to train the model, and the number of images used in testing.

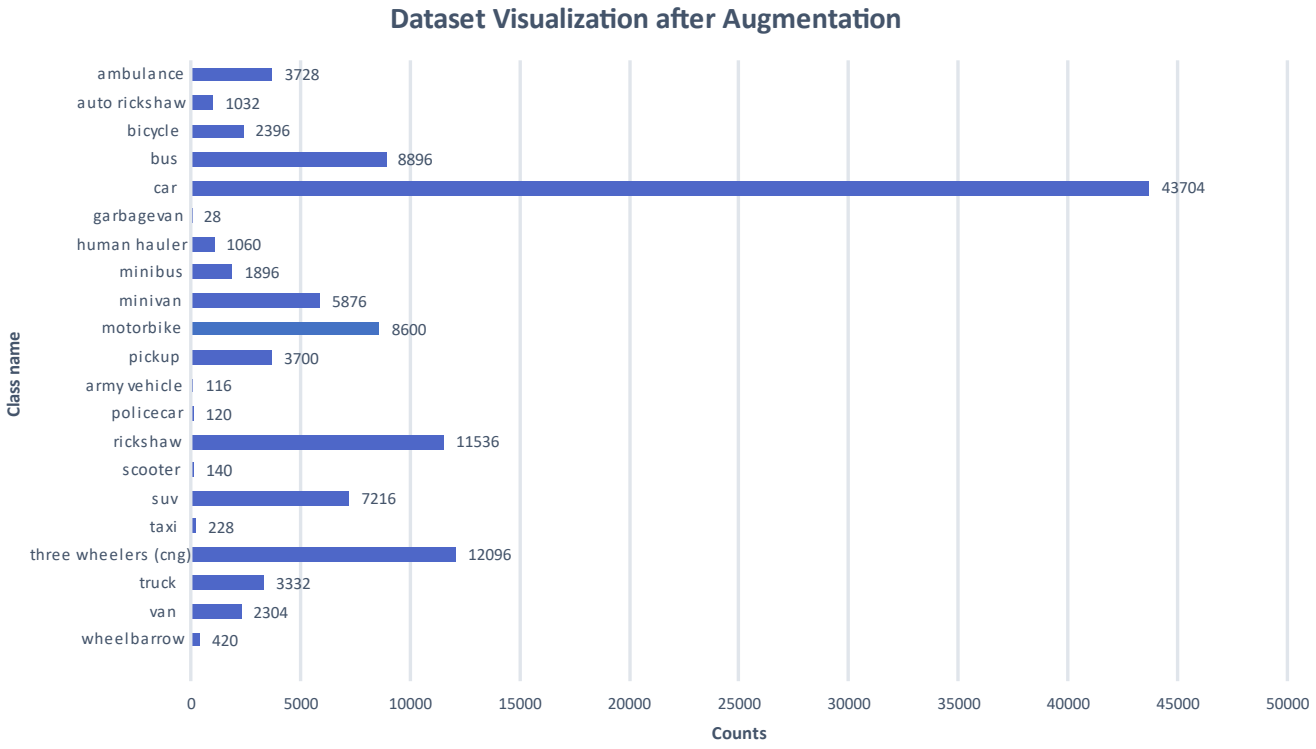

Fig. 9. Number of Bounding Box Instances Per Class after Augmentation.

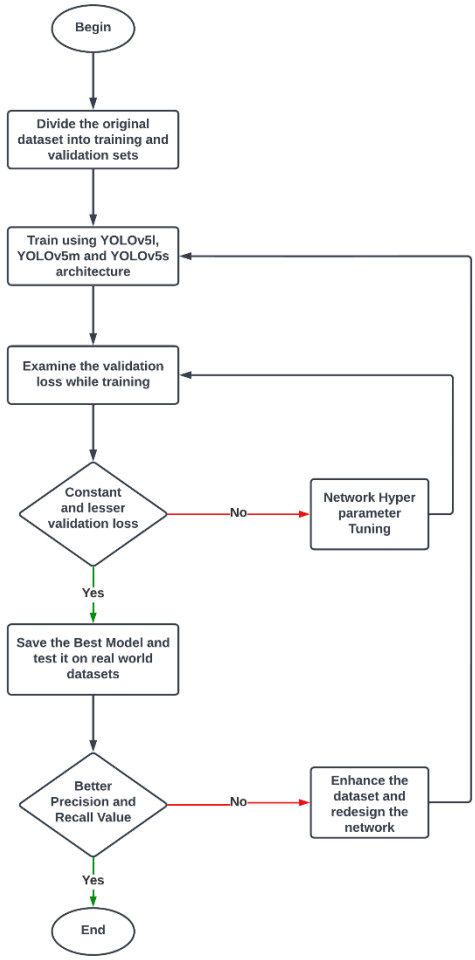

Fig. 10. Flow Diagram of Model Training.

