ISSN (Online): 2456-0448
International Journal of Innovative Research in Management, Engineering, and Technology
Vol. 7, Issue 3, March 2022

The Real Time Remote Image Scene Classification From Pre-Trained And Custom Weights Using Deep Learning Methods

[1] Mr.G.Sekar M.E,[2] Mr.B.Rajmohan M.E,[3] M.Ramya

[1] Assistant professor, Department of Computer Science and Engineering, Adhiparasakthi Engineering College, Melmaruvathur.

[2] Assistant professor, Department of Computer Science and Engineering, Adhiparasakthi Engineering College, Melmaruvathur

[3] Department of Computer Science and Engineering, Adhiparasakthi Engineering College, Melmaruvathur

Abstract: ***Remote sensing image scene classification is a hot research area for its wide applications. It aims to quickly and accurately identify and locate a large number of objects of predefined categories in a given image scenes. Here we are using some pre-defined model and custom model to recognize the image scenes from given video feed and image. In the representative algorithms of each stage are introduced in detail. Then the public and special datasets commonly used in target detection are introduced, and various representative algorithms are analyzed and compared in this field. Finally, the potential challenges for target detection are prospected.***

Keywords: Object detection and recognition, Deep Learning, Classification performance.

1. INTRODUCTION

The process of recognizing objects in videos and images is known as Object recognition. This computer vision technique enables the autonomous vehicles to classify and detect objects in real-time. An autonomous vehicle is an automobile that has the ability to sense and react to its environment so as to navigate without the help or involvement of a human. The object detection and recognition are considered to be one of the most important tasks as this is what helps the vehicle detect obstacles and set the future courses of the vehicle. Therefore, it is necessary for the object detection algorithms to be highly accurate

Figure 1.1: Hauler

Though there are many machine learning and deep learning algorithms for object detection and recognition, such as Support vector machine (SVM), Convolutional Neural Networks (CNNs),
Regional Convolutional Neural Networks (R-CNNs), You Only Look Once (YOLO) model etc., it is important to choose the right algorithm for autonomous driving as it requires real-time object detection and recognition. Since machines cannot detect the objects in an image instantly like humans, it is really necessary for the algorithms to be fast and accurate and to detect the objects in real-time, so

that the vehicle controllers solve optimization problems at least at a frequency of one per second. This thesis is part of a collaborative project between Project Development Re- search Laboratory - BTH and Volvo CE. The intelligent machine navigation systems which includes autonomous driving, on scale site where it is cheaper and easier to radically
innovate and later implement the same in the real
The scale site is located at the PDRL lab in BTH. It is a small-scale representation of the real-world construction/excavation sites, which can be seen. The construction vehicles and construction site environment are quite different from that of city transportation environment, both of them serving unique purposes.

Figure 1.2: Excavator
Autonomous construction vehicles are ground-breaking as they can address the labor shortage problem as well as perform tasks for lengthy periods of time with minimal errors.

Figure 1.3: Wheeled Loader

AIM & OBJECTIVES

The aim of this thesis is to evaluate the classification performance of the suitable deep learning models for real-time object recognition and tracking of construction vehicles. The following objectives have been identified to fulfil the aim of this thesis work:

- To identify suitable and highly efficient deep learning
construction vehicles.
- Evaluate the classification performance of the selected deep learning models.
- Compare the classification performance of the selected models among each other and present the results.

PROBLEM STATEMENT
Though the learning approach for construction vehicles is the same as for traditional transportation vehicles such as cars, when it comes to autonomous driving, there exist unique challenges for the construction vehicles as their surroundings (construction and excavation sites) and driving conditions are different compared to cars or other transportation vehicles. Additionally, the characteristics and purpose of a construction vehicle is different from traditional transportation vehicle. Therefore, there needs to be research carried-out to evaluate the existing state-of-the-art deep learning models and identify the best deep

Figure 1.4: PDRL site

learning model for the detection and tracking of objects in the construction/excavation environments, as only little research has been carried out in this area of study, to date. There are several deep learning models that are currently in practice. Over Feat, VGG16, Faster R-CNN, YOLO are some of the popular deep learning models.

These models differ from each other majorly in their architecture and performance due to the variables such as Layer depth and Prediction time. For example, Over Feat model is 8 layers deep while VGG16 model is 16-19 layers deep. Fast R-CNN model is known for its hybrid ability to capture the accuracy of deep layer models as well as improving their speed at the same time.

YOLO, which is a 12-layer model, is known for its amazing prediction speed as it can predict up to 45 frames per second.

DOCUMENT STRUCTURE

This thesis report discusses about the state-of-the-art neural networks suitable for real-time object recognition and evaluates their performance against the dataset of construction vehicles at the scale site. The background and related work about the neural networks and related work performed previously by other authors in the field of object detection using neural networks. The research questions formulated for this thesis and research methodologies selected to answer them. Details about the literature review and experiment are also discussed. The findings of literature review and results of the experiment are presented. Discuss and presents analysis regarding the results of literature review and experiment and finally conclusions drawn from the analysis of results and future work in this area of study are discussed in the of the document.

Figure 1.5: Exemplar Excavation Site [1]

Figure 1.6: Exemplar Construction Site [2]

II.LITERATURE REVIEW

The following conclusions have been drawn from the results obtained through the literature review.

AUTHOR: Z. Deng, H. Sun, S. Zhou, J. Zhao, L. Lei, and H. Zou

YEAR: 2020

- From the results of performance of various object detection models on the MS COCO dataset, it can be deduced that SSD and R-FCN models are faster when compared to the Faster R-CNN.

- But if accuracy is given preference over speed, then Faster R-CNN performs better than SSD and R-FCN models.

AUTHOR: C. C. Nguyen ,G. S.Tran ,T. P. Nghiem, N. Q. Doan, D. Gratadour
YEAR: 2019

- Faster R-CNN is the most accurate model while using Inception Res Net, running at a speed of 1 image per second which satisfies the minimum requirement to perform object detection and recognition in real-time.
- SSD is faster compared to other object detection models but has difficulty in detecting small objects.
- Speed of the Faster R-CNN increases as the number of proposals decrease, also decreasing the accuracy of the model.

AUTHOR: J. Redmon, S. Divvala, R. Girshick, and A. Farhadi

YEAR: 2018

- According to Redmond et al [8][9] YOLOv3 is able to detect 10 times faster than the state-of-the-art methods. Hence YOLOv3 and its variant Tiny-YOLOv3 has been selected for the experimentation.

It has to be noted that since the construction vehicles move rather slow, speed need not be a concern as long as the algorithm is able to perform object detection and recognition in real-time. But considering the future scope of this research which is autonomous driving of these vehicles at the construction site and that the construction vehicles look similar at certain angles; accuracy has to be given importance. Considering all the above- mentioned points, it can be deduced that Faster R-CNN performs at the same speed as that of SSD and R-FCN models at an accuracy of 32 mAP, by reducing the number of proposals to 50. Therefore, Faster R-CNN, YOLOv3 and Tiny-YOLOv3 have been considered to be suitable and efficient models in real-time detection and tracking of the construction vehicles at the scaled site.

III.SYSTEM ANALYSIS

EXISTING SYSTEM

To learn about thousands of objects from millions of images, we need a model with a large learning capacity. However, the immense complexity of the object recognition task means that this problem cannot be specified even by a dataset as large as ImageNet, so our model should also have lots of prior knowledge to compensate for all the data we don't have.

DRAWBACKS OF EXISTING SYSTEM

- It requires the high system volume
- The Accuracy is poor
- It never recognize some trained objects
- It takes large time to Training and testing the model

PROPOSED SYSTEM

- In process, the work usually encounters occurrence of errors or the slow processing of detection and classification due to the tiny and light-weight datasets to beat these problems, this paper proposes You Only Look Once version 3 (YOLOv3) based detection and custom classification approach.
- This is Darknet based approach for classifying the image and it can sense the image scenes.
- This model improves the time of computation and speed also as efficiently identify the objects in images and videos

TECHNOLOGY USED

- Here we are using python programming for Training and testing the model.

