Heavy-Vehicle Detection Based on YOLOv4 featuring Data Augmentation and Transfer-Learning Techniques

V Sowmya1 and R Radha2
1Research Scholar, Research Dept. of Computer Science, SDNBV College for Women, University of Madras, Chennai, India
2Associate Professor, Research Dept. of Computer Science, SDNBV College for Women, Chennai, India
E-mail: v.sowmy81@yahoo.in

Abstract. Real-time Vehicle detection is crucial in today’s era for our complex interconnected transportation ecosystem built on an advanced technological network of intelligent systems encompassing a wide range of applications such as autonomous vehicles, traffic Surveillance, advanced driver assistance systems, and etcetera. The significance of its application to digital transportation infrastructure embarks upon a distinct framework for heavy-vehicle detection integrated with the YOLOv4 algorithm for real-time detection. In this proposed work, two entities of heavy vehicles such as buses, trucks are considered. The crux of the model, an algorithmic computational mechanism incorporates Mosaic Data augmentation and Transfer-learning techniques that are applied to avoid over-fitting and to improve the optimal speed during training. Subsequently, a fine-tuning YOLOv4 algorithm is implemented for detecting the heavy-vehicle. The algorithm is tested for real-time situations in various traffic densities through Computer Vision. Experimental results show that the proposed system achieves higher detection accuracy of 96.54% mAP. More specifically, the performance of the proposed algorithm with the COCO test set and PASCAL VOC 2007 test set demonstrates improvement when compared with other state-of-the-art approaches.

1. Introduction

In the recent past, Heavy-Vehicle detection has become one of the challenging tasks in the field of computer vision such as traffic density estimation, traffic monitoring, self-driving cars, and so on [1] [2]. To tackle this challenge, we are applying an enhanced YOLOv4 model. Vehicle detection technique helps to identify, locate, and localize several visual instances of vehicles in a picture or a video. The proposed method can be used to calculate the number of instances of specific objects and identify their exact locations, along with labelling. The efficiency of this method has improved over time, enabling us with real-time environments. The deep learning approach enables the entire detection process without specifying the characteristics for the classification. The deep approach using the Convolutional Neural Networks (CNNs) provides a framework for this fine-tuned YOLOv4 model. Deep convolutional neural networks (DNNs) are the detection-oriented model to determine the objects in regions. Deep learning-based object detection frameworks can be mainly divided into two classes as Region-based CNN (RCNN) and its variants and YOLO and its variants. RCNN algorithms are known as a two-stage detector that divides the pipeline into two parts-proposal generations and region classification. YOLO is known as one stage detector which does not have a proposal generation. It spatially divides the entire image into a finite number of grid cells. Next, predicts the bounding boxes, then determines its class variables to label the boxes. YOLO's rapid prediction model has its significant real-time use.
Pooja Mahto et al [5] proposes an approach for vehicle detection using the YOLOv4 algorithm. UA-DETRAC dataset is considered. In this work, Spatial Attention Module (SAM), Anchor-box optimization using K-means clustering (ABK), Self-adversarial training (SAT), and Non-Maximum Suppression using distance IoU (DIoU-NMS) are the techniques applied in the YOLOv4 algorithm. The result shows that 67.7%mAP is achieved by the proposed system which is 10% higher than the base model. Jian Chen et al [7] proposes vehicle detection and tracking based on a machine learning algorithm. Initially, the authors have taken positive and negative samples of vehicle images. Then, they select the features through Haar-cascade and are further trained by the Ada-boost algorithm. For vehicle tracking, the author integrates the CamShift algorithm and the Kalman filter. Further, it combines with the Gauss mixture model for tracking the vehicle. Zhibin Cheng et al [3] proposes an anchor-based flower detection using the YOLOv4 algorithm. In this work, the algorithm is integrated with the mobile device and detects the flowers using computer vision. The result proves that the detection of flowers is much faster compared to other approaches. Tien-Szu Pan et al [4] proposes a real-time detection of an object underwater in video frames. In this work, Multi-scale Resnet (M-Resnet) is used for detection to improve accuracy even for small objects. The proposed system utilizes multiscale operations such as concatenation, feature pyramid networks, upsampling, dilated convolutions, and bottleneck residual blocks to increase efficiency. A success rate of 96.5% mAP (mean-Average Precision) is achieved using the proposed system. Fetulhak Abdurahman et al [6] propose a malaria parasite detection using modified YOLOv3 and YOLOv4 models. In this work, Anchor-box optimization using K-means clustering is applied for detecting the small images. Further increases the feature scale and also adds more detection layers without affecting the detection speed. The performance shows that YOLOv4-MOD and YOLOv3-MOD2 have achieved 96.32%mAP and 96.14%mAP, respectively. Kumar. B.C. et al[8] propose object detection for traffic surveillance. The proposed system uses the YOLOv3 algorithm for image datasets and the YOLOv4 algorithm for video datasets. KITTI dataset is considered for varying illumination. The result shows that 98% accuracy has been achieved for the image dataset and 99% accuracy achieved for the video dataset.

