Obstacle Pattern Recognition From Vehicle Perspective

Md. Imamul Mursalin sujoy[1], **Md. Asif Rahman**[2], **Jannatul Ferdoush**[3] **and Utsha Sen Dhruba**[4]

†*Department of Computer Science and Engineering, BRAC University*

May 02, 2024

Abstract

The rapid evolution of autonomous vehicle technology has spurred intensive research into computer vision algorithms capable of robust and efficient object detection. Among these methods, You Only Look Once (YOLO) has emerged as a groundbreaking approach, offering real-time performance and high accuracy. In this paper, we investigate the application of YOLO in the context of object detection for autonomous vehicle, focusing on its architecture, training process, and performance characteristics. We provide an overview of related work in object detection for autonomous vehicles and delve into the intricacies of the YOLO architecture, highlighting its single-pass design and feature extraction methodology. Additionally, we discuss the training process of YOLO, covering data preparation, loss functions, and optimization techniques. Through a series of evaluations, we assess the performance of YOLO in various scenarios relevant to autonomous driving, identifying its strengths and limitations. Our findings underscore the potential of YOLO as a key component of object detection for autonomous vehicles, contributing to the advancement of intelligent transportation systems and the realization of safer and more efficient object detection system.

Keywords: YOLO, convolution network, RCNN, localization, SSD

Corresponding author: Mr. Annajiat Alim Rasel *E-mail address:* annajiat@bracu.ac.bd

INTRODUCTION

Autonomous vehicles have sparked public interest in a world where technology is developing at a never-before-seen rate. They have the potential to completely transform the way we move and engage with our urban surroundings. In the current scenario for autonomous vehicles, identifying obstacles and being able to maneuver through complex surroundings with accuracy are critical to their success. The purpose of object detection for an autonomous vehicle is to identify and categorize the locations of 3D objects in a dynamic environment.

Accordingly, considerable concern has been given by researchers to this area in the past few years. Object detection algorithms using region proposal includes RCNN, Fast RCNN [1], and Faster RCNN models create region proposal networks (RPN) and then the region proposals are divided into categories afterward. On the other hand, object detection algorithms using regression includes SSD and YOLO which also generate region proposal networks (RPN) but divide these region proposals into categories at the moment of generation [2]. All of the procedures mentioned above have significant accomplishments in object localization and recognition. Among these object detection methods, YOLO (You Only Look Once) stands out as a pioneering approach, offering real-time performance without compromising on accuracy. YOLO revolutionized object detection by introducing a unified framework that simultaneously predicts bounding boxes and class probabilities for multiple objects within an image.

BACKGROUND

Region based convolutional neural networks (RCNN) algorithm uses a group of boxes for the picture and then analyses in each box if either of the boxes holds a target. It employs the method of selective search to pick those sections from the picture. In an object, the four regions are used. These are varying scales, colours, textures, and enclosure. Drawbacks of RCNN method- Based on a selective search, 2,000 sections are excerpted per image. For every region or part of the image, we have to select features using CNN. For this, if we have 'i' number of images, then selected regions will become i×2,000. The whole method of target identification through RCNN utilizes the following three models: Linear SVM classifier for the identification of objects, CNN is employed for characteristic extraction, and a regression model is required to tighten the bounding boxes. All these three processes combine to take a considerable amount of time. It increases the running time of RCNN method. Therefore, RCNN needs almost 40 to 50 seconds to predict the result for several new images [7].