- This project involves with both machine learning and deep learning concepts.
- For further development we can use Flask or Django framework for deployment purpose.

IV.SYSTEM REQUIREMENTS

FUNCTIONAL REQUIREMENTS

The Functional requirements specify which input should be given to the model and which output should be produced by the model. Each functional requirement should specify the detailed description of all data and their sources.

NON-FUNCTIONAL REQUIREMENTS

These are the quality constraints that the system must satisfy according to the project contract. The priority or extent to which these factors are implemented varies from one project to another. They are also called non-behavioral requirements.

HARDWARE REQUIREMENTS

- Processor : Any Processor above 500 MHz.
- Ram : 4 GB
- Hard Disk : 4 GB
- Inputdevice :StandardKeyboardand Mouse.
- Output device: VGA and High Resolution Monitor.

SOFTWARE REQUIREMENTS
OperatingSystem: Windows 7 or higher
Programming: Python 3.6 and related libraries
Software :AnacondaPrompt(Anaconda3)

V.BACKGROUND & RELATED WORK

In this chapter, theoretical knowledge required for understanding the methods dis- cussed in the chapter 3 has been provided. Details about machine learning, neural networks, and computer vision have been discussed, followed by the explanation of the Faster R-CNN, YOLOv3 and Tiny- YOLOv3. Related work performed by other researchers related to this area of study has also been presented towards the end of this chapter.

MACHINE LEARNING

Machine learning is one of the applications of Artificial Intelligence (AI) which enables the computers to learn on their own and perform tasks without human intervention. There are numerous applications of machine learning algorithms in the field of computer vision. With the help of machine learning, formulation of some of the most complex problems have been performed easily. Various computer programs which were previously programmed by humans, sometimes by-hand, are now being programmed without any human contribution with the help of machine learning. In the recent years, due to remarkable increase in the availability of humongous sources of data and feasibility of computational resources, machine learning has become predominant with wide range of applications in our daily lives.

TYPES

- Supervised Learning
- Unsupervised Learning
- Reinforcement Learning

SUPERVISED LEARNING

Supervised learning is considered to be the most elementary class of machine learning algorithms. As the name suggests, these algorithms require direct supervision. In this type of learning, the data labelled/annotated by humans is spoon-fed to the algorithm. This data contains the classes and locations of the objects of interest. Eventually, the algorithm learns from the annotated data and predicts the annotations of the new data previously not known to the algorithm, after the completion of training process. Some of the popularly utilized supervised learning algorithms are:

Neural Networks

- Decision Trees
- Random Forest
- K-Nearest Neighbors
- Linear Regression
- Logistic Regression
- Support Vector Machines

UNSUPERVISED LEARNING

In the unsupervised learning, the algorithm tries to learn and identify useful properties of the classes from the given annotated data, without the help or intervention of a human. Apriori algorithm, K-means clustering, etc. are some of the common unsupervised learning algorithms.

REINFORCEMENT LEARNING

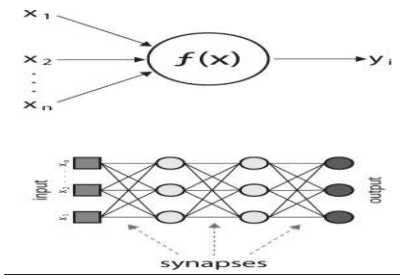
In this type of learning, the machine is allowed to train itself continually using trial and error. As a result, the machine learns from past experience and attempts to capture the best knowledge possible to predict accurately. Markov Decision Process, Q-learning, Temporal difference, etc. are some of the examples of reinforcement learning.

ARTIFICIAL NEURAL NETWORKS

Artificial neural networks are a popular type of supervised learning model. A special case of a neural network called the convolutional neural network (CNN) is the primary focus of this thesis. The name 'Artificial Neural Networks' was given to this model because they were developed to imitate the neural function of the human brain. An artificial neural network consists of a set of neurons connected to each other and are grouped into layers to replicate the neural function of our brain.

Similar to the neurons in a human brain, the neurons in an artificial neural network function as units of calculation. The connections between neurons are known as 'synapses' which are nothing but weighted values. There- fore, in a simple sense, when an input value is provided at a neuron (x1, x2, . . . , xn), it traverses the synapse, multiplying its value with the weighted value of

Figure 5.1: A Simple Artificial Neural Network



the synapse (w1, w2, . . . , wn). Bias 'b' is then added to the summation of these values. This will be the output of the neuron. Since a neuron does not know its boundary, a mapping mechanism is required to map the inputs to the output, known as the 'Activation function'.

In a fully connected feed-forward multi-layer network, all the outputs of a layer of neurons is fed as an input to every neuron of the next layer.

As a result, some of layers get to process the original input data, while some layers get to process the data that has been obtained from neurons from the previous layer. Therefore, the number of weights of any neuron in the network is equal to the number of neurons in the layer previous to the layer of the neuron in question.

ISSN (Online): 2456-0448
International Journal of Innovative Research in Management, Engineering, and Technology
Vol. 7, Issue 3, March 2022

$$y = \sum_{i=1}^{n} (w_n * x_n) + b \qquad (2.1)$$

In the above equation, 'x' is the input value given at the neuron, 'w' is the weighted value of the synapse, 'n' is the number of neurons, 'b' is the bias and 'y' is the output of the network. Therefore, according to the equation (2.1), the value of output 'y' is equal to the summation of the product of the values of 'x' with their corresponding weights and bias 'b'.

A multi-layered artificial neural network, typically includes three types of layers: an input layer, one or more hidden layers and an output layer. The input layer usually merely passes data along without modifying it. Most of the computation happens in the hidden layers. The output layer converts the hidden layer activation to an output, such as a classification.

BACKPROPAGATION

Figure 5.2: Structure of a single perceptron or neuron [4]

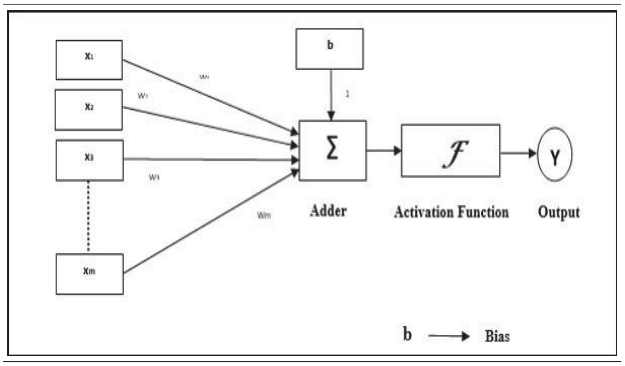


Figure 5.3: Multi-layer Artificial Neural Network [5]

Though the artificial neural networks have shown predominant applications in various fields and aided in achieving groundbreaking innovations during recent times, the concept of neural networks is quite old. The neural networks were previously known as 'perceptrons' and have been in action since the 1940s. They were not popular as they are now due to the fact that they were single layered and required high computational power and data which was difficult to find during that time. They have come to limelight mainly due to the inception of a technique known as 'Backpropagation'. The technique was first put forth by Rum el hart et al. in the year 1986. Using this technique, networks can rearrange the weights of hidden layers in case the output is different from the expected output. The error is calculated and backpropagated to all the layers of the network to adjust the weights according to the requirement.

COMPUTER VISION

Computer vision is the area of study in which computers are empowered to visualize, recognize and process what they see in a similar way as that of humans. The main aim of computer vision is to generate relevant information from image and video data in order to deduce something about the world. It can be classified as a sub-field of artificial intelligence and machine learning. This is quite different from image processing, which involves manipulating or enhancing visual information and is not concerned about the contents of the image.