To overcome the above challenges, in this work, real-time vehicle detection is proposed based on the YOLOv4 algorithm. A fine-tuned YOLOv4 algorithm is employed for vehicle detecting heavy vehicles. The crux of the model, an algorithmic computational mechanism incorporates Mosaic Data augmentation and Transfer-learning techniques that are applied to avoid over-fitting and to improve the optimal speed during training. A computer-vision algorithm is applied for testing a model in a real-time environment. The experiment results prove that the proposed method achieves highly significant accuracy than other models.

The remaining part of the paper is arranged as follows – Section 2 explains the framework of the proposed model. Section 3 discusses the experimental analysis and results, and Section 4 concludes the paper.

2. The Proposed Framework
The architecture of Heavy-Vehicle detection is shown in Figure 1. The configuration of the proposed system is as follows; the foundation of the design is the CSPDarknet53 architecture that is utilized as a backbone network in the YOLOv4 model. Then, in the next layer, Spatial Pyramid Pooling (SPP), a Path Aggregation Neck (PAN), and YOLOv3 as head of the architecture are added to improve the training speed and accuracy. Next, for the enhancement process, the Mosaic Data Augmentation method is applied to reduce overfitting in the model, and the Greedy Non-maximum Suppression (NMS) is used to ensure that there is no overlapping in the proposed region. Finally, a fine-tuned YOLOv4 algorithm is used for detecting the vehicle. A Computer-Vision algorithm is used for testing a model in the form of images and videos in a real-time environment.
2.1. The Base Network

Bochkovskiy et al [9] proposed a YOLOv4 model published in the year April 2020 with significant changes in terms of speed and accuracy. YOLOv4 is the enhanced version of YOLO, which was initially developed by Redmon et al [1] in the year 2015. To get better precision, this paper proposes a sophisticated network CSPdarknet53 as shown in Table 1. CSPdarknet53 is a backbone network that constitutes CSPNet and darknet 53. It is applied as a feature extractor. CSPdarknet53 is built on a Dense net. The Dense net is designed to connect the layers with CNN to reduce the number of parameters, solving gradient problems, and also reuse the network features. When maintaining accuracy, the CSPDarknet decreases computation by incorporating gradient changes into feature maps. It also boosts the learning ability of CNN and removes the computational bottlenecks. Each convolutional layer contains Convolution, Leaky Relu, and Batch Normalization. Spatial Pyramid Pooling (SPP) is used in RCNN (Regional Convolution Neural Network), and it is an extra block over CSPDarknet53, which increases the receptive field and causes no reduction in speed, and differentiate most important context features. YOLOv4 chooses PAN instead of FPN (Feature Pyramid Network) used in YOLOv3 as the method of parameters from backbone level. YOLOv3(anchor-based) is utilized as the head of the architecture which improves mAP and small object detection. In this work, we used Mish activation which replaces the Leaky Relu to obtain better accuracy and generalization.