TABLE II. DATASET DISTRIBUTION

Features	Before Augmentation	After Augmentation
Number of Classifiers	21	21
Total Input Images	11,808	47,232
Images to be Trained	9,447	37,786
Images to be Tested	2,361	9,446

V. DISCUSSION

We were able to create a big dataset with over 21 unique classes and about 11,808 images. The dataset size increased to exactly 47,232 images after the various Data Augmentation methods were applied.

A. Performance Matrices

Confusion Matrix: Assists us in this by providing a comprehensive assessment of each model's performance, including faults.

- TP (True Positive) - When the anticipated value is equal to the actual value and the result is positive.
- TN (True Negative) - When the projected value is the same as the actual value, but the value is negative.
- FP (False Positive) - The anticipated value was incorrectly predicted as positive when the actual value was negative.
- FN (False Negative) - The anticipated value was incorrectly predicted as negative when the actual value was positive.

A variety of performance measures can be calculated using these numbers.

Accuracy: Measurement as a singular metric is invalid because it assigns equal costs to different types of errors and can only be used with a well-balanced dataset. The formula is as follows:

$$\text{Accuracy} = \frac{TP+TN}{TP+FP+TN+FN} \quad (1)$$

Precision: Precision is a metric for how many accurately anticipated situations turn out to be positive, which can help establish a model's eligibility.

The formula for precision is as follows:

$$\text{Precision} = \frac{TP}{TP+FP} \quad (2)$$

Recall: Recall is the measure of the positive cases that were correctly classified by our model and is defined by the following formula:

$$\text{Recall} = \frac{TP}{TP+FN} \quad (3)$$

mAP (mean Average Precision): mAP is used to evaluate object detection models like Fast R-CNN, YOLO, and Mask R-CNN. It considers both types of errors, false positives (FP) and false negatives (FN), as well as the trade-off between precision and recall (FN).

$$mAP = \frac{1}{N} \sum_{i=1}^{N} AP_i \quad (4)$$

The mAP is calculated by averaging each class's Average Precision (AP) over several classes.

F1 Score: It combines the values of Precision and Recall into a single metric that must be maximized to enhance our model. However, interpreting the F1 score is challenging, leaving us oblivious to which of the metrics the model is optimizing. The formula is as follows:

$$\text{F1 Score} = \frac{2}{\frac{1}{\text{Recall}} + \frac{1}{\text{Precision}}} \quad (5)$$

Loss function: We will summarize YOLOv5 losses and metrics to help you better comprehend the outcomes. The three parts of the YOLO loss function are box loss, obj loss, and class loss.

VI. RESULT ANALYSIS

For this research, three distinct YOLOv5 designs – small, medium, and large – were implemented and used. Before training, the dataset was pre-processed and supplemented. The complexity and depth of the three YOLOv5 architectures differ. Each model was tested against our dataset, and the results were compared to determine which model performed the best in terms of vehicle detection.

Mean average precision is a well-known and commonly used object detection evaluation metric. Faster R-CNN [35], YOLO [36], and MobileNet [37] are all state-of-the-art models that use mAP to evaluate their models. We tried to test the performance of the three YOLOv5 models – small, medium, and large – in our implementation.

A. Mean Average Precision (mAP) Analysis

The mAP 0.5 of the three designs employed throughout 100 epochs is illustrated in Fig. 11 and Fig. 12. The YOLOv5l model achieves more accuracy in Fig. 11 and Fig. 12 than the other two models both before and after augmentation, as shown in the graphs.

B. Training Loss Analysis

The training loss is a metric that measures how well a deep learning model matches the training data. That is, it evaluates the model's error on the training data. The training set is a subset of the dataset that was used to train the model originally. The training loss is calculated computationally by adding the sum of errors for each sample in the training set. It is also worth noting that the training loss is calculated after every batch.

According to Fig. 13 and Fig. 14, it showed that the training loss is minimal for the YOLOv5l model. The lesser the training loss, the faster a model's performance can be obtained.