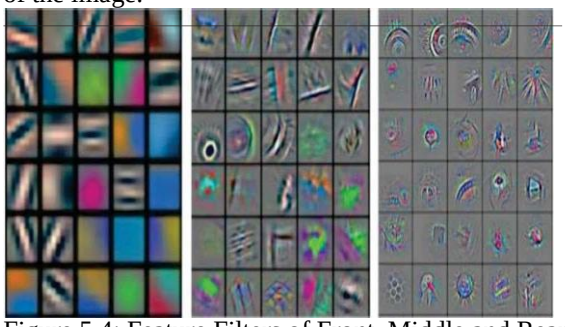


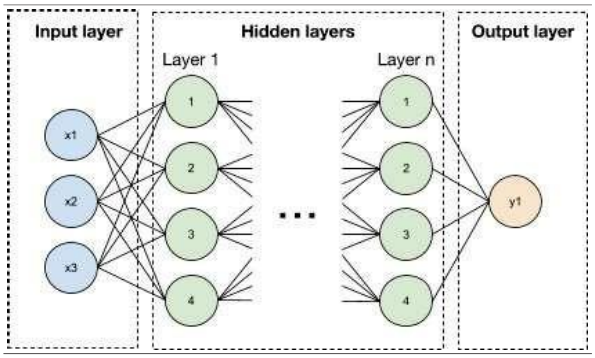
Figure 5.4: Feature Filters of Front, Middle and Rear-End Layers in a CNN [6]

Applications of computer vision include image classification, visual detection, 3D scene reconstruction from 2D images, image retrieval, augmented reality, machine vision and traffic automation. Today, machine learning is a necessary component of many computer vision algorithms. These algorithms are typically a combination of image processing
and machine learning techniques. The major requirement of these algorithms is to handle large amounts of image/video data and to be

able to perform computation in real-time for wide range of applications. For example, real-time detection and tracking.

CONVOLUTION NEURAL NETWORK

There are various types of artificial neural networks that are considered to be very important such as Radial basis function neural network, Feed-forward neural network, Convolutional neural network, Recurrent neural network, Modular neural network etc. Among these types of networks, the convolutional neural networks (CNNs) are effective in applications such as image/video



recognition, semantic parsing, natural language processing and paraphrase detection. A convolutional neural network typically comprises of three layers Convolutional layer, Pooling layer and Fully connected layer.

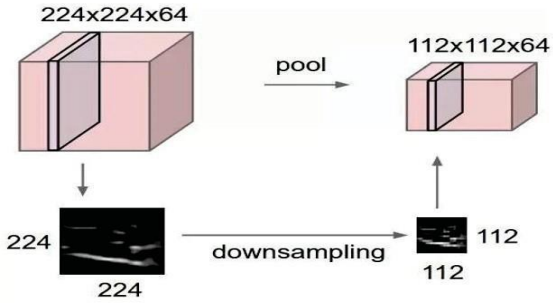


Figure 5.5: Pooling Layer [7],

ISSN (Online): 2456-0448
International Journal of Innovative Research in Management, Engineering, and Technology
Vol. 7, Issue 3, March 2022

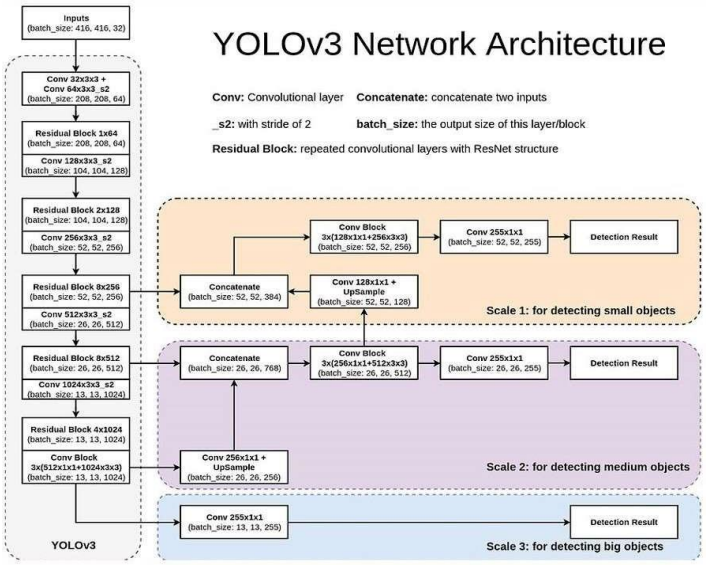


Figure 5.5: Pooling Layer [7],

CONVOLUTIONAL LAYER

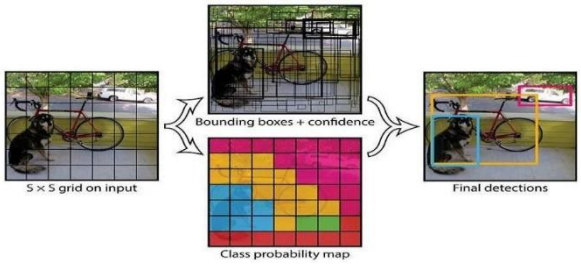
A convolutional neural network consists of one or more convolutional layers. These layers can either be pooled or fully connected. A convolutional layer generally executes tasks that require heavy computation. It comprises of a set of filters that have the ability to learn. Though the filters are small in ∗size∗, they reach to the entire depth of the input. The dimensions of a filter are generally represented by l w d, where 'l' denotes the height of the length of the filter, 'w' denotes the width while 'd' denotes the depth of the feature filter which is equal to the number of color channels present.

In general, the convolution process is executed by a feature filter upon sliding on the input layer of the neural network, as a result of which a feature map is generated. The layer executing the convolution process is known as a convolutional layer. Hence, the networks that consist of convolutional layers are called as convolutional neural networks. In the initial stages, the input layer is searched for any specific pattern by the filter. During the training of the algorithm, the filter searches for the sake of learning to recognize a pattern which eventually becomes a search to validate the existence of a specific pattern, during the testing stages. In reality, many feature filters exist, learning to recognize various patterns.

POOLING LAYER

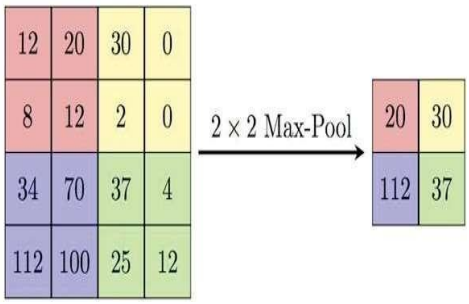
Pooling layers are also an important component of a convolutional neural network. The main function of a pooling layer is to decrease the number of parameters and computation present in the network by decreasing the spatial size gradually and continuously. This action is necessary to cut down the features that the filter has learnt and no longer requires the whereabouts of their location.

Figure 6.1: Image Splitting and Bounding-boxes Prediction [8]



There are many benefits using a pooling layer such as limiting of over-fitting, which is a state that occurs when the algorithm fits the data very closely by showing low bias and high variance. Though there are various types of pooling, max pooling is one of the most popular ones in practice. This type of pooling conveniently down- samples the layer while keeping the depth constant. The depiction of a pooling layer while provides an example of a 2x2 map pooling.

ISSN (Online): 2456-0448
International Journal of Innovative Research in Management, Engineering, and Technology
Vol. 7, Issue 3, March 2022



VI. PROJECT DESCRIPTION
SYSTEM ARCHITECTURE

YOLOv3

The state-of-the-art object detector YOLOv3 is designed to achieve high accuracy along with real-time performance. YOLOv3 is an improvement over the previous version of YOLO. It uses a single neural network, which predicts the objects position and class score in a single iteration.This is achieved by considering object detection problem as a regression problem, which in turn changes the input images to their corresponding class probabilities and positions. YOLO generates Many S x S grids from the input image and boundary boxes B are predicted, which consists of height, width, box center x and y. Each of these boxes have their own P (object probability) value and predicts the number of classes in it as C and has a conditional class probability Pclass in the S x S having an object in it. The overall prediction of the network is S x S x (Bx5 + C) in which the digit 5 represents each box coordinates as 4 and 1 as object probability. During the test, the network computes the number of classes present in each grid by using the equation (2.2). Pmin is defined at the start of the test and system detects only the objects whose Pclass > Pmin. During the post-processing stage, the duplicated detection of the same object is omitted using Non-maximal suppression.