Table 1. The architecture of the backbone network CSPDarknet53

| Type                      | Filters | Output   |
|---------------------------|---------|----------|
| Input                     | 32      | 416 x 416|
| Darknet Conv2D SPP BN Mish |         |          |
| Mosaic Data Augmentation  | 64      | 208 x 208|
| Dense net 2 x Conv        | 128     | 104 x 104|
| 8 x Conv                  | 256     | 52 x 52  |
| 8 x Conv                  | 512     | 26 x 26  |
| 2 x Conv                  | 1024    | 13 x 13  |
2.2. Loss Function

In this work, DIoU and CIoU are used to optimize the loss function. DIoU loss is based on IoU (Intersection over Union), which uses normalized distances between the bounding box and the ground truth. For the removal of the redundant detection box, the DIoU loss function is applied in NMS (Non-maximum Suppression). The loss function of DIoU can be defined as in equation (1).

\[ R_{\text{DIoU}} = \frac{p^2(b,b^{\text{gt}})}{c^2} \]  

where \( b, b^{\text{gt}} \) are the mid-point of the anchor frame and target frame and \( p \) represent the Euclidean distance between two centers. Also, \( c \) denotes the diagonal distance of the rectangle which represents the boundary of the target and anchor box simultaneously.

CIoU can increase the intersection area between the prediction and ground truth box. and also reduces the distance between the center points. It maintains the frame height at a constant ratio. The loss function of CIoU can be defined as in equation (4), based on the equation (2), \( v \) is to measure the consistency of the aspect ratio as in equation (3).

\[ R_{\text{CIoU}} = \frac{p^2(b,b^{\text{gt}})}{c^2} + \alpha v \]  
\[ v = \frac{4}{\pi^2} (\arctan\frac{w^{\text{gt}}}{h^{\text{gt}}} - \arctan\frac{w}{h})^2 \]

where \( w \) and \( h \) denote the width and height respectively.

\[ L_{\text{CIoU}} = 1 - \text{IoU} + \frac{p^2(b,b^{\text{gt}})}{c^2} \]  

where IoU is defined as “Area of Intersection / Area of Union”

\[ \alpha = \frac{v}{(1-\text{IoU})+v'} \]

where \( \alpha \) represents a positive trade-off parameter, and it can be defined in formula (5).

2.3. Activation Function

The activation function transforms features that are progressed through the network. Mish [22] is an activation function that is configured to push left and right signals. In this work, we used Mish activation which replaces the Leaky Relu to achieve precise accuracy and generalization. It can be defined in equation (6).

\[ f(x) = x . \tanh(\zeta(x)) \]  

Where, \( \zeta(x) = \ln (1 + e^x) \) is the soft plus activation function. It is more precious and valuable than Swish [23] defined in equation (7) and Relu [24] defined in equation (8) while performing the experiments.

\[ f(x) = x . \text{sigmoid}(\beta x) \]  
\[ f(x) = \text{max}(0,x) \]

2.4. Training a Model enabled with deep features

In this proposed system, the essential features required for the detection mechanism of the YOLOv4 model are activated for training purposes to achieve fine-tuning. The activated features are Mosaic Data Augmentation and Transfer Learning Techniques, Cosine Annealing Scheduler, Eliminate Grid Sensitivity, CmBN (Cross-mini Batch Normalization), and DropBlock regularization. These enabled features, during the training, the model performs task-oriented execution on the datasets. The main goal of the data augmentation approach is to enhance the generalization of the proposed model. The new contribution is mosaic data augmentation, which consolidates four training images for tracking smaller objects, and encourages the model to localize different types of an image in various spatial portions of the frame. Transfer learning is the process of appropriating a precompressed CNN, that replaces the last convolutional layer and the creation of such layers in the dataset. By freezing layers weights, deep CNN
can generate image features like edges, and fully connected layers that proceed with this learning also apply it to classify data on the problem. The application of Transfer Learning is a tailored technique of the pre-trained model by optimization of the fine-tuning parameters. A cosine function is used to update the learning rate. The advantage of the cosine function is that it is cyclic; allowing it to get out of the local minima effectively than the step method or SGD. The previous versions of the YOLO model have a certain number of imperfections in predicting the boundary of anchor box regions. It is beneficial to define box coordinates differently to eliminate this problem. Cross Mini Batch Normalization (CmBN) populates statistics inside the entire batch, and its execution is performed on any GPU as carried out by the user. Drop block eliminates a block of the training features randomly at a given step in the network to train the model not to rely on key detection features. Drop block is a technique to deep-train the network for features that it may not otherwise rely upon. The proposed system is programmed with the following algorithm:

Step 1: Collect the Data from various resources such as icrawlers, real-time video, internet.
Step 2: Pre-processing the Data:
   - Cropping and Resizing the images using the Computer Vision technique.
   - Create Annotations(.xml) for each image.
   - Convert all Annotations files from .xml to .txt file.
Step 3: Train the Dataset using CSPDarknet53 (backbone network).
Step 4: Apply SPP (Spatial Pyramid Pooling), PAN (Path Aggregation Network), Mosaic Data Augmentation features while training the Dataset. Train the network with proper filter weights.
Step 5: Input the trained data and the transfer learning methodology is applied to fine-tuning the network parameters for detecting the heavy-vehicle with a learning rate of 0.0001 and 0.001.
Step 6: The dataset is trained and Save the final weights.
Step 7: Test the trained Dataset with COCO Dataset and PASCAL VOC 2007 Dataset
Step 8: Finally, Test the algorithm in a real-time environment such as Chennai and Pune and Visualize the test-images using Computer-vision Technique. The network YOLOv4 works fine for heavy-vehicle dataset classification.

3. Experimental Analysis and Results
The vehicle detection test environment is Google Colab, equipped with a Tesla K80 GPU for faster and efficient training of the network. The system environment for simulation is as follows: CPU - Intel Core i5-8265 1.80 GHz, SSD-512, RAM - 8 Gb.

3.1 Dataset
The custom vehicle dataset consists of 3500 images of buses and trucks. The images of the vehicles are downloaded from icrawlers. Some of the images are taken from real-time traffic video and also downloaded from the internet. Besides, rescale the images with respective width and height. This dataset is classified into a training set and a testing set. The Vehicle training dataset is set as two objects of different categories with a ratio of 7:3 where the count of images in the experimentation datasets are 2450 and 1050, respectively.

3.2 Training a Dataset for Heavy-Vehicle Detection
At the end of dataset pre-processing, it undergoes the creation of an annotation file for each image. The images and their respective annotation files with label files are bound. The YOLO layers are trained using yolo.cfg file to optimize configurations. The dataset was trained for 1000 iterations. The optimization of the training speed requires the attributes of the batch and subdivisions to be set to 64, and 16 respectively. Also, to achieve greater accuracy and maximum detection the height and width are set to 416*416. The total number of filters and the total number of classes are inter-dependent as filters=(classes+5) *3 i.e 21. The total network training time of the features was nearly 2 hours. The
yolov4, conv.137 pre-trained network is used for the proposed algorithm that implements the following activated features - Cosine Annealing Scheduler, CmBN (Cross-mini Batch Normalization), DropBlock regularization and Eliminate Grid Sensitivity. Then, set the parameters of the YOLOv4 algorithm’s network. The batch normalization layer undergoes a frozen process during training. Later, the iterations are carried out in cycles; each cycle equates to 500 times, with a learning rate of 0.0001. The predetermined parameters are frozen excluding the last three YOLO convolutional layers, and the final weights are saved. During the repeated process, the network is trained another 500 times with a learning rate of 0.001. In the end, the final weights are captured and applied for testing the model, and also to assess the performance metrics. The result of the average loss of YOLOv4 equates to 0.2841 as mentioned in Figure 2.

