A COMPARISON OF DEEP LEARNING METHODS FOR OBJECT IDENTIFICATION IN AUTONOMOUS OUTDOOR CLEANING ROBOT

Mohammed Raheem P
CET Center for Interdisciplinary Research
College of Engineering Trivandrum
Kerala, India
mohammedraheemp@gmail.com

Dr. Sreeni K.G
Dept of Electronics and Communication Engineering
College of Engineering Trivandrum
Kerala, India
sreenikg79@gmail.com

***Abstract*—This paper presents a technique for automatic identification of waste particle using a deep learning framework. The objective of the work is to identify the waste particles such as plastic covers, waste papers, leaf etc for the functioning of an outdoor cleaning robot. Use of visual identification system in the cleaning robot optimizes the power consumption during the cleaning process. Classifica- tion of waste particles and obstacles using Keras model and YOLO mark detection methods are compared in this work. Both methods are trained using deep learn- ing architecture. Even though the training process using YOLO bounding box is time consuming, the accuracy obtained for YOLO based classification is very high. The obtained accuracy for YOLO mark detection method is 89% with an input of 2000 testing images; make it useful for waste particle identification in an outdoor cleaning robot.**

***Index Terms*—Keras, Tensorflow, Convolutional Neural Network, Deep Learning, YOLO, Bounding Box, Darknet, IoU**

I. Introduction

The major difference in the outdoor cleaning compared to indoor cleaning using autonomous vehicles is its complexity because of its high power consumption during the cleaning tasks. This can be reduced to a great extend by using the visual identification of waste particles in the outdoor environments. Using visual identification, this system can be used to clean the surface of floor area having waste particles instead of cleaning everywhere. Thus the power loss can be optimized and more area can be cleaned. Visual system can also be used to avoid obstacles by detecting different classes of obstacles like trees, stones, timbers etc. This paper introduces different methods to classify the objects and obstacles by visual object detection methods. Object classification is one of the challenging application in computer vision. Deep learning based classification gives more accurate results compared to other detection methods. This report discusses two Deep learning frameworks for object identification and the comparison between them. First method is object identification using keras and tensorflow . In this software framework method, dataset of images around 15000 are used for each classes (objects and ground alone) are trained using keras model and tensorflow backend. This modeled dataset can train by deep learning algorithm such as convolutional neural network. After learning process or training process, a weight file that contains types of classes and types of images are generated. Using this weight file new unseen image classes can be predicted. Predicted result shows the probability of each trained classes. This value can be used (by setting a threshold) for the outdoor cleaning robot to do further actions or cleaning actions.

Each video is formed by a number of frames and hence from the real time videos frames of images can be obtained using which object detection be done in autonomous ground vehicles or AGV system. For comparison of Deep Learning methods for object identification in Autonomous Outdoor Cleaning Robot, data was prepared by a collection of images obtained from the outdoor environments of college campuses, roads, hostel courts etc. Python tool was used to convert videos into frames. After preparing the dataset, a deep learning model using keras is created and supported with a backend of tensorflow to do the execution. Finally, the prepared model dataset moves to the training process of convolutional neural network which contains number of hidden layers. The accuracy of prediction highly depends on number of hidden layers or number of hidden layer is directly proportional to the accuracy of outputs. 8 hidden layers have been used in this project. The problems of keras model based methods is that it does not give any information of location of objects as well as the obstacles. Another deep framework of YOLO based approach contains bounding boxes of every classes can also locate the obstacle position which helps in the motion of the vehicle. YOLO object detection algorithm is difficult to create, but it needs only few input images as compared to the first method. Here, the dataset of objects, obstacles and the combined objects are used. All input images need to be marked using bounding boxes. Every bounding box marking provides information of classes and locations as well. After marking the objects the dataset model is trained to create weight files. Similar to the first method, using weight file can predict the bounding boxed outputs of unseen images. This can be done for real time video as well. Accuracy of YOLO based approach is higher than

the Keras based model, proving it better for outdoor cleaning robot.