P (classi) = P (classi|object) ∗ P (object) (2.2)

Here, P (classi) is the probability of ith class. P (Object) is the probability of grid containing the object and P (cl|assi object) is the conditional class probability of the ith class in which the object is present. In YOLO, only the bounding boxes with the greatest value of confidence are selected since every grid-cell is predicting multiple bounding bo∗xes. Therefore, YOLO generates a tensor as an output whose value is equal to S x S x (B 5 + C).In YOLOv3, the bounding boxes have been replaced by 'Anchors' which resolve the unstable gradient issue that used to occur while training of the algorithm. Therefore, YOLOv3 predicts outputs with confidence scores by generating a vector of bounding boxes whenever an input is given to the algorithm in the form of an image or a video. YOLOv2 used a feature extractor known as the Darknet-19, which consisted of 19 convolutional layers. The newer version of this algorithm, YOLOv3 uses a new feature extractor known as Darknet-53 which, as the name suggests, uses 53 convolutional layers while the overall algorithm consists of 75 convolutional layers and 31other layers making it a total of 106 layers. Pooling layers have been removed from the architecture and replaced by another convolutional layer with stride '2', for the purpose of down-sampling. This key change has been made to prevent the loss of features during the process of pooling which is created by 'Cyber ail AB' clearly depicts the architecture of YOLOv3 algorithm.1 x 1 detection kernelsare applied on the feature maps with three unique sizes located at three unique places in the network. The shape of the detection kernel is1 x 1 x (B (4 + 1 +C)), where 'B' is the number of bounding boxes that can be predicted by a cell located on the feature map, '4' represents the number of bounding box attributes, '1' represents the object confidence and 'C' represents the number of classes. Splitting of an image and bounding-box prediction in YOLOv3 architecture of YOLOv3 algorithm trained on COCO dataset which has 80 classes and bounding boxes are considered to be 3. Therefore, kernel size would be 1 x 1 x 255. In YOLOv3, the dimensions of the input image are down sampled by 32, 16 and 8 to make predictions at scales 3,2 and 1 respectively. In the size of the input image is 416 x 416. As mentioned in the earlier section, the total number of layers in YOLOv3 is 106. As shown in the network architecture diagram Figure 2.8, the input image is down sampled by the network for the first 81 layers. Since the 81st layer has a stride of 32, the 82nd layer performs the first detection with a feature map of size 13 x 13. Since a 1 x 1 kernel is used to perform the detection, the size of the resulting detection feature map is 13x 13 x 255 which is responsible for the detection of objects at scale 3. Following this, the feature map from 79th layer is up sampled by 2x after subjecting it to a few convolutional layers, resulting in the dimensions 26 x 26. This is then concatenated with the feature map from 61st layer. The features are fused by subjecting the concatenated feature map to a few more 1 x 1 convolutional layers. As a result, the 94th layer performs the second detection with a feature map of 26 x 26 x 255, which is responsible for the detection of objects at scale 2. Following the second

detection, the feature map from 91st layer is up sampled by2x after subjecting it to a few convolutional layers, resulting in the dimensions 52 x 52. This is then concatenated with the feature map from 36th layer. The features are fused by subjecting the concatenated feature map to a few more 1 x 1 convolutional layers. As a result, the 106th layer performs the third and final detection with a feature map of 52 x 52 x 255, which is responsible for the detection of objects at scale 1. As a result, YOLOv3 is better at detecting smaller objects when compared to its predecessors YOLOv2 and YOLO.

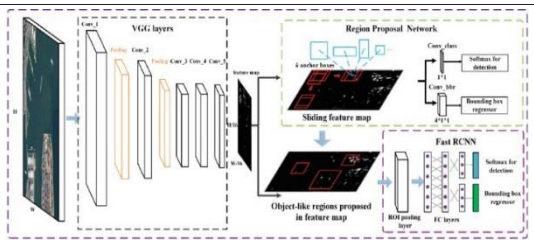
FASTER R-CNN

Faster R-CNN by Ren et al. is an integrated method. The main idea is to use shared convolutional layers for region proposal generation and for detection. The authors discovered that feature maps generated by object detection networks can also be used to generate the region proposals. The fully convolutional part of the Faster R-CNN network that generates the feature proposals is called a Region Proposal Network (RPN). The authors used Fast R-CNN architecture for the detection network.

A Faster R-CNN network is trained by alternating between training for Region of Interest (RoI) generation and detection. First, two separate networks are trained. Then, these networks are combined and fine-tuned. During fine-tuning, certain layers are kept fixed and certain layers are trained in turn. The structure of Faster R-CNN model. It typically comprises of three neural networks namely a Feature Network, a Regional Proposal Network and a Detection Network. The Feature Network is responsible for generating good features from the input images while maintaining the original attributes of the input image in the output, such as shape and structure. An image classification network generally takes the role as a Feature Network.

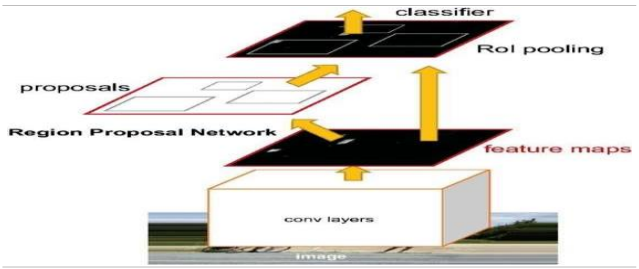
The Regional Proposal Network (RPN) comprises of three convolutional layers, a layer each for classification and bounding box regression while the third one is a common layer that feeds into these two layers. The Regional Proposal Layer is responsible for generating

Figure 6.3: Structure of Faster R-CNN model [10]



numerous bounding boxes that have a high probability of including an object. These bounding boxes are also called as Region of Interests (ROIs). A bounding box is identified using the co-ordinates of the pixels located at two diagonal corners of the box, followed by a value of 1, 0 or -1. A value of 1 indicates that there is an object present in that particular bounding box. Similarly, a value of 0 indicates that there is no object present while a value of -1 indicates that the particular bounding box can be ignored.The Detection Network is responsible for generating the final class and its corresponding bounding box by taking the input from both the Feature Network and Regional Proposal Network. The Detection Network generally comprises of a classification layer and a bounding box regression layer. Additionally, a pair of stacked common layers are shared among the two layers. These four layers are fully connected layers. The features are cropped as per the bounding boxes so that the network classifies only the internal part of the bounding boxes.Using shared convolutional layers, region proposals are computationally almost cost-free. Computing the region proposals on a CNN has the added benefit of being realizable on a GPU. Traditional RoI generation methods, such as Selective Search, are implemented using a CPU. For dealing with different shapes and sizes of the detection window, the method uses special anchor boxes instead of using a pyramid of scaled images or a pyramid of different filter sizes. The anchor boxes function as reference points to different region proposals centered on the same pixel.

Figure 6.4:TheArchitecture of Faster R-CNN [11]



In the architecture diagram of Faster R-CNN, the trained network receives a single image as input. In this case, the Feature Network is VGG, which is an image classification network. The shared fully convolutional layers of this network generate good feature maps from the input image while maintaining the size and structure of the original image in the output of this network. The resulting feature maps are fed into the Regional Proposal Network (RPN). Here, a number of bounding boxes are generated by a mechanism called as anchor boxes. Anchors are nothing but the pixels present on the feature image. In general, 9 boxes of different shapes and sizes, with the anchor as their center, are generated for each anchor.