![Figure 2. Average loss of the custom vehicle dataset is 0.2841 with a learning rate of 0.0001 for 500 epochs and 0.001 for another 500 epochs.](image2.png)

### 3.3. Evaluation of Metrics

The parameters applied for testing the effectiveness of the proposed model are mAP, IoU, F1 score, and Confusion Matrix as shown in Figure 3 and Table 2. Mean Average Precision(mAP) is the mean value of average precisions, and Intersection over Union (IoU) is the average intersect over the union of objects and detections for a conf_threshold, and F1 score depends on the precision & recall and can be calculated based on confusion matrix. The Overall mAP is 96.54%, the Average IoU is 72.41% with a detection time is 1 second and the Average FPS is 34.2 is shown in Table 1. The evaluation analysis and confusion matrix are shown in Figure 3. 84% of Precision, 93% of Recall, 88% of F1Score are achieved as shown in Figure 3(a). The AP of 98.23% and 92.86% for Bus, Truck is achieved as shown in Figure 3(b).

![Figure 3. Various Evaluation Metrics(a) Classification Report of Precision, Recall and F1 Score (b) Confusion Matrix of 98% of Bus and 93% of Truck.](image3.png)
### 3.4. Testing a model with COCO Dataset and PASCAL VOC 2007 Dataset

COCO dataset and PASCAL VOC 2007 are public datasets for evaluating the proposed model. COCO facilitates multi-object labeling, segmentation mask annotations, image captioning, key-point detection, and panoptic segmentation annotations with a total of 81 categories. It is a multi-purpose dataset. PASCAL VOC 2007 caters to standardized image datasets for object class recognition and defined tools for accessing the datasets and annotations. It contains 9,963 images containing 24,640 annotated objects. This paper proposes an optimized model that applies YOLOv4 which is trained on a custom dataset and is reprocessed to learn features (or transfer them) to be trained on a new dataset (COCO test dataset and PASCAL VOC 2007 test dataset). In conception, Transfer Learning facilitates fine-tuning to execute the learned features on the ImageNet dataset and configuration of these features. Table 3 shows the comparison of detection results using the COCO test dataset. An overall increment of greater than 15% point in mAP (@.50) at 95.09% based on empirical results (COCO test dataset). Table 4 shows the comparison of detection results using the PASCAL VOC 2007 test dataset. An overall increment of greater than 12% point in mAP (@.50) at 94.33% based on empirical results (PASCAL VOC 2007 test dataset).

### Table 2. Testing Results of the Training Model

| Indicator | mAP   | Avg_IoU | Inference Time | Avg_FPS |
|-----------|-------|---------|----------------|---------|
| Value     | 96.54%| 72.41%  | 1s             | 34.2    |

### Table 3. Comparison of detection results using COCO test dataset

| Method       | mAP | Bus | Truck |
|--------------|-----|-----|-------|
| Faster RCNN  | 78.8%| 88.1%| 86.6% |
| SSD 300 [13] | 79.6%| 87.3%| 85.9% |
| SSD 512 [13] | 81.6%| 88.6%| 87.4% |
| Our Model    | 95.09%| 98.11%| 92.08% |

### Table 4. Comparison of detection results using the PASCAL VOC 2007 test dataset

| Method       | mAP | Bus | Truck |
|--------------|-----|-----|-------|
| Resnet101[15]| 82.1%| 90.8%| 90.9% |
| VGG16 [15]   | 73.2%| 80.9%| 73.01%|
| RCNN(Alex) [11]| 58.5%| 66.3%| 62.5% |
| RCNN(VGG16)  | 66.0%| 75.1%| 70.4% |
| SPP-Net (ZF) [12]| 60.9%| 67.7%| 67.9% |
| Our Model    | 94.33%| 97.75%| 90.91% |
Figure 4. (a),(b),(c),(d),(e) Heavy Vehicle Detection using fine-tuned YOLOv4 at various places in Chennai and (f),(g),(h) Heavy Vehicle Detection at various junctions in Pune.
3.5. Testing a model in real-time