II. SYSTEM DESCRIPTION

In this system Raspberry Pi is used as the controller which can interface camera and also image processing can be done with high processing speed. Using Pi camera, real time video capturing can be achieved which includes the number of frames and thus can check whether it contains any objects or waste particles using deep learning algorithm. When the frame detects the waste particles then the Pi passes the signal to the motor controller for the cleaning tasks using brush motors. Ultrasonic sensor used to detect the obstacles and can also measure the obstacle distance from the moving vehicle hence can avoid obstacles easily. Wheel encoder are also connected to Pi which gives the speed of wheels and to know how much distance have been covered by the vehicle from the starting position.

Cleaning vehicle is mainly divided into 3 sections.

- Mechanical design of vehicle.
- Path planning for vehicle movement.
- Object detection and identification.

A. Mechanical Design of Vehicle

Cleaning robots find great use in the society since it reduces fatigue to human as well as reduces the time consumed for the purpose. Inspired from the India Government mission of Swachh Bharath Adiyan, which is basically targeting at keeping our environment clean and tidy, a low cost autonomous cleaning robot that could clean our streets and paths with the prior knowledge of the environment could serve a better nation.

Two powered wheels are used that could traverse in any terrain and are differentially driven. The two wheels (Figure 1) on the front are castor wheels which follow the motored rear wheels. 24 kg-cm torque motors are used for the purpose and so they could carry heavy weights. The chassis is been made with aluminum and has a ground clearance of 25 cm. Two round brushes that rotates in opposite directions collects the debris and directs it into another horizontal brush placed behind it. This horizontal brush directs the debris into a waste bin placed at its immediate back which can be dumped by manually controlling the servo motors attached to the waste bins.

B. Path Planning

The robot vehicle is deferentially driven type, where the both the wheels are separately driven with different velocity commands. The path planning section (Figure 2) consist of development of the path planning algorithm and also tracking of the vehicle. Complete coverage of a given area can be done using different type of patterns like random walk, spiral, S shaped or boustrophedon and wall following. Most commonly used one is S shaped pattern because of its simplicity and it takes comparatively less time to complete the task.

Fig. 1: A typical model of a cleaning vehicle.

The work space is divided into girds consisting of square cell each representing the vehicle. Covering each cell is assumed as complete coverage and cleaning of that particular area. The vehicle will be designed to cover from one cell to another. For bigger and complicated maps, it will be decomposed into smaller maps using boustrophedon decomposition method and divides the whole area into divisions. After covering one particular division of the work space it moves to the next division taking the shortest path to the available point in that division or cell. This enables easy and fast coverage of the area. An ultrasonic sensor is attached to the front according to which the speed control of the vehicle is been obtained. Also for obstacle detection ultrasonic sensors is been used. The algorithm can be applied to workspace in presence of different obstacles with the prior knowledge of the environment. Figure

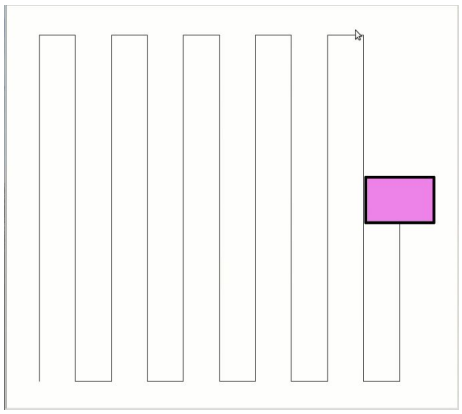


Fig. 2: Path plan proposed to cover a rectangular area.

2 shows the path taken by the robot in an environment without any obstacles.

C. Object Detection and Identification

Optimization of power consumption is the main aim of this project by using camera based approach. Raspberry Pi controlled camera is used for capturing real time video while running our vehicle through its allotted path. A Raspberry Pi-Camera module V2 of 8 Megapixel used for real time detection is shown in Figure 3.