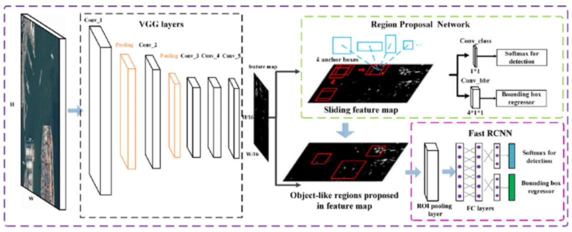

Figure 1. Structure of faster RCNN

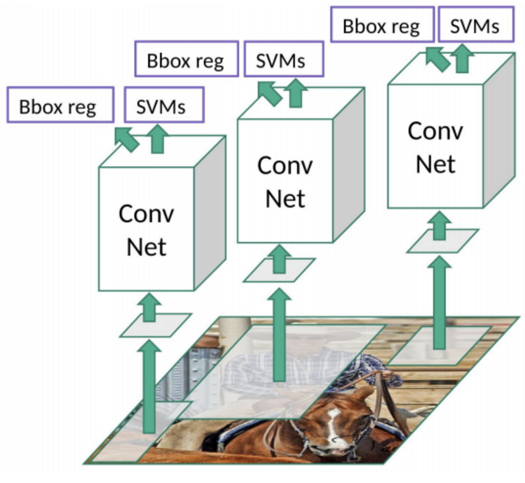

Figure 2. RCNN layers

In place of using three different models of RCNN, Fast RCNN employs one model to excerpt characteristics from the different regions. Then it distributes the regions into several categories based on excerpted features, and the boundary boxes of recognized divisions return together. Fast RCNN uses the method of spatial pyramid pooling [11] to calculate only one CNN representation for the whole image. It passes one region for each picture to a particular convolutional network model by replacing three distinct models for excerption of characteristics, distributing into divisions, and producing bounding boxes. Drawbacks of Fast RCNN method- Fast RCNN also employ a selective search method to detect concerned regions. This method is prolonged and demands a lot of time. Usually, for the detection of objects, this complete procedure needs almost two seconds for each picture. Therefore its speed is quite good in contrast to RCNN. However, if we contemplate extensive real-life datasets, then the execution of fast RCNN approach is still lacked in speed.

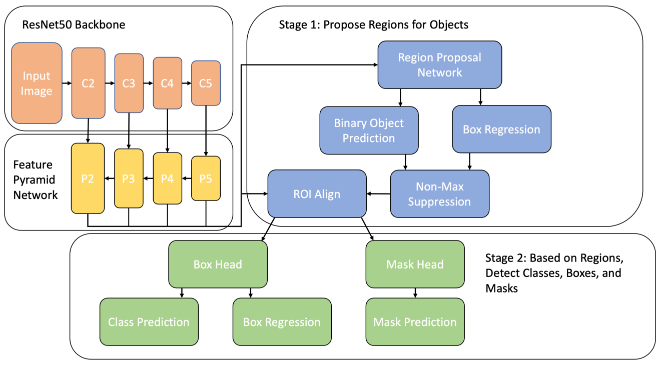

Figure 3. Resnet50 process

Faster RCNN is a transformed variant of fast RCNN. The significant difference between both is that faster RCNN implements region proposal network (RPN) but fast RCNN utilizes a selective search technique for producing concerned regions. In input, RPN accepts feature maps of pictures and produces a collection of object recommendationsand an objectness score per recommendation in output. Usually, this approach takes ten times less time in contrast to fast RCNN approach because of RPN. Drawbacks of faster RCNN method- To excerpt all the targets in a given picture, this procedure needs multiple passesfor that particular picture [4]. Different systems are working in a sequence therefore, the performance of the upcoming operation is based on the performance of preceding operations. This approach uses region proposal networks to localize and identify the objects in a picture. But RPNs do not contemplate the complete picture because it uses only those portions of the picture, which have high probabilities of the presence of targets.

SSD is a single shot detector. It manages an excellent balance of speed with the accuracy of result. In this, we apply for single time a CNN based model to the input picture for computing the feature map. It also employs anchor boxes similar to faster RCNN at various aspect ratios and learns the offset instead of determining the box. The processing occurs on numerous layers of CNN, where every layer functions on a varying range of scale and uses multiple feature maps. Therefore, the detection of targets of several sizes is possible. Experimentally, SSD has much better accuracy on different datasets even on inputs pictures of small size as compared to the other single stage methods. Unlike YOLO, SSD does not divide the image into grids of random size. For every location of the feature map, it predicts the offset of predefined anchor boxes (default boxes) [7]. Relative to the corresponding cell, each box has a fixed size, proportion, and position. In a convolutional manner, all the anchor boxes cover the entire feature map. Anchors of SSD are slightly different from the anchors of YOLO. Because YOLO makes all the predictions from a single grid, and the size of anchors used by YOLO ranges from dimensions of one grid cell to the dimensions of the entire picture. The anchors of SSD specialize its detector for distinct feasible viewpoints and dimensional ratios of its target shapes, but not enough on the size of targets. The calculation for the anchors of SSD uses a simple formula, while the anchors of YOLO are calculated by applying k-means clustering on the training data [12]. SSD doesn't use confidence score, but YOLO calculates it to show the faith in predicted results. A unique background class is employed by SSD for this work. A low value of confidence score in YOLO is equivalent to the predicted output of background class in SSD [3]. Both indicate that for the detector, the possibility of getting a target is null.