The proposed system is evaluated to ensure the efficacy of the model, and compared with the other most recent advanced approaches is shown in Table 5. The classifier is experimented with by adopting a Computer Vision technique. Hence, at the final stage of detection, the tracking function helps to reduce the overhead of the computing process in future frames until the targeted vehicle subsequently dissipates from the video. During the computer vision technique, the video is converted into images. Fine-tuned YOLOv4 utilizes the proposed techniques. 84% of Precision, 93% of high Recall, 88% of F1Score based on empirical results are shown in Table 5. Average IoU is 72.41% and an overall increase of value with 20% point in mAP (@.50) at 96.54% based on empirical results as shown in Table 5. The classifier was experimented on various traffic intersections in Chennai and Pune, as shown in Figure 4. The proposed system is tested in live traffic densities in Chennai at various places as shown in Figure 4 (a), (b), (c), (d), (e).

Table 5. Comparisons results of Vehicle Detection models

| Models         | Overall(mAP) | Precision | Recall | F1Score | Avg_IoU |
|----------------|--------------|-----------|--------|---------|---------|
| Refined YOLOv4 | 67.7%        | 0.77%     | 0.71%  | 0.74%   | 65.7%   |
| [5]            |              |           |        |         |         |
| YOLO [16]      | 94.7%        | 0.89%     | 0.88%  | 0.88%   | -       |
| YOLOv2 [18]    |              | 0.79%     | 0.78%  | -       | 68%     |
| CNN [14]       | 75.7%        | -         | 0.92%  | -       | 82.3%   |
| YOLOv3 [19]    | 76.7%        | -         |        |         |         |
| Our Fine-tuning| 96.54%       | 0.84%     | 0.93%  | 0.88%   | 72.41%  |
| YOLOv4 model   |              |           |        |         |         |

4. Conclusion

In this study, a fine-tuned Deep-Learning architecture for accelerated and accurate vehicle detection is proposed. The effectiveness of the detection is obtained by the maximum utilization of the kernel parameters integrating with the YOLOv4 algorithm for detecting the vehicles, besides employing the Transfer-learning feature and Mosaic Data Augmentation techniques to improve speed, accuracy, and reduction of over-fitting with a focus on fine-tuning the limitations to achieve optimum performance. In comparison with other models, this proposed streamlined approach has the advantage of detection time. Experimental results show that the proposed system achieves higher detection accuracy of 96.54% mAP. In the future, this paper will proceed with emerging algorithms such as Efficient net for further enhancements to improve the detection results of heavy vehicles with different climatic conditions.

References

[1] Redmon J, Divvala S, Girshick R, and Farhadi A, 2016. You Only Look Once: Unified, Real-Time Object Detection. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. pp 779-88. DOI: 10.1109/CVPR.2016.91.

[2] Redmon, Joseph, and Farhadi Ali, 2018. YOLOv3: An Incremental Improvement. s://arxiv.org/pdf/1804.02767.pdf

[3] Zhibin Cheng and Fuquan Zhang, 2020. Flower End-to-End Detection Based on YOLOv4 Using a Mobile Device. *Wireless Communications and Mobile Computing*, Article ID 8870649, 9 pages.

[4] Tien-Szu Pan, Huang, Huang-Chu & Lee, Jen-Chun & Chen, and Chung-Hsien. 2020. Multi-scale ResNet for real-time underwater object detection. *Signal, Image, and Video Processing*, pp 1-9.
[5] Pooja Mahto, Garg, Priyamm, Seth, Pranav, and Panda, J, 2020. Refining Yolov4 for Vehicle Detection. *International Journal of Advanced Research in Engineering and Technology (IJARET)*. **11**(5) pp 409-19.

[6] Fetulahk Abdurahman, Kinde Fante, and Mohammed Aily, 2020. Malaria Parasite Detection in Thick Blood Smear Microscopic Images Using Modified YOLOV3 and YOLOV4 Models. 10.21203/rs.3.rs-74079/v1.