These probability scores are later normalized using SoftMax function. The resulting bounding boxes (ROIs) are fed into the Detection Network along with the output of the Feature Network. Since the resulting feature maps can be of various sizes, ROI pooling layer is introduced to crop and scale the features to 14 x 14. These features are later max-pooled to 7 x 7 and fed into the Detection Network in the form of batches. The pair of stacked common fully connected layers, along with the classification layer and bounding box regression layer can be seen in the Figure 2.10. The output of this network is the generation of final class and its corresponding bounding box.

TINY-YOLOv3

Tiny-YOLOv3 is the smaller and simplified version of YOLOv3. Even though the number of layers in Tiny-YOLOv3 is quite less when compared to that of YOLOv3, the accuracy of the model is almost the same as that of its bigger self when high frame rates are considered. Tiny-YOLOv3 consists of only 13 convolutional layers and 8 max-pool layers and therefore, requires minimal memory to run which is way less than the layers in YOLOv3. The major difference between YOLOv3 and Tiny- YOLOv3 is that the former is designed to detect objects at three different scales while the later can only detect objects at two different scales. Apart from these differences, the working of both these variants is similar.

Compared to YOLOv3, the number of convolutional layers is greatly reduced in Tiny-YOLOv3. The primary structure of Tiny-YOLOv3 only has 13 convolutional layers while the overall number of layers is 23. A limited number of 1 x 1 and 3 x 3 kernels are utilized to extract the features in Tiny-YOLOv3. Unlike YOLOv3, which uses convolutional layers of stride 2 for the purpose of down sampling, the Tiny-YOLOv3 uses the pooling layer. The convolutional layer structure of Tiny- YOLOv3 is similar to that of YOLOv3. Tiny-YOLOv3 performs detections at two different scales, as shown in1 x 1 detection kernels are applied on the feature maps with two unique sizes. located at two unique places in the network. The shape of the detection kernelis1 x 1 x (B (4 + 1 + C)), where ‘B’ is the number of bounding boxes that can be predicted by a cell located on the feature map, ‘4’ represents the number of bounding box attributes, ‘1’ represents the object confidence and ‘C’ represents the number of classes.

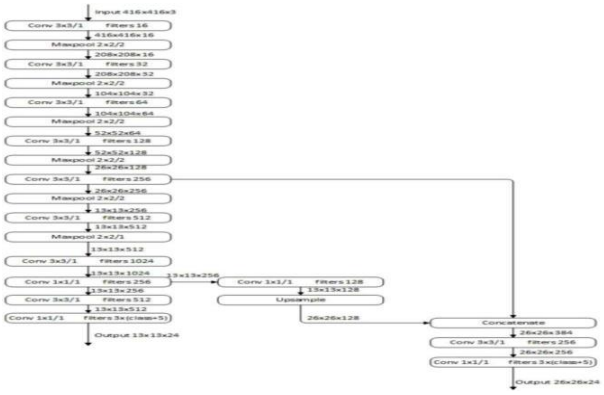


Figure 6.5: Architecture of Tiny-YOLOv3 [12]

The architecture of Tiny-YOLOv3 algorithm trained on COCO dataset which has 80 classes and bounding boxes are considered to be 3. Therefore the kernel size would be 1 x 1 x 255.

In the size of the input image is 416 x 416. As mentioned in the earlier section, the total number of layers in Tiny-YOLOv3 is 23. As shown in the network architecture diagram Figure 2.11, the input image is max-pooled by the network for the first 15 layers. The 15th layer performs the first detection with a feature map of size 13 x 13. Following this, the feature map from 14th layer is up sampled by 2x after subjecting it to a convolutional layer, resulting in the dimensions 26 x 26. This is then concatenated with the feature map from 9th layer. The features are fused by subjecting the concatenated feature map to a 1 x 1 and a 3 x 3 convolutional layer. As a result, the 23rd layer performs the second and final detection with a feature map of 26 x 26 x 255, which is responsible for the detection of objects at scale 1.

RELATED WORK

Yukui Luo et al, presented an OpenCL based implementation of the Deep Convolutional Neural Network, which is one of the most advanced deep learning frame works. Their framework aimed at three major contributions- a real-time object recognition system, framework with low power consumption, that can be applied even in portable devices, framework that can work on various compute devices [40].The framework was evaluated by comparing its speed with the CUDA framework, based on YOLO V2 benchmark.
Al pay din proposed an adaptive fuzzy based network topology which is run alongside Deep Convolutional Neural Networks, to achieve highly efficient object recognition for long range images that are low in contrast and having variable, noisy backgrounds.

Daniel et al , presented a 3D Convolutional Neural Network (CNN) architecture 'VoxNet', to achieve accurate and efficient object detection, using LiDAR data and RGBD point clouds. They evaluated their approach on state- of-the-art bench- marks that are publicly available and found that their approach achieved accuracy beyond these benchmarks while classifying the objects in real- time. Lewis, in his paper, proposed a DIY network called as Simple Net, that per- forms deep object recognition without pre-processing or deep evaluations that are otherwise very costly. Though the accuracy is quite less compared to the state-of-art, Simple Net looks to draw power from appropriate loss functions with finite number of parameters while other networks draw power from the depth of the layers. The author compared various CNN models such as Over Feat, VGG16, Fast R-CNN, YOLO with Simple Net to give the audience a profound insight into all these CNN models in terms of performance. feature extraction and an expensive per-region computation which are the first and second stages in the Faster R-CNN network, hinder the speed of the network. Addressing this issue, Kim et al made changes in the feature extraction stage by utilizing the cutting-edge technical innovations and presented a newer net- work known as PVANET. This network is capable of detecting objects from multiple categories with accuracy that is on par with its counterparts, while reducing the computationalcost. Dai et al, constructed a fully convolutional network called as R-FCN, while adopting the existing ResNet which are state-of-the-art when it comes to object detection. In an attempt to increase the object detection accuracy, the fully connected layers in Fast R-CNN have been replaced by a set of score maps that are position- sensitive as well as capable of encoding spatial information. As a result, R-FCN dis- played similar accuracy as that of Faster R-CNN but at better computational speeds. Kong et al presented a network called as HyperNet, which is capable of detecting objects at multiple scales by performing detection at multiple output layers. This network is similar to the MS-CNN, proposed by, which provides an efficient framework for detecting objects at multiple scales.
Liu et al presented a simple and straightforward network called as Single Shot multi-box Detector (SSD) which is capable of delivering real-time performance at high accuracy. This network does not utilize regional proposal method. In this network, the object localization and classification are performed in a single forward pass of the network while using a technique known as 'multi-box' for performing the bounding box regression. The SSD is hence capable of performing end-to-end computations.

Redmon et al, in their paper, presented YOLOv3 which is an updated version of their revolutionary network YOLO. This model surpassed all the other state- of- the-art networks such as Faster R-CNN, VGG-16, ResNet, etc., in terms of computational speed and accuracy, thus making it an ideal network for performing real-time detections and tracking while maintaining high accuracy which the other networks have failed to do. The YOLOv3 is also capable of detecting objects of small size as it can detect objects of three different scales effectively.

From the above papers, one can say that CNNs are the best suited deep learning algorithm for real-time object detection and recognition. From the knowledge gathered, it is evident that most of the research and development of autonomous driving systems is being implemented on transportation vehicles such as cars while only a little research is being carried-out to evaluate the existing state-of-the-art deep learning models and identify the best deep learning model for the detection and tracking of objects in the construction/excavation environments, as only little research has been carried out in this area of study, to date Therefore, this thesis will be using CNN models to recognize small scale vehicles at real-time to evaluate the performance of these algorithms and as a step towards future innovations.