YOLO is a cutting-edge method for object recognition that is well-known for its rapidity and precision. Created by Ultralytics, it is the fifth installment of the YOLO series and brings substantial improvements compared to its earlier versions. By partitioning the input picture into a grid and immediately predicting bounding boxes and class probabilities, YOLO's architecture follows the one-stage object identification technique. To enhance data transfer between nodes, it makes use of the innovative CSP (Cross-Stage Partial) module on top of the CSPDarknet53 backbone, which is an improved variant of the Darknet design. To improve the model's object identification capabilities, the detection head uses PANet modules for efficient feature aggregation across scales. Simplicity, efficiency, and accessibility are the emphasized features of YOLOv5, in contrast to YOLOv4. Its simplified form makes it easy to grasp and put into practice With its improved training and inference speeds, YOLO is perfect for real-world applications that don't have a lot of processing power. In spite of using decreased model sizes, YOLO still manages to perform competitively. To train YOLO, data is heavily enhanced via random scaling, translation, rotation, and flipping, which strengthens the model. It is usual practice to use transfer learning, which involves fine tuning the model for a particular detection job using pre-trained weights on a big dataset. To meet the needs of a wide range of users, YOLO provides a number of model versions with varying degrees of complexity and accuracy. The YOLO algorithm has several potential uses in various fields, such as object recognition in visual media, surveillance, autonomous cars, and situations requiring detection in real-time. It is widely used for computer vision jobs that need object identification skills in real-time because of its efficiency, accuracy, and user-friendliness. To sum up, YOLOv5 is a huge step forward for the YOLO series; it combines speed, precision, and simplicity to effectively identify objects in all sorts of scenarios.

METHOD

You Only Look Once algorithm utilizes features learned by a deep convolution neural network (CNN) to detect objects. It is a fully convolutional network (FCN), thus making it invariant to the size of the input image. The input image is firstly resized to the network resolution. It is then divided into S x S grid cells, and each of these grid cells is "responsible" for predicting objects whose center falls within it. In practice, a grid cell might detect an object even though the center of the object does not fall within it. This leads to multiple detections of the same object by different grid cells. Non-max suppression cleans up the detections and ensures that each object is detected once. This is done by selecting the bounding box with the highest object detection probability as the output bounding box and suppressing bounding boxes that have a high IoU with the output bounding box. In addition, predefined shapes called anchor boxes enable the detection of multiple objects whose centers fall within the same grid cell. Each object is associated with the anchor box with the highest IoU [8]. The K-means clustering algorithm isused to determine the height and width of the anchor boxes. Each bounding box prediction is a vector. The components of the vectors are the following: confidence score of object detection, x,y coordinates of the center of the bounding box,the height and width h,w of the bounding box and C class probabilities. If there are A anchor boxes, the vector is A(5+C) in dimension.

