Journal of **N**onlinear **A**nalysis and **O**ptimization
Vol. 13, Issue. 2 : 2022
ISSN : **1906-9685**

INCORPORATING ARTIFICIAL INTELLIGENCE IN TRAFFIC MONITORING AS A SUCCESSFUL AND EFFICIENT VEHICLE DETECTION SYSTEM

[1]P.Jagadeshwar, [2]Ramavath Vinod Kumar, [3]Pendem Bhanuprasad, [4]Villa Ramalxmi
[1,2,3]Assistant Professor, [4]Student, [1,2,3,4]Department of Computer Science Engineering , Siddhartha Institute of Engineering and Technology, Hyderabad, India.

ABSTRACT
Traffic congestion is a serious issue in developing nations. Light configurations are automatically modified by smart traffic light systems in response to traffic circumstances in real-time. For the system to adjust correctly, information on traffic density would be required. A vehicle counting system that can be used to determine the number of vehicles on busy highways with the aid of neural networks. YOLO (You Only Look Once), an object detection technique based on neural networks, is used by this system to identify cars. Simple Online and Real-time Tracking (SORT) algorithms are used to count and categories the vehicles in traffic recordings. We will determine the direction of vehicle movement after counting and attempt to apply the type of vehicle count, such as how many cars, how many bicycle and so on. The proposed neural network structure is better suited for real-time vehicle tracking because the computational complexity is reduced.

Keywords: Vehicle movement, YOLO V3, COCO Database, High Vehicle Density, Simple Online and Real-time Tracking (SORT).Feature Pyramid Networks(FPN).

1. INTRODUCTION

Through vehicle counting and traffic monitoring, a traffic monitoring system enables accident detection and traffic surveillance [1]. Through a framework, a traffic monitoring system can identify and infer the location of moving cars from video images while they are still in the frame [2]. In traffic flows with different vehicle models and a high vehicle density, it is challenging to precisely find and classify vehicles. Vehicle detection is further complicated by environmental modifications, diverse vehicle attributes, and generally sluggish detection speeds [3]. As a result, the creation of an algorithm capable of accurate vehicle detection and real-time computation is necessary for a real-time traffic monitoring system. As a result, it is both theoretical and feasible to detect automobiles efficiently and precisely from traffic photos or videos. [4].

Traffic monitoring via an intelligent transportation system allows for accident detection and assisted traffic surveillance. A traffic monitoring system is essentially a framework for detecting and estimating the location of vehicles in video images while they are still present in the scene [5]. In complex scenes with multiple vehicle models and a high vehicle density, it is difficult to accurately locate and classify vehicles in traffic flows. Furthermore, changing environmental conditions, varying vehicle characteristics, and relatively slow detection speeds all contribute to vehicle detectionlimitations [6]. As a result, a real-time traffic monitoring system requires the development of analgorithm capable of real-time computation and accurate vehicle detection. As a result, detecting vehicles in traffic images or videos can be done accurately and quickly [7].

Deep learning-based object detection algorithms have gained a great deal of attention. Using machine learning, these algorithms can automatically extract the features, to provide those with powerful image abstraction and automatic high dimensionality of the feature representation abilities [8].

http://doi.org/10.36893/JNAO.2023.V13I2.0120-0126

2. METHODOLOGY

2.1 Modules

2.1.1 Importing Libraries and Setting path

2.1.2 Model Backbone

2.1.3 Model Neck

2.1.4 Feature pyramids Network

2.1.5 GUI DESIGN

2.1.6 Getting Bounding Boxes

2.1.7 Non-Maximum Suppression.

2.1.8 Implementation of YOLO V3

2.1.1 Importing Libraries and Setting path - Using the Video Capture function in cv2, import the video with the objects and labels to be recognized.

2.1.2 Model Backbone - The primary purpose of Model Backbone is to automatically extract features from images [9]. To retrieve informative features from the input images, YOLO v5 uses Cross Stage Partial Networks (CSP) as a backbone.

2.1.3 Model Neck - The Model Neck is mostly used for making feature pyramids. When it comes to object scaling, feature pyramids help models generalise well. It aids in identifying the same object in different sizes and scales.

2.1.4 Feature pyramids Network - PANet is used as a neck in YOLO v5 to obtain feature pyramids. Detection of Objects Feature Pyramid Networks are primarily used within the model head to perform the final detection part, which is extremely useful and helps models perform well on unobserved data [10]. Other models, such as FPN, BiFPN, and PANet, are available and to use various types of feature pyramid techniques. (as shown in figure 1), etc.