VII.IMPLEMETATION

EXPERIMENTAL RESULTS

In this chapter, we begin discussing the experimental part of the thesis. First, we will discuss selection criteria for methods and datasets. Then we will describe the selected methods, their parameters and the selected datasets. Finally, we will discuss post- processing and evaluation. The implementation of the methods is mostly discussed in the following chapter. However, some implementation details are also discussed in this chapter, since they influence method selection.

RESEARCH QUESTIONS

As discussed in the earlier sections, the overall goal of this research is to identify suitable and highly efficient deep learning models for real-time object recognition and tracking of construction vehicles, evaluate the classification performance of the selected deep learning models and finally, to compare the classification performance of the selected models among each other and present the results.

The following re- search questions have been formulated to fulfill these research objectives.

- RQ1:What are the most suitable and efficient Deep Learning models for real- time object recognition and tracking of construction vehicles?

--Since there are several deep-learning models that can perform object detection and recognition, this research question has been formulated to identify the best suited deep learning models for performing object recognition of construction vehicles in real-time, so that it would be useful in future projects in developing intelligent machine navigation systems, autonomous driving of heavy machinery as they require object recognition to be done in real-time with high accuracy.

- RQ2:How is the classification performance of the Deep Learning models that are selected for object recognition?

--This research question has been formulated to evaluate the classification performance of the selected deep-learning models using relevant metrics so as to compare the results among each other.

LITERATURE REVIEW

To answer the RQ1, literature review has been selected as the research method. The literature review is selected in order to gain knowledge and deep understanding about various deep learning models and their efficiency so that the most suitable and efficient method can be selected from the identified models.

SEARCH PROCESS

The main focus of the search process was to find all the papers in which "Object detection" and "Deep Learning" have been mentioned. Therefore, search strings such as "Object detection AND Recognition AND Deep Learning" have been created for the search process. The search process has been carried out on IEEE Xplore, Springer Link and ACM Digital Library databases. The papers have been selected following the inclusion and exclusion criteria discussed in the subsection 3.2.2. The selected research papers have been filtered by reading the title of the collected articles, followed by reading the abstract of the articles filtered from the previous stage and ultimately, by reading the entire text of the articles that were selected from the previous stage.

Inclusion and Exclusion Criteria

The following inclusion and exclusion criteria have been followed while collecting the articles for the literature review:

Only those articles that discussed about object detection/recognition and deep learning models have been included.

- Only the articles published between the years 2009 and 2019 have been included, as they reflect the most recent research conducted in this area.
- Only the journal articles, conference papers, magazines and reviews have been included.
- Only the articles written in English language have been included for understand-ability purposes.
- Abstracts and PowerPoint presentations have been excluded.

EXPERIMENT

An experiment has been selected as the research method to answer the RQ2. An experiment has been chosen as the research method because when it comes to dealing with quantitative data, experiment is considered to be the best method. The main goal of this experiment is to evaluate the deep learning models for object recognition and tracking of construction vehicles in real time, the deep learning models being the ones selected from the literature review.

System	Dell Precision 7710
GPU	NVIDIA Quadro M4000M
CPU	Intel Core i7-6820HQ
Installed Memory (RAM)	65536 MB
Display Memory (VRAM)	4053 MB
Operating System (OS)	Windows 10

Table 7.1: Hardware Environment

EXPERIMENTAL SETUP

- Hardware Environment: Hardware specifications of the system on which the algorithm has been trained and implemented.Prior to the commencement of the training process, the following steps have been completed that are essential for the training of the algorithm.

• Dataset Collection: A dataset has been created by collecting the images of the three types of vehicles- Hauler, Excavator and Wheeled Loader in various angles, brightness and contrasts. The collected images consisted of at least one of the three classes mentioned, alongside other objects in the PDRL lab.

A total of 1097 images have been collected, among which 250 each are the images of Hauler, Excavator and Wheeled Loader, and the remaining 347 images comprise of all the three objects of interest. As the scaled construction site is a constrained environment and has a limited scope, it resulted in the collection of limited number of images.

Since the rule of thumb for deep learning is to have a minimum of 1000 images per class in the dataset, data augmentation methods have been applied. Among several data augmentation methods like image panning, zooming, flipping, rotating, etc., the images in the dataset have been augmented by rotating them at 90, 180- and 270-degree angles, and also flipping them horizontally, thus multiplying the number of images in the dataset by a factor of 5, resulting in a dataset of 5,485 images. Out of these images, 1,250 each are the Hauler, Excavator and Wheeled Loader while the remaining 1,735 images comprise of all the three vehicles. Additionally, a test video has also been included in the test dataset, as False Negatives for each of the algorithms can only be yielded using a test-video as it contains certain frames where there are no objects of interest present while it has been ensured that the images had at least one object class present in them.

• Data Pre-processing: For the Faster R-CNN, it has been ensured that all the images are of dimensions 608 x 608 before feeding into the network. The size of the input image depends mainly on the backbone convolutional neural network that the image is being fed into. It is suggested that the input image must be resized in such a way that the shorter side of the image is around 600px while the other side is no greater than 1000px. While for the YOLOv3 and Tiny-YOLOv3, it has been ensured that all the images are resized into416 x 416 before feeding into the network.

Though the YOLO is unaffected by the size of the input image, it is suggested that a constant input size is maintained throughout the dataset as problems might creep up later during the implementation of the algorithm. Additionally, since an input size of 416 x 416 provides ideal results of accuracy and speed and is widely followed by various practitioners, the images in the dataset have been resized accordingly.

• Dataset Labelling: The images collected in the dataset have been labelled manually using a tool known as 'Label Img'. Each image is labelled by drawing bounding boxes perfectly surrounding the desired objects in the image and selecting their respective classes, as shown in the Figure3.1. As a result, an XML file, also known as 'Annotation file', is generated for each image and saved into a specific folder. The annotation files contain details about the objects in the image such as Image name and label name, Image path, class name of the object(s), coordinates of the bounding boxes surrounding the

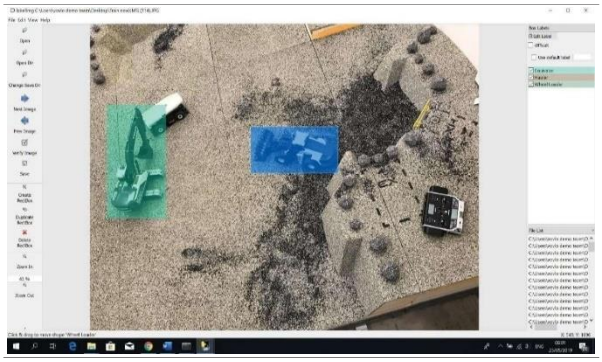


Figure 7.1: Dataset Labelling

objects present in the image. These files are further used to train and enable the algorithm to detect the desired class objects.

• Framework: TensorFlow's Object Detection API is identified to be a powerful tool, as it enables us to build and deploy image recognition software quickly. Hence it has been selected to train Faster R-CNN in this thesis.

Configuration: Various changes have been made to the default configuration files of the Faster R-CNN provided by the TensorFlow Object Detection API, such as dataset, number of classes to be trained, batch size and label map. The dataset has been split into two parts train dataset, test dataset. The train dataset consisted of 80% images while test dataset consisted of 20% images from the original dataset, which is the general rule of thumb followed by various researchers while splitting the dataset. Number of classes to be trained is also changed to 3 classes, since the Faster R- CNN algorithm being trained is expected to detect and recognize three classes Wheel Loader, Hauler and Excavator. A label map has been created comprising of class names and their corresponding class ids.