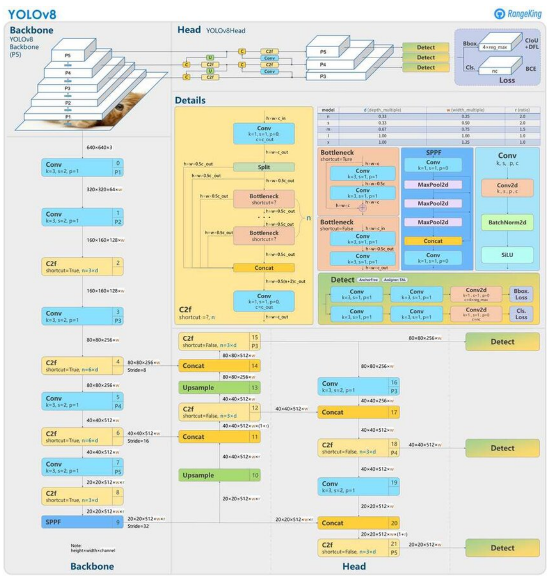

Figure 4. Architechture of YOLOv8

DATASET

The nuScenes dataset is a publicly available multimodal dataset by nuTonomy. The data was gathered in Boston and Singapore; two mega cities with busy traffic, thus ensuring a diverse scenario of traffic situations. The dataset comprises of 1600 x 900, images with 3D annotations of 23 classes. Objects were annotated by considering the full suite of sensors, 6 cameras, 1 Lidar and 5 Radar. Each annotated object was covered by at least one lidar or radar point; hence even objects with low visibility: 0 percent to 40 percent visibility were annotated. The annotations were done by expert annotators and numerous validation steps were performed to ensure the quality of the annotations. The diversity and quality of the annotations is why nuScenes was selected for our project. For the purposes of 2D object detection, we converted the given 3D bounding boxes into 2D bounding boxes. The global coordinates of the 8 corners of the 3D bounding boxes were provided and were converted into camera coordinates via the get sample data function provided by nuTonomy. The given functions can be accessed at www.nuscenes.org. We wrote our own function, all 3d to 2d to convert the camera coordinates to the image coordinates by utilizing the intrinsic camera calibration matrices. The 2D bounding boxes were then extracted by taking the minimum and maximum of the x and y coordinates of the 3D bounding boxes via our extract bounding box function. These coordinates form the corners of our resulting 2D bounding boxes.

Further, we have used another dataset from Roboflow. This dataset, meticulously curated for autonomous vehicle navigation and obstacle detection, comprises 9183 images. Seamlessly partitioned, 85% of the dataset fuels the training process, while 10% is dedicated to validation and a further 5% serves as testing grounds. Spanning diverse scenarios, this dataset encapsulates 10 distinct classes vital for comprehensive training. Originating from the esteemed Roboflow Universe, renowned for its quality datasets, this resource promises to empower cutting-edge advancements in autonomous vehicle technology.

We only acquired the 2D bounding boxes for objects whose visibility exceeded 40 percent and whose center fall within the image boundaries. This is to ensure that the extracted bounding box annotations were similar to that of data only acquired only via cameras. We also combined the 'adult', 'child', 'police officer' and 'construction worker' classes together to form our pedestrian class. We generated the train dataset, validation dataset and test dataset by randomly splitting the nuScenes dataset into 70 percent for training, 15 percent for validation and 15 percent for testing.

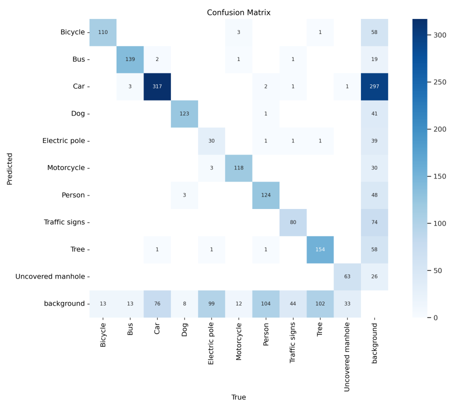

Figure 5. Confusion matrix

We trained the model on the train dataset. The base model and the initial pre-trained weights were acquired via the official YOLO website. We trained 4 different versions of the base model by tuning the following hyperparameters: resolution and subdivision. In addition, we changed the default anchor box values to that generated by the K-means clustering algorithm. The resolution and subdivision of the base model are 416 x 416 and 8 respectively. The model resizes any input data to the resolution value. The subdivision value refers to number of mini-batches that is sent to the GPU for processing.

RESULT ANALYSIS

The 4 different versions of the YOLO model were trained by tuning the following hyperparameters: resolution and subdivision. The trained models were then validated using the validation dataset. The mean average precision value is the area under the precision and recall curve. It is a metric that is used to compare the performance of various models. The model with the input resolution of 832 and subdivision value of 8 was selected as the best performing model as it has the highest mAP score of 61.76% at the IoU threshold of 50%. The loss during the training of the model with resolution 832 and subdivision 8, declined rapidly before stagnating at 1.2.