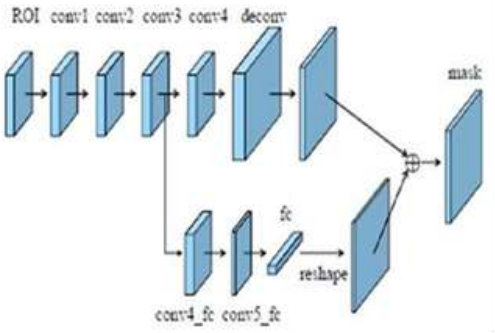

Figure 1: PANet

2.1.5 GUI DESIGN

The official Python module for the Qt for Python project is called PySide6, and it gives users access to the whole Qt 6.0+ framework [11].

It is simpler to incorporate into commercial projects when compared with pyqt.

Flexibility and cleaner codebase

- Main window, upload video, play, stop, counting vehicle tab, total count tab.

2.1.6 Getting Bounding Boxes: In the original study report, YOLO was only able to predict two bounding boxes per grid cell. Although it is possible to raise that number, only one class prediction can be performed for each grid cell, which restricts the detections when several objects are present in a single grid cell like "bicycle", "car", "motorbike", "bus", "truck" .

2.1.7 Non-Maximum Suppression.

Even though we eliminated the low confidence bounding boxes, it's possible that one object will still be the subject of much detection.

The final stage of these object detection methods, known as non-max suppression, is used to choose the item's ideal bounding box.

These object detection algorithms use non-max suppression to select the best bounding box from among the many predicted bounding boxes. Using this method, the less likely bounding boxes are "suppressed" in favour of the best one.

http://doi.org/10.36893/JNAO.2023.V13I2.0120-0126

To choose one bounding box, we pass it the confidence threshold value and NMS threshold value as arguments.

How does non-max suppression work?
Non-max suppression is required to select the best bounding box for an object and reject or "suppress" all other bounding boxes. The NMS considers two factors.
The model provides the score for objectivity.
The intersection of the bounding boxes, or
IOU.

2.1.8 Implementation of YOLO V3:

To implement the pre-trained YOLOv3 network, all that is required from the library is the config file of YOLOv3 which defines the layers and other essential specifics of the network like the number of filters in each layer, learning rate, classes, stride, input size for each layer and channels, output tensor etc. The config file gives the basic structure of the model by defining the number of neurons in each layer and differentkinds of layers. With the help of the config file, one could start training their model with either a pre-existing dataset like COCO, Alex net, MNISTdataset for handwritten digits detection etc. A database called Common Objects in Context (COCO) seeks to facilitate future studies on objectdetection, instance segmentation, image captioning, and the location of human important points as shown in figure 2. A sizable object detection, segmentation, and captioning dataset is called COCO.

Figure 2: Large scale object detection

2.2 YOLO V3

For object detection, YOLOv2 employed a customised CNN called Darknet-19 that had 30 layers total, including 19 from the original CNN and 11 more. Despite having a 30 layer architecture, YOLOv2 had trouble detecting small objects, which was thought to be because as the input travelled through each pooling layer, fine-grained information were lost. Identity mapping and concatenating characteristics were utilized from preceding layers to determine low lever features, in order to compensate for this.

	Type	Filters	Size	Output
	Convolutional	32	3 × 3	256 × 256
	Convolutional	64	3 × 3 / 2	128 × 128
1×	Convolutional	32	1 × 1	
	Convolutional	64	3 × 3	
	Residual			128 × 128
	Convolutional	128	3 × 3 / 2	64 × 64
2×	Convolutional	64	1 × 1	
	Convolutional	128	3 × 3	
	Residual			64 × 64
	Convolutional	256	3 × 3 / 2	32 × 32
8×	Convolutional	128	1 × 1	
	Convolutional	256	3 × 3	
	Residual			32 × 32
	Convolutional	512	3 × 3 / 2	16 × 16
8×	Convolutional	256	1 × 1	
	Convolutional	512	3 × 3	
	Residual			16 × 16
	Convolutional	1024	3 × 3 / 2	8 × 8
4×	Convolutional	512	1 × 1	
	Convolutional	1024	3 × 3	
	Residual			8 × 8
	Avgpool		Global	
	Connected		1000	
	Softmax			

Table 1: Darknet -53

http://doi.org/10.36893/JNAO.2023.V13I2.0120-0126

Ever after all this, it lacked several important aspects of an object detection algorithm which made it stable such as residual blocks, skip connections and up sampling layers.
These corrections were made and a new version of YOLO was born and that is known as YOLOv3. Additionally, YOLOv3 employs a variation of Darknet-53 with 53 convolutional layers learned on an image net for the purpose of classification with an additional of 53 more layers stacked onto it to make it a full-fledged network to perform classification and detection as shownin table 1. As a result, YOLOv3 is slower than the second version but a lot more accurate than its predecessors.