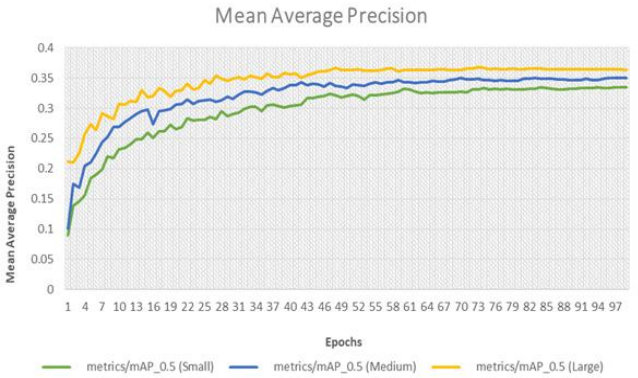

Fig. 11. mAP Comparison between Models before Augmentation.

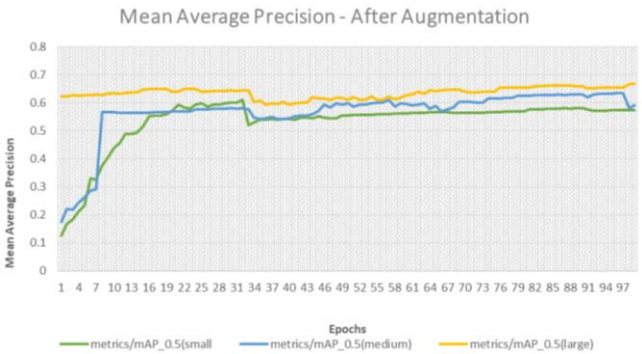

Fig. 12. mAP Comparison between Models after Augmentation.

Fig. 13. Train/Box Loss Comparison between Models before Augmentation.

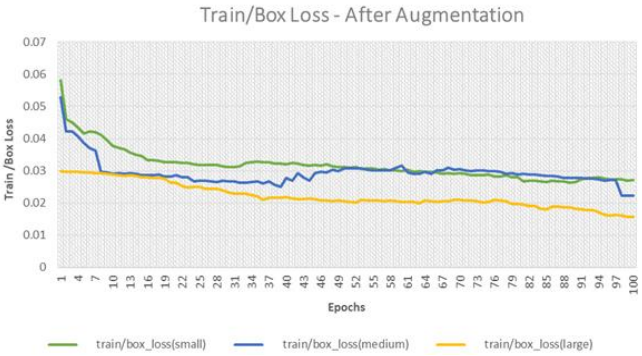

Fig. 14. Train/Box Loss Comparison between Models after Augmentation.

C. F1 Score Analysis

By comparing the F1 Score among the three models before and after the augmentation of the dataset as seen in Fig. 15 and Fig. 16, we can see that the large model has a higher score, followed by the medium model and the small model.

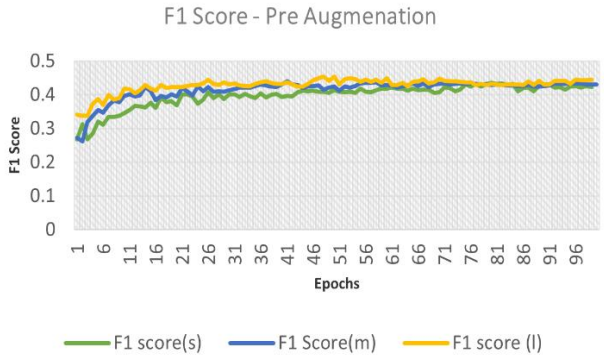

Fig. 15. F1 Score Comparison between Models before Augmentation.

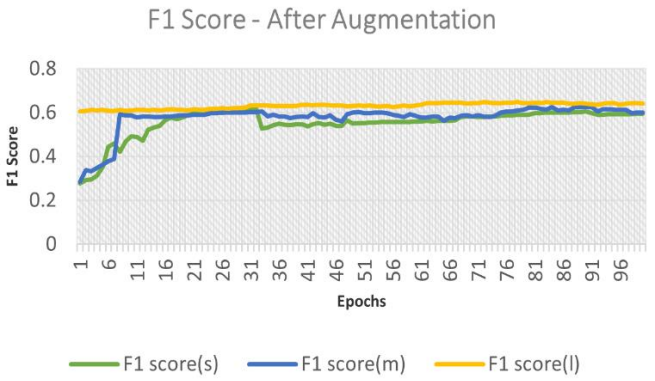

Fig. 16. F1 Score Comparison between Models after Augmentation.

D. Augmented Vehicle Dataset Epoch Results

We have augmented our dataset and run for 100 epochs. At the end of the last epoch, the precision was 0.64, recall was 0.63 and mAP was 0.66 for an augmented dataset in YOLOv5 three architecture, Table III.

TABLE III. PERFORMANCE MEASUREMENT AND COMPARISON OF DIFFERENT YOLO MODELS

Model	Precision	Recall	mAP_0.5
YOLOv5l	0.64214	0.63763	0.66877
YOLOv5m	0.59261	0.60812	0.60112
YOLOv5s	0.58038	0.61149	0.57469

The performance is not up to the mark as our dataset was unbalanced and we have a higher amount of car dataset by comparing with other types of vehicles.