[7] Chen J and Dai L, 2019. Research on Vehicle Detection and Tracking Algorithm for Intelligent Driving. *International Conference on Smart Grid and Electrical Automation (ICSGEA)*, Xiangtan, pp 312-15.

[8] Kumar B C, Punita R, and Mohana, 2020. YOLOv3 and YOLOv4: Multiple Object Detection for Surveillance Applications. *Third International Conference on Smart Systems and Inventive Technology (ICSSIT)*. pp 1316-21.

[9] Bochkovskiy, Alexey-Wang, Chien-Yao-Liao and Hong-yuan, 2020. YOLOv4: Optimal Speed and Accuracy of Object Detection.

[10] Ren S, He K, Girshick R, and Sun J, 2017. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. in *IEEE Transactions on Pattern Analysis and Machine Intelligence*. **39**(6) pp 1137-49.

[11] Girshick R, Donahue J, Darrell T, and Malik J, 2014. Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation. *2014 IEEE Conference on Computer Vision and Pattern Recognition*. pp 580-87.

[12] He K, Zhang X, Ren S and Sun J, 2015. Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition. in *IEEE Transactions on Pattern Analysis and Machine Intelligence*. **37**(9) pp 1904-16.

[13] Liu, Wei & Anguelov, Dragomir & Erhan, Dumitru & Szegedy, Christian & Reed, Scott & Fu, Cheng-Yang & Berg, Alexander, 2016. **SSD: Single Shot MultiBox Detector**. *9905* pp 107-11.

[14] Chen, Linkai & Ye, Feiyan & Ruan, Yaduan & Fan, Honghui & Chen, Qimei, 2018. An algorithm for highway vehicle detection based on a convolutional neural network. *EURASIP Journal on Image and Video Processing*. 10.1186/s13640-018-0350-2.

[15] Leung, Ho, Chen, Xi-Zhi and Yu, Chao-Wei & Liang, Hong-Yi & Wu, Jian-Yi & Chen, Yen-Lin, 2019. A Deep-Learning-Based Vehicle Detection Approach for Insufficient and Nighttime Illumination Conditions. *Appl. Sci.*, **9** p 4769.

[16] Shinde, Shubham and Kothari, Ashwin and Gupta and Vikram, 2018. YOLO based Human Action Recognition and Localization. *Procedia Computer Science*. **133** pp 831-38.

[17] Tayara H, Gil Soo K, and Chong K T, 2018. Vehicle Detection and Counting in High-Resolution Aerial Images Using Convolutional Regression Neural Network. in *IEEE Access*. **6** pp 2220-30.

[18] Enzeng Dong J, Yanfang Zhu, Yuehui Ji and Shengzhi Du, 2018. An Improved Convolution Neural Network for Object Detection Using YOLOv2. *IEEE International Conference on Mechatronics and Automation*. pp.1184-88.

[19] Du Juan, 2018. Understanding of Object Detection Based on CNN Family and YOLO. *Journal of Physics: Conference Series*. **1004**. 012029. 10.1088/17426596/1004/1/012029.

[20] Fukai Zhang, Ce Li, and Feng Yang, 2019. Vehicle Detection in Urban Traffic Surveillance Images Based on Convolutional with Feature Concatenation. *Sensors*. **19** p 595.

[21] Pavan Vadapalli, 2020. *Ultimate Guide to Object Detection using Deep Learning*, https://www.upgrad.com/blog/ultimate-guide-to-object-detection-using-deep-learning/.
[22] Pierrick Rukery, 2020. *Explanation of YOLO V4 a one-stage detector*. https://becominghuman.ai/explaining-yolov4-a-one-stage-detector-cdac0826cbd7.

[23] Ramachandran P, Zoph B, and Le Q V, 2017. *Searching for activation functions*. http://arxiv.org/abs/1710.05941.

[24] Nair V and Hinton G E, 2010. Rectified linear units improve restricted Boltzmann Machines. *Proceedings of International Conference on Machine Learning (ICML)*. pp. 807–14.