2.3 Structure of YOLOv3

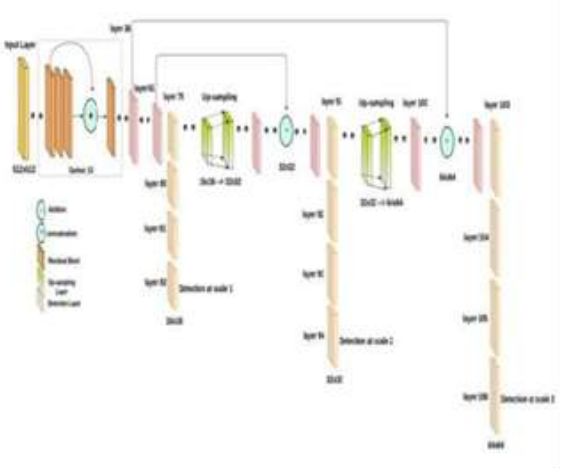

Figure 3: YOLOv3 Network Architecture

The main difference between YOLOv3 (as shown in above figure 3) and it's a predecessor is that it forecasts on three distinct scales. The initial detection is performed in the 82nd layer. If an input image of 416x416 is fed into the network, the feature map thus acquired would be of size 13x13. The other two scales at which detections happen are at the 94th layer yielding a feature map of dimensions 26x26x255 and the final detection happens at the 106th layer, resulting in afeature map with dimension 52x52x255. The detection which happens at the 82nd layer, is liable for the detection of large objects and the detections which happen at the 106th layer is liable for detecting small objects with the 94th layer, staying in-between these 2 with a dimension of 26x26, detecting medium size objects.
This kind of varied detection scale renders YOLOv3 good at detecting small objects than its predecessors as seen in below figure 4.

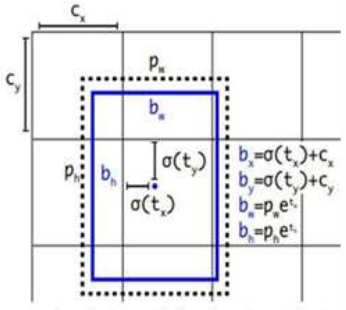

Figure 4: Object detection scale YOLOv3 uses 9 anchor boxes to localize

objects with 3 for each detection scale. The anchor boxes are assigned in the descending orderwith the largest 3 boxes of all for first detection layer which is used to detect large objects, the next 3 for the medium sized objects detection layer and the final 3 for the small objects' detection layer.
YOLOv3 utilizes 10 times as many bounding boxes than YOLOv2 does since it detects at three different scales. For instance, for an image of input size 416x416, YOLOv2Would predict 13x13x5 = 845 boxes whereas YOLOv3 would go for 13x13x5 + 26x26x5 + 52x52x5 = A whopping 10,647 boxes.
The loss function of YOLOv3 was modified

http://doi.org/10.36893/JNAO.2023.V13I2.0120-0126

$$\lambda_{coord}\sum_{i=0}^{S^2}\sum_{j=0}^{B}\mathbb{1}_{ij}^{obj}\left(x_i-\hat{x}_i\right)^2+\left(y_i-\hat{y}_i\right)^2$$
$$+\lambda_{coord}\sum_{i=0}^{S^2}\sum_{j=0}^{B}\mathbb{1}_{ij}^{obj}\left(\sqrt{w_i}-\sqrt{\hat{w}_i}\right)^2+\left(\sqrt{h_i}-\sqrt{\hat{h}_i}\right)^2$$
$$+\sum_{i=0}^{S^2}\sum_{j=0}^{B}\mathbb{1}_{ij}^{obj}\left(C_i-\hat{C}_i\right)^2$$
$$+\lambda_{noobj}\sum_{i=0}^{S^2}\sum_{j=0}^{B}\mathbb{1}_{ij}^{noobj}\left(C_i-\hat{C}_i\right)^2$$
$$+\sum_{i=0}^{S^2}\mathbb{1}_{i}^{obj}\sum_{c\in classes}\left(p_i(c)-\hat{p}_i(c)\right)^2 \quad (3)$$

This is the loss function of YOLOv2 where the last 3 terms in this image correspond to the function which penalizes for the objectlessscore predicted by the model for responsible for predicting things are bounding boxes. The second-to-last term is in charge of the bounding boxes that do not contain any objects, while the last term penalises the model for the class prediction score for the bounding box that contains predicted objects. The terms involved in the calculation of loss function in YOLOv2 were calculated using Mean Squared Error method while in YOLOv3; it was modified to Logistic Regression since this model offer a better fit than the previous one.