E. YOLOv5 Overall Performance Analysis

For analyzing the performance of YOLOV5 based on three different architectures, we have measured the performance based on before and after augmentation data. The outcome is presented in Tables IV and V.

Based on the findings, we can conclude that the YOLOv5l model does better out of the three architectures when it comes to vehicle detection for our dataset.

TABLE IV. RESULT COMPARISON OF YOLOV5S, YOLOV5M AND YOLOV5L BEFORE AUGMENTATION

Attributes	YOLOv5s	YOLOv5m	YOLOv5l
mAP_0.5	0.33505	0.35135	0.36361
mAP_0.5:0.95	0.20561	0.23274	0.24556
train/box_loss	0.03026	0.02182	0.01924
train/class_loss	0.01448	0.00737	0.00503
validation/box_loss	0.03443	0.03407	0.03309
validation/class_loss	0.02466	0.02801	0.02853
F1 Score	0.42211	0.43096	0.4438

TABLE V. RESULT COMPARISON OF YOLOV5S, YOLOV5M AND YOLOV5L AFTER AUGMENTATION

Attributes	YOLOv5s	YOLOv5m	YOLOv5l
mAP_0.5	0.57469	0.60112	0.66877
mAP_0.5:0.95	0.41897	0.43868	0.48071
train/box_loss	0.02711	0.0225	0.01574
train/class_loss	0.00977	0.013675	0.00158
validation/box_loss	0.02235	0.022442	0.02112
validation/class_loss	0.00952	0.009342	0.00802
F1 Score	0.59552	0.600264	0.63987

VII. CONCLUSION

The application of the different modules of YOLOv5 has significantly improved the detection of vehicular objects in traffic and on the road. The accumulated dataset has provided a vast amount of variety in vehicle classes which led to a richer and more accurate result across all the models. For performance comparison, we utilized different models with different nodes, layers, and speeds. After an extensive data training and processing, period was carried out, it was seen that YOLOv5l had outperformed the rest. Its deeper node complexity and increased number of convolutional layers make it the most efficient in terms of processing, but it was also seen that it takes the most time although by a short margin. The significant performance efficiency of the YOLOv5l model compared to YOLOv5s and YOLOv5m solidifies the real-world application opportunities for the detection of uniquely South Asian vehicles.

REFERENCES

[1] Bangladesh Road and Transport Authority. Vehicles registered in Bangladesh. 2021. http://www.brta.gov.bd/site/page/74b2a5c3-60cb-4d3c-a699-e2988fed84b2.

[2] Paspelava, Computer Vision Object Detection: Challenges faced. 2021.

[3] C. Snyder, D. Gonzales, M.Do, and T. Ma, “Congestion estimation using traffic cameras,” 2019.

[4] P. Shinde, S. Yadav, S. Rudrake, and P. Kumbhar, “Smart traffic control system using YOLO,” Int. Res. J. Eng. Technol, vol. 6, issue. 12, pp. 169-172, 2019.

[5] Y. Li, and J. Ibanez-Guzman, “Lidar for autonomous driving: The principles, challenges, and trends for automotive lidar and perception systems,” IEEE Signal Processing Magazine, vol. 37, issue. 4, pp. 50-61, 2020.

[6] Ultralytics. YOLOv5: A family of Object Detection Architectures and Models.

[7] C. Supeshala, Yolo v4 or Yolo V5 or PP-Yolo? Medium. 2020.

[8] Vehicle detection at intersections by Lidar System. Smart Sensing for Traffic Monitoring, 81–96, 2020.

[9] J. Zhang, “MASFF: Multiscale Adaptive Spatial Feature Fusion Method for vehicle recognition,” Journal of Computers, 33(1), 001–011, 2022.

[10] K. Kowol, M. Rottmann, S. Bracke, and H. Gottschalk, “YOdar: Uncertainty-based sensor fusion for vehicle detection with camera and radar sensors,” Proceedings of the 13th International Conference on Agents and Artificial Intelligence. 2021.

[11] F. T. Barwell, Vehicle detection. Automation and Control in Transport, 76–83. 1973.

[12] J. Ciberlin, R. Grbic, N. Teslić, and M. Pilipović, "Object detection and object tracking in front of the vehicle using front view camera," 2019 Zooming Innovation in Consumer Technologies Conference (ZINC), 2019.

[13] C. Hisham, "Classifier-based approaches for top-down salient object detection," Doctoral thesis, Nanyang Technological University, Singapore, 2017.