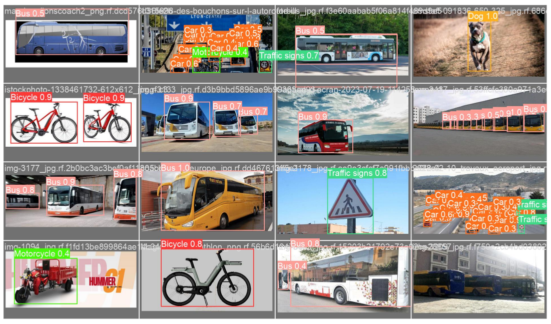
Figure 6. Detection outcome

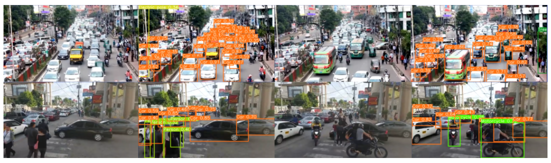
Figure 7. Raw images vs after object detecting images

Further training will probably not improve the model's performance. The mAP value reached a maximum of 61.76% at iteration 1000. Hence the weights from this iteration was used as our final weights. The mAP score declined after iteration 1000. The decline could be due to overfitting. This could be verified by training the model for several more iterations and determining if the declining mAP trend continues. This model had the highest mAP score out of all trained models; and was thus chosen as the model for further analysis. The default confidence score threshold of YOLO during detection is 25%. At this threshold, the precision, recall and F1-scores are 0.81, 0.57 and 0.67 respectively. The high precision of 0.81, indicates low false positives and the low recall value of 0.57 indicates high false negatives.

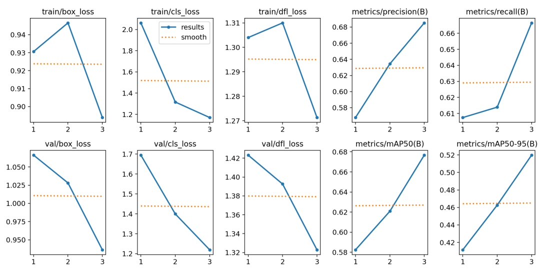

Figure 8. Matrics

We tested our selected model on the test dataset. A mAP score of 63.39% at IoU threshold of 50% was achieved. At confidence threshold of 10%, the precision, recall and F1-score are 0.67, 0.69 and 0.68 respectively. The average precision values for pedestrians, cars and cyclists are 55.41%, 76.72% and 58.04% respectively. The mAP scores during validation and testing are 61.76% and 63.39% respectively The difference is 1.63%. The difference in mAP score is insignificant. Thus, we conclude that our model generalizes well and was not overfitted.