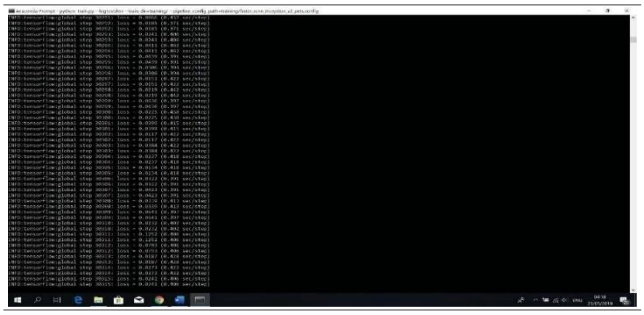
Batch size, which represents the number of train images that are used by the algorithm in one iteration, is also changed as it affects the VRAM consumption. Higher the batch size, greater is the VRAM consumption. Therefore, a smaller batch size has been selected to perform the training process.

ISSN (Online): 2456-0448
International Journal of Innovative Research in Management, Engineering, and Technology
Vol. 7, Issue 3, March 2022

TRAINING

After finishing all the steps mentioned above and making necessary changes in the configuration file, the training process is initialized. The step count and classification loss in each step can be seen on screen. It can be noted that the classification loss starts at a really high value and gradually decreases as the algorithm learns as the iterations progress. This has been visualized in the form of a graph, with the help of TensorFlow Board.In the 'Global step' represents the iteration or batch number that is being processed. 'Loss' value given is the sum of Localization loss and Classification loss. These represent the price paid for inaccuracy of predictions. The optimization algorithm keeps reducing the loss value until a point where the network is considered to be trained by the researcher. In general, lesser loss implies better training of the model.

Figure 7.2: Training & Classification loss



'Sec/step' is the time taken to process that corresponding step.

METRICS

The following metrics are used to evaluate the classification performance of the algorithm:

ACCURACY

It is defined as the number of correct predictions made by the model over the total number of predictions. This is a good measure, especially when the target variable classes are balanced in the data. This can be represented as

No. of correct predictions (CP) = True Positives + True Negatives Total no. of

Predictions (TP)= True Positives +True Negatives + False Positives

Accuracy=CP/TP

Where, a True Positive is defined as a correct detection of the object class trained. A True Negative is defined as a correct misdetection, meaning that nothing is being detected when there is no object that must be detected. A False Positive is defined as a wrong detection, meaning that there is a detection even though there is no object that must be detected. A False Negative is defined as a ground truth being not detected, meaning that the algorithm failed to detect an object that is required to be detected.

F1 SCORE

The balanced F-measure is used to measure a test's accuracy. The F1 score is considered to be good if the overall number of false positives and false negatives is low. It is defined as the harmonic mean of Precision and Recall.

$$F\,1\text{Score} = 2 * \frac{\text{Precision} * \text{Recall}}{(\text{Precision} + \text{Recall})}$$

Where Precision and Recall are defined as follows:

- Precision: It is defined as the number of true positive results divided by total number of positive results predicted by the classifier.

$$\text{Precision} = \frac{\text{True positives}}{\text{True Positives} + \text{False Positives}}$$

- Recall: It is defined as the number of true positive results divided by the sum of true positives and false negatives.

$$\text{Recall} = \frac{\text{True positives}}{\text{True Negatives} + \text{False Negatives}}$$

EXPERIMENT

The results obtained by Faster R-CNN, YOLOv3 and TinyYOLOv3 algorithms have been tabulated. The representation of the results obtained by the algorithms after evaluating every frame in the test video.

ACCURACY

The number of true positives, true negatives, false positives and false negatives that have been obtained by the models on the test video have been presented in the Table.

Table 7.2: Calculated Values of the Algorithms

Algorithm	**Faster R-CNN**	**YOLOv3**	**Tiny-YOLOv3**
True Positives	1133	1214	986
False Positives	44	39	56

From the values presented in the Table, the accuracy of the models has been calculated to be 84.07%, 89.51%, 74.05% respectively for Faster R-CNN, YOLOv3 and tiny YOLOv3 and it is visualized. The reason for the accuracy score of YOLOv3 being higher is because of its architecture where the object detections are performed at three different scales, making YOLOv3 more efficient in detecting smaller objects or detecting objects in difficult scenarios such as objects appearing partly in a certain frame. Since in certain scenarios, the objects are located on the farther side of the scaled site, they appear smaller.

F1 SCORE

Since the F1 score of a model is defined as the harmonic mean of precision and recall, it is essential that they are calculated prior to the calculation of the F1 score.

Algorithm	**Faster R-CNN**	**Yolo v3**	**Tiny-YOLOv3**
True Positives	1133	1214	986
True Negatives	192	195	184
False Positives	44	39	56
False Negatives	207	126	354

Table 7.3: Calculated Values of the Algorithms

PRECISION

The precision of a model is dependent on the number of true positives and number of false positives. The number of true positives and false positives have been obtained by models on the test video.From the values presented in the Table, the precision of the models has been calculated as 0.9626, 0.9688, 0.9462 respectively for Faster R-CNN, YOLOv3 and Tiny-YOLOv3. The precision of YOLOv3 is higher that Faster R-CNN and Tiny- YOLOv3 is because, its predictions are very precise as it can detect at three different

Figure 7.2: Graphical representation of F1 score

ISSN (Online): 2456-0448
International Journal of Innovative Research in Management, Engineering, and Technology
Vol. 7, Issue 3, March 2022

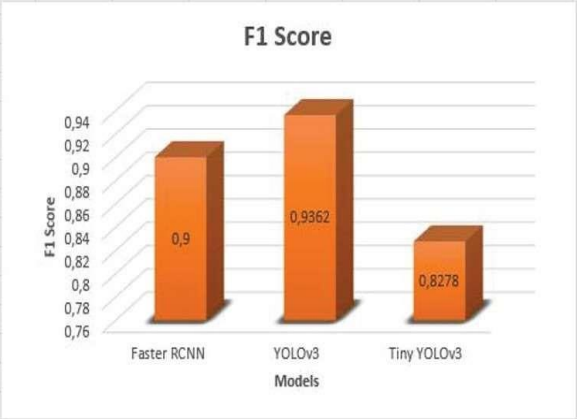
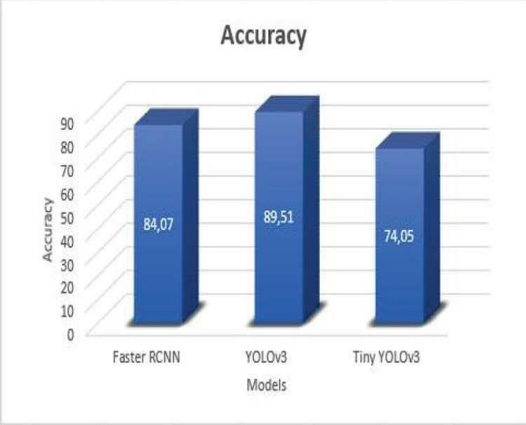


Figure 7.3: Graphical Representation of Accuracy



Therefore, it can be concluded that the precision of
scales, whereas Faster R-CNN and Tiny-YOLOv3 struggled to show correct prediction where the size of the object is considerably small. YOLOv3 in real-time detection and tracking of the construction vehicles is really good.

RECALL

The recall of a model is dependent on the number of true positives and number of false negatives. The number of true positives and false negatives have been obtained by the models on the test video.

Algorithm	**Faster R-CNN**	YOLOv3	**Tiny-YOLOv3**
True Positives	1133	1214	986
False Negatives	207	126	354

Table 7.4: Calculated Values of the Algorithm

From the values presented in the Table, the recall of the models has been calculated as 0.8455, 0.9059, 0.7358 respectively for Faster R-CNN, YOLOv3 and tiny YOLOv3. The recall values for Faster R-CNN and Tiny-YOLOv3 is lower than YOLOv3 as they have shown incorrect detections in many frames where the object is farther away or the size of the object is smaller, while YOLOv3 provided better results. Therefore, it can be concluded that the recall of YOLOv3 in real-time detection and tracking of the construction vehicles is really good compared to other models selected.