YOLOv3 executes classification ofmultiple labels for objects that are detected in images and videos. In YOLOv2, soft maxing is performed on all the class scores and whichever has the maximum class score is assigned to that object. This rests on the assumption that if one object belongs to one class, it can't be a part of another class. For instance, it is not necessary that an object belonging to the class Car would not belong to the class Vehicle. An alternative approach to this would be using logistic regression to predict class scores of objects and setting a threshold for predicting multiple labels. Classes that have scores greater than the thresholdscore are then assigned to the box.
YOLOv3 was benchmarked against popular state of the art detectors like RetinaNet50 and RetinaNet101 with the COCO mAP 50 benchmark where 50 stands how closely the predicted and actual bounding boxes match up will determine how accurate the model is as shown in chart diagram 1. This metric of evaluating CNNs is known as IOU, Intersection over Union. 50 over here equate to 0.5 on the evaluation's IOU scale. A mislocalization and false positive are both considered when the model's forecast is less than 0.5. YOLOv3 is really fast and accurate. When measured at 50 mAP, it is on par with the RetinaNet50 and RetinaNet101 but it is almost 4x faster than thosetwo models.In benchmarks where the accuracy metric is higher (COCO 75), the boxes need to be more aligned with the Ground Truth label boxes and here is where RetinaNet zooms past YOLO in terms of accuracy.

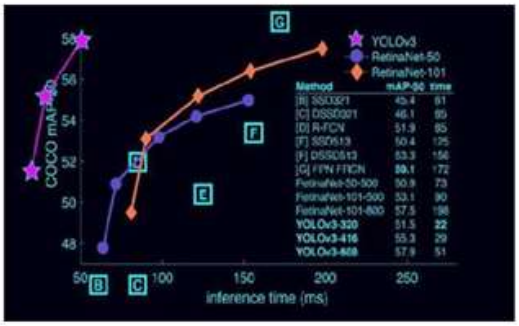

Chart diagram 1: Benchmark scores of YOLOv3 and other networks against COCO mAP 50

	backbone	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
Two-stage methods							
Faster R-CNN+++ [5]	ResNet-101-C4	34.9	55.7	37.4	15.6	38.7	50.9
Faster R-CNN w FPN [8]	ResNet-101-FPN	36.2	59.1	39.0	18.2	39.0	48.2
Faster R-CNN by G-RMI [6]	Inception-ResNet-v2 [21]	34.7	55.5	36.7	13.5	38.1	52.0
Faster R-CNN w TDM [20]	Inception-ResNet-v2-TDM	36.8	57.7	39.2	16.2	39.8	**52.1**
One-stage methods							
YOLOv2 [18]	DarkNet-19 [18]	21.6	44.0	19.2	5.0	22.4	35.5
SSD513 [11, 3]	ResNet-101-SSD	31.2	50.4	33.3	10.2	34.5	49.8
DSSD513 [3]	ResNet-101-DSSD	33.2	53.3	35.2	13.0	35.4	51.1
RetinaNet [9]	ResNet-101-FPN	39.1	59.1	42.3	21.8	42.7	50.2
RetinaNet [9]	ResNeXt-101-FPN	**40.8**	**61.1**	**44.1**	**24.1**	**44.2**	51.2
YOLOv3 608 x 608	Darknet-53	33.0	57.9	34.4	18.3	35.4	41.9

Table 2: Benchmark scores of object detection networks against COCO Dataset

http://doi.org/10.36893/JNAO.2023.V13I2.0120-0126

These are the metrics for different models with different benchmarks and it is quite observable that the mAP (mean Average Precision) for YOLOv3 is 57.9 on COCO 50 benchmark and 34.4 on COCO 75 benchmark.RetinaNet is better at detecting small objects but YOLOv3 is so much faster than versions of RetinaNet as shown in table 2.
YOLO uses a technique known as Non-Maximal Suppression (NMS) to eliminate duplicates of the same object being detected twice or more than that. It essentially retains on the bounding box with the highest confidence score. The initial step is to discard all the bounding boxes which have a confidence score lesser than the input of the threshold set for detected objects. If the threshold is set to 0.55, it retains bounding boxes with confidence scores more than or equal to 0.55
YOLO also uses an Intersection over Union metric to grade the algorithm's accuracy. IOU is a simple ratio of the predicted box's area of intersection to the predicted box's area of union with the ground truth box. Following the removal of bounding boxes with detection probabilities lower than the NMS threshold, YOLO discards all the boxes for objects with IOU scores lesser than the IOU threshold to eliminate duplicate detections further.