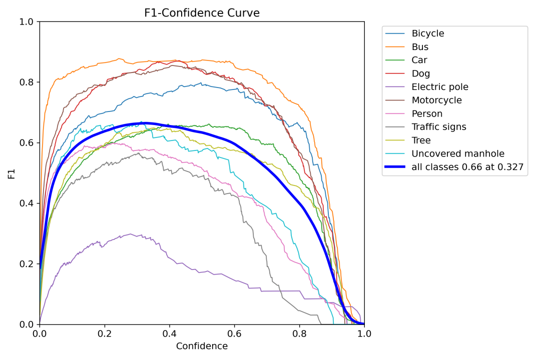

Figure 9. F1 confidence curve

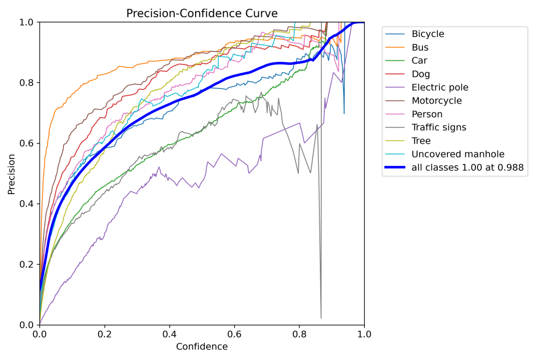

Figure 10. Precision recall curve

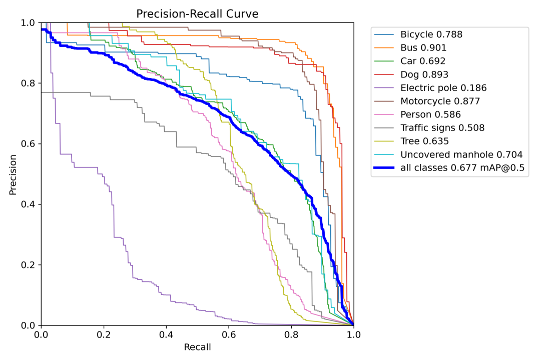

Figure 11. Precision confidence curve

CONCLUSION

To identify and localize objects, there exist many methods with a trade-off in speed performance and accuracy of result. But yet we can't say any single algorithm is best over others. One can always select the method that suits the requirement at best. In a short period, object detection applications got much popularity and still a lot to cover in this area because of its vast scope of research. Finally, object tracking could also be a extension of our work. It enables learning about the behaviour of agents surrounding the autonomous vehicle, thus decisions based on predictive behaviour can be made [10]. For instance, if a pedestrian is predicted to be a high-risk jaywalker, the autonomous vehicle should take that into account and drive more conservatively. Therefore, in future we would extend this project for different datasets and thus improve the accuracy and efficiency of this model.

REFERENCES

[1] Häne, C., Heng, L., Lee, G. H., Fraundorfer, F., Furgale, P., Sattler, T., Pollefeys, M. (2017). 3D visual perception for self-driving cars using a multi-camera system: Calibration, mapping, localization, and obstacle detection. Image and Vision Computing, 68, 14–27. https://doi.org/10.1016/j.imavis.2017.07.003

[2] Zaarane, A., Slimani, I., Okaishi, W. A., Atouf, I., Hamdoun, A. (2020). Distance measurement system for autonomous vehicles using stereo camera. Array, 5, 100016. https://doi.org/10.1016/j.array.2020.100016

[3] F. Naser, I. Gilitschenski, G. Rosman, A. Amini, F. Durand, A. Tor-ralba, G. W. Wornell, W. T. Freeman, S. Karaman, and D. Rus.Shadowcam: Real-time detection of moving obstacles behind A corner for autonomous vehicles. In Proceedings of International Conference on Intelligent Transportation Systems (ITSC), 2018.

[4]Howard, Ian (2012). Perceiving in Depth. New York: Oxford University Press. ISBN 978-0-199-76414-3. Sternberg, R. K. (2012). Goldstein E.B. (2014, 2017) Sensation and perception (10th ed.). Pacific Grove CA: Wadsworth.

[5]Smolyanskiy, N. (2018). On the Importance of Stereo for Accurate Depth Estimation: An Efficient Semi-Supervised Deep Neural Network Approach.

[6]Yasir Dawood Salman, Ku Ruhana Ku-Mahamud, Eiji Kamioka. (2017). Distance measurement for self-driving cars using stereo camera in Zulikha, J. N. H. Zakaria (Eds.), Proceedings of the 6th International Conference of Computing Informatics (pp 235-242). Sintok: School of Computing.

[7]Kadambi, A., Bhandari, A., Raskar, R. (2014). 3D Depth Cameras in Vision: Benefits and Limitations of the hardware. In Springer eBooks (pp. 3–26). https://doi.org/10.1007/978-3-319-08651-41

[8]Xu, J., Liu, X., Bai, Y., Jiang, J., Wang, K., Chen, X., Ji, X. (2022). MultiCamera collaborative depth prediction via consistent structure estimation. Proceedings of the 30th ACM International Conference on Multimedia. https://doi.org/10.1145/3503161.3548394

[9]Wofk, D., Ma, F., Yang, T., Karaman, S., Sze, V. (2019). Fast-Depth: Fast Monocular Depth Estimation on Embedded Systems. IEEE. https://doi.org/10.1109/icra.2019.8794182

[10]Khan, F. N., Salahuddin, S., Javidnia, H. (2020). Deep Learning-Based Monocular Depth Estimation Methods—A State-of-the-Art Review. Sensors, 20(8), 2272. https://doi.org/10.3390/s20082272

[11]Birkl, R. (2023, July 26). MiDaS v3.1 – A Model Zoo for Robust Monocular Relative Depth Estimation. arXiv.org. https://arxiv.org/abs/2307.14460

[12] Seo, B., Park, B., Choi, H. (2022). Sensing range extension for Short-Baseline Stereo camera using monocular depth estimation. Sensors, 22(12), 4605. https://doi.org/10.3390/s22124605