Using the Precision and Recall values obtained from the previous steps, the F1 score of the Faster R-CNN, YOLOv3 and tiny YOLOv3

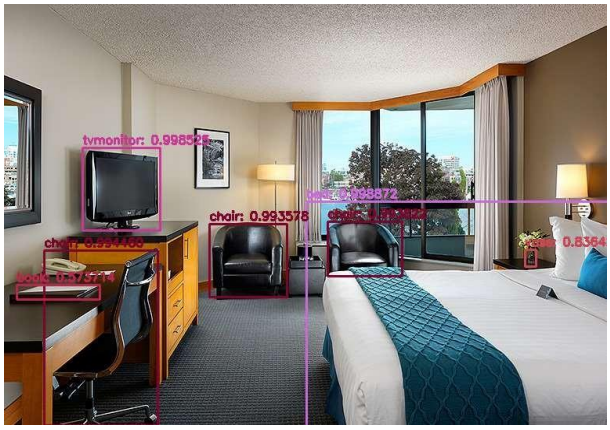
models has been calculated as 0.9, 0.9362 and 0.8278 respectively and it is visualized. Since the performance of the model is directly proportional to the F1 score and the upper limit of the F1 score being 1, it can be said that the performance of YOLOv3 model in real-time detection and tracking of the construction vehicles at the scaled site is higher than other models used in this experiment.

VIII.SAMPLE CODE

```
import cv2
import argparse
import numpy as np
ap = argparse.ArgumentParser()
ap.add_argument('-i', '--image', required=True,help = 'path to input image')
ap.add_argument('-c', '--config', required=True,help = 'path to yolo config file')
ap.add_argument('-w', '--weights', required=True,help = 'path to yolo pre-trained weights')
ap.add_argument('-cl', '--classes', required=True,help = 'path to text file containing class names')
args = ap.parse_args()
def get_output_layers(net):
 layer_names = net.getLayerNames()
 output_layers = [layer_names[i[0] - 1] for i in net.getUnconnectedOutLayers()]
return output_layers
def draw_prediction(img, class_id, confidence, x, y, x_plus_w, y_plus_h):
label = str(classes[class_id])
 color = COLORS[class_id]
cv2.rectangle(img, (x,y), (x_plus_w,y_plus_h), color, 2)
cv2.putText(img,label,(x-10,y-10), cv2.FONT_HERSHEY_SIMPLEX, 0.5, color, 2)
image = cv2.imread(args.image)
Width = image.shape[1]
Height = image.shape[0]
scale = 0.00392
classes = None
with open(args.classes, 'r') as f:
 classes = [line.strip() for line in f.readlines()]
COLORS= np.random.uniform(0, 255, size=(len(classes), 3))
net = cv2.dnn.readNet(args.weights, args.config)
blob = cv2.dnn.blobFromImage(image, scale, (416,416), (0,0,0), True, crop=False)
net.setInput(blob)
outs = net.forward(get_output_layers(net))
class_ids = []
confidences = []
boxes = []
conf_threshold = 0.5
nms_threshold = 0.4
for out in outs:
for detection in out:
scores = detection[5:]
class_id = np.argmax(scores)
confidence = scores[class_id]
if confidence > 0.5:
center_x = int(detection[0] * Width)
center_y = int(detection[1] * Height)
        w = int(detection[2] * Width)
        h = int(detection[3] * Height)
        x = center_x - w / 2
        y = center_y - h / 2
        class_ids.append(class_id)
        confidences.append(float(confidence))
        boxes.append([x, y, w, h])
indices = cv2.dnn.NMSBoxes(boxes, confidences, conf_threshold, nms_threshold)
```

```
for i in indices:
    i = i[0]
    box = boxes[i]
    x = box[0]
    y = box[1]
    w = box[2]
    h = box[3]
    draw_prediction(image, class_ids[i], confidences[i], round(x), round(y), round(x+w), round(y+h))
cv2.imshow("object detection", image)
cv2.waitKey()
    cv2.imwrite("object-detection.jpg", image)
cv2.destroyAllWindows()
```

IX.SNAPSHOTS



CONCLUSION

Since the images for YOLOv3 have been resized into 608 x 608, while for the Faster R-CNN and Tiny-YOLOv3 the images have been resized into 416 x 416, this might be taken into consideration as an internal validity threat. Errors might occur while collection of the data and calculation of metrics, which are a threat to validity. This threat is mitigated by taking down the readings and confirming multiple times. Important data used in the results such as classification loss, number of iterations, etc. are taken from the Tensor Board and Darknet to avoid any errors.

It has been ensured that the train and test dataset have been properly separated in such a way that no clips from the train dataset is present in the test dataset. Even though the algorithms have been trained using the data obtained from scaled site provided at PDRL, the algorithms have also been tested on real world images taken at the construction sites and the results have also been presented evaluation. In this, conclusion validity is avoided by selecting suitable metrics for evaluating the selected algorithms as well as following proper steps for conducting are search.

REFERENCES

[1] R. Philippe, Volvo ExcavationSite, 2020. [Online].Available:https: //www.korestudios.com/portfolio/volvo-construction-equipment/

[2] AHK, Exemplar Construction Site,2018.[Online].Available:https://urbantoronto.ca/news/2018/04/torontos-largest-construction- site-well-spadina-front

[3] V. G. Maltarollo, K. M. Honório, and A. B. F. da Silva, “Applications of artificial neural networks in chemical problems,” Artificial neural networks-architecturesand applications, pp. 203–223, 2013.

[4] TutorialsPoint,SupervisedLearning,2020. [Online]. Available: https://www.tutorialspoint.com/artificial_neural_network/artificial _neural_network_supervised_learning.htm

[5] B. Frank, Deep Learning the Beautiful Mind, 2016. [Online]. Available: www.mindwise-groningen.nl/deep-learning-the-beautiful- mind/
[6] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional net- works," in European conference on computer vision. Springer, 2014, pp. 818– 833.
[7] P. Firelord, Pictorial example of max-pooling, 2018. [Online]. Available:https://computersciencewiki.org/index.php/Maxpooling_/_Pooling
[8] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Uni- fied, real-time object detection," in Proceedings of the IEEE conference on com- puter vision and pattern recognition, 2016, pp. 779–788.
[9] CyberAILab,ACloserLookat YOLOv3, 2018. [Online]. Available: https://www.cyberailab.com/home/a-closer-look-at-yolov3
[10] C. C. Nguyen, G. S. Tran, T. P. Nghiem, N. Q. Doan, D. Gratadour,
J. C. Burie, and C. M. Luong, "Towards real-time smile detection based on faster region convolutional neural network," in 2018 1st
[11] Z. Deng, H. Sun, S. Zhou, J. Zhao, L. Lei, and H. Zou, "Multi-scale object detection in remote sensing imagery with convolutional neural networks," ISPRSjournal of photogrammetry and remote sensing, vol. 145, pp. 3–22, 2018.
[12] D. Xiao, F. Shan, Z. Li, B. T. Le, X. Liu, and X. Li, "A target detection model based on improved tiny-yolov3 under the environment of mining truck," IEEE Access, vol. 7, pp. 123 757–123 764, 2019.
[13] B. Volodymyr, Machine Learning Algorithms: 4 Types You ShouldKnow,2018.[Online].Available:www.theappsolutions.com/blog/development/ machine-learning- algorithm-types/
[14] S. Ray, Essentials of machine learning algorithms (with python and rcodes),2017.[Online].Available: http://www.analyticsvidhya.com/blog/2015/ 08/common-machine- learning-algorithms
[15] C. M. Bishop, Pattern recognition and machine learning. springer, 2006.
[16] J. Aaron, Everything You Need to Know About Artificial Neural Networks, 2015. [Online]. Available: www.medium.com/technology- invention-and-more/ everything-you-need-to-know-about-artificial-neural networks-57fac18245a1

```
import cv2
imp
image
cv2.imshow("object detection", image)

cv2.imwrite("object-detection.jpg", image)
cv2
```

conducting a researc and by seecting inappropriate metrics for ives