RESULTS AND DISCUSSION:
3.1EXISTING SYSTEM
R-CNN is the classical algorithm in object detection.
background subtraction technique
hierarchical traffic recognition
Pneumatic Tube Vehicle Counting
Embedded magnetometers
Inductive detector loops

DISADVANTAGE
- Not all units count or categorize cars.
- Tube installations are not long-lasting; the lifespan of tubes is only a few weeks.
- The two wheelers cannot be detected.
- The sample rate of IDL data delivered to traffic control systems is quite low.
- Not suited for installation on bridge decking made of metal.

3.2 PROPOSED SYSTEM
- The Deep Sort Algorithm uses Kalman Filters to track the objects, for better predictions.
- Also, we introduce an algorithm to count the vehicles using the movement direction such as "northbound" and "southbound" separately, then the intelligence system will take the decision to reduce the time based on the count of vehicle, further the traffic is managed accordingly.

3.3 Inputs and outputs
- The input is made up of a series of images, each with the shape (m, 608, 608, 3).
- The output includes a list of recognized classes along with a list of bounding boxes. As previously described, each bounding box is represented by six numbers: pc, bx, by, bh, bw, and c. Each bounding box in an 80-dimensional vector of c is represented by 85 numbers.

3.4 Encoding
Let's take a closer look at what this encoding represents in the below figure 5.

aA
Figure 5: YOLO architecture for Encoding

http://doi.org/10.36893/JNAO.2023.V13I2.0120-0126

If the center/midpoint of an object falls into a grid cell, that grid cell is responsible for detecting the object. Because we're using five anchor boxes, each 19 x 19 cell encodes data for five boxes.
Anchor boxes are defined solely by their width and height. For the sake of simplicity, we will flatten the last two dimensions of the shape encoding (19, 19, 5, 85). So the Deep CNN's output is (19, 19, 425).

CONCLUSION

This project develops a vehicle counting system for traffic surveillance. For efficient traffic maintenance, each type of vehicle is counted separately. Frontside-1x zoom footage was used. YOLOv3 is helpful in vehicle detection because counting is restricted to the detected object. The object "car" has the highest counting accuracy while seeing the output, followed by "motorcycle," "bus," and "truck," which have the lowest. Performance is also impacted by the video frame rate because it symbolizes the accuracy of the data the system processes. All things considered, this project was satisfactorily finished. Any future developments should lead to a better system.

REFERENCES

1. Shaoqing Ren, Kaiming He, Ross Girshick,and Jian Sun, “Towards Real-Time ObjectDetection with Region Proposal Networks”, Jan 2016.
2. Joseph Redmon, Santosh Divvala, Ross Girshick, Ali Farhadi, “YOLO, You Only Look Once: Unified, Real time object detection”, June 2015
3. Joseph Redmon, Ali Farhadi,“YOLOv3: An Incremental Improvement”, 2016
4. Indumathi.K, Gnana Abinaya, Thangamani K, Ashok Deva A, “Detection of Indian Traffic Signs”, 2016
5. K. S. Sang, B. Zhou, P. Yang and Z. Yang,"Study of group route optimization for iot enabled urban transportation network", IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData), pp. 888-893, 2017
6. Yu-Yun Tseng, Po-Min Hsu, Jen-Jee Chen and Yu-Chee Tseng, "Computer vision-assisted instant alerts in 5g", 2020 29th International Conference on Computer Communications and Networks (ICCCN), pp. 1-9, 2020.
7. Z. Dai, H. Song, X. Wang, Y. Fang, X. Yun, Z. Zhang, et al., "Video-based vehicle counting framework", IEEE Access, vol. 7, pp. 64460-64470, 2019.
8. M. A. Abdelwahab, "Accurate vehicle counting approach based on deep neural networks", 2019 International Conference on Innovative Trends in Computer Engineering (ITCE), pp. 1-5, 2019.
9. G. Oltean, C. Florea, R. Orghidan and V. Oltean, "Towards real time vehicle counting using yolo- tiny and fast motion estimation", IEEE 25th International Symposium for Design and Technology in Electronic Packaging (SIITME), pp. 240-243, 2019.
10. P. Sharma, A. Singh, S. Raheja and K. Singh, "Automatic vehicle detection using spatial time frame and object based classification", pp. 8147-8157, 2019.
11. Z. Liu, W. Zhang, X. Gao, H. Meng, X. Tan, X. Zhu, et al., "Robust movement-specificvehicle Countingatcrowded intersections", IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), pp. 2617-2625, 2020.

http://doi.org/10.36893/JNAO.2023.V13I2.0120-0126