[14] A. Aggarwal, Yolo explained. What is YOLO and How does it work? Medium. 2020.

[15] L. Tan, T. Huangfu, L. Wu, and W. Chen, "Comparison of yolo v3, faster R-CNN, and SSD for real-time pill identification," Research Square, 2021.

[16] J. Liu, and D. Zhang, "Research on vehicle object detection algorithm based on improved yolov3 algorithm," Journal of Physics: Conference Series, 2020.

[17] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016.

[18] G. Chandan, A. Jain, H. Jain, and Mohana, "Real time object detection and tracking using deep learning and openCV," International Conference on Inventive Research in Computing Applications (ICIRCA), 2018.

[19] Z. Chen et al., "Real time object detection, tracking, and distance and motion estimation based on deep learning: Application to smart mobility," Eighth International Conference on Emerging Security Technologies (EST), 2019.

[20] J. Kim, S. J. Sung, and S. Park, "Comparison of faster-RCNN, Yolo, and SSD for real-time vehicle type recognition," ICCE-Asia, 2020.

[21] D. J. Phillips et al., "Real-time prediction of automotive collision risk from monocular video," ArXiv, abs/1902.01293, 2019.

[22] X. Wang et al., "Vision-based detection and tracking of a mobile ground target using a fixed-wing UAV," International Journal of Advanced Robotic Systems, volume 11, issue 9, page 156, 2014.

[23] J. Sokalski, T. Breckon, and I. Cowling, "Automatic salient object detection in uav imagery," Proc. of the 25th Int. Unmanned Air Vehicle Systems, pages 1-12, 2010.

[24] K. Kanistras, G. Martins, M. Rutherford, and K. Valavanis, "Survey of unmanned aerial vehicles (UAVs) for traffic monitoring." Handbook of unmanned aerial vehicles, pages 2643-2666, 2014.

[25] B. Xiao, and S. Kang, "Development of an image dataset of construction machines for deep learning object detection," Journal of Computing in Civil Engineering, 35(2), 2021.

[26] B. Zoph et al., "Learning data augmentation strategies for object detection," Computer Vision – ECCV 2020, 566–583. 2020.

[27] J. Lin, and M. Sun, "A YOLO-based Traffic Counting System," 2018 Conference on Technologies and Applications of Artificial Intelligence (TAAI), 2018.

[28] J. Tao, H. Wang, X. Zhang, X. Li, and H. Yang, "An object detection system based on YOLO in the traffic scene," The 6th International Conference on Computer Science and Network Technology (ICCSNT), pp. 315-319, 2017.

[29] A. Corovic, V. Ilic, S. Duric, M. Marijan, and B. Pavkovic, "The real-time detection of traffic participants using YOLO algorithm," 2018 26th Telecommunications Forum (TELFOR), pp.1-4, 2018.

[30] A. Salarpour, A. Salarpour, M. Fathi, and M. Dezfoulian, "Vehicle tracking using Kalman filter and features," Signal & Image Processing 2, no. 2, 2011.

[31] H. N. Phan, L.H Pham, D. N. Tran, and S. V. Ha, "Occlusion vehicle detection algorithm in crowded scene for traffic surveillance system," In 2017 International Conference on System Science and Engineering (ICSSE), pp. 215-220. IEEE, 2017.

[32] L. Ding, Y. Wang, R. Laganière, X. Luo, and S. Fu, "Scale-aware RPN for vehicle detection," In International Symposium on Visual Computing, pp. 487-499. Springer, Cham, 2018.

[33] U. Nepal, and H. Eslamiat, "Comparing YOLOv3, YOLOv4 and YOLOv5 for autonomous landing spot detection in faulty UAVs," Sensors 22, no. 2, 464, 2022.

[34] K. Wang, et al., "Panet: Few-shot image semantic segmentation with prototype alignment," In Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 9197-9206. 2019.

[35] A. Syaharuddin, Z. Zainuddin, and Andani, "Multi-pole road sign detection based on faster region-based convolutional neural network (Faster R-CNN)," 2021 International Conference on Artificial Intelligence and Mechatronics Systems (AIMS), 2021.

[36] L. Jiang, H. Liu, H. Zhu, and G. Zhang, "Improved YOLO v5 with balanced feature pyramid and attention module for traffic sign detection," In MATEC Web of Conferences, vol. 355. EDP Sciences, 2022.

[37] D. Sinha, and M. El-Sharkawy, "Thin mobilenet: An enhanced mobilenet architecture," In 2019 IEEE 10th annual ubiquitous computing, electronics & mobile communication conference (UEMCON), pp. 0280-0285. IEEE, 2019.