Before implementing on the final vehicle system, a prototype of the vehicle was made which is a differential driven

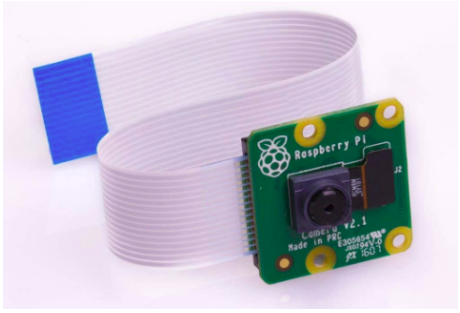


Fig. 3: A Pi-Camera used in the cleaning vehicle.

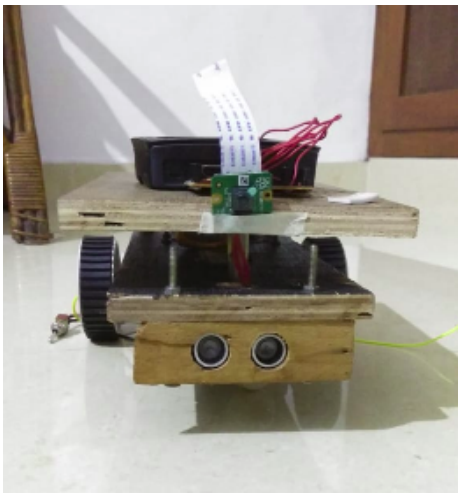


Fig. 4: A Pi-Cam mounted on a mobile robot.

mobile robot. It was attached with the Raspberry PI Camera Module V2 (Figure 4) to check whether the vehicle is detecting the waste particles while moving times. This 8MP camera module is capable of 1080p video and still images and connects directly to Raspberry Pi. When the camera detect the waste particles ,then a control signal is passes to motor controller to clean the surface of ground. Pi camera is better than the USB camera for the faster and better processing. Neural network based projects are generally used as Pi camera instead of using USB cam.

The testing of the method was first started in a prototype of the vehicle to check whether the entire dataset is perfect and also to find the proper architecture of the convolutional neural network with real time video capturing. After checking in the prototype, it was also implemented in the final vehicle with same devices and setups. Collected the dataset of images containing waste particles as well as the obstacles to train or learn the model. Detailed object identification steps is been discussed in the next chapters.

III. PROPOSED METHODOLOGY

In this section details of the two different methods that have been used for object identification tasks is given. First method represents Keras-Tensorflow model which is software based object classification and second method represents the YOLO mark object detection method for graphical marking. Both are Deep Learning based learning process.

A. Object Classification Using Keras and Tensorflow

Keras is a Python deep learning library used for high level neural network API which can be run on top of Tensorflow, Theano, CNTK. The main advantages of Keras model for deep learning are that which can be runs in both CPU as well as GPU and which can be used in Convolutional Neural Network as well as Recurrent Neural Network or both and which prototyping allows very easily. Keras is the best model to reduce the cognitive load and user action numbers which is the consistent model and user errors can easily cleared by actionable feedbacks. To create a new model using combined layers, cost functions, optimizers, activation functions, regularization etc they are stand alone modules. The simplest type of keras model is the Sequential model which can be stack the layers linearly.

Tensorflow is an open source machine learning library offered by Google used for tensor manipulation framework or dataflow programming. Tensorflow backend is used in every execution of codes for the creating model and training dataset and also for prediction purposes from the frames as well as the video. Tensorflow provides some versions. The short version is flexible with all GPUs, CPUs, servers etc when programming without any change of lines of codes. It is an excellent method for training Deep Learning neural network as compared with the Theano and CNTK. Convolutional Neural Network architecture is used as Deep learning algorithm here.

1) Convolutional Neural Networks (ConvNets): Generally there are four steps in ConvNets (Convolutional neural network)

- Convolution
- Subsampling
- Activation
- Fully connected

Figure 5 represents the generally used architecture for a convolutional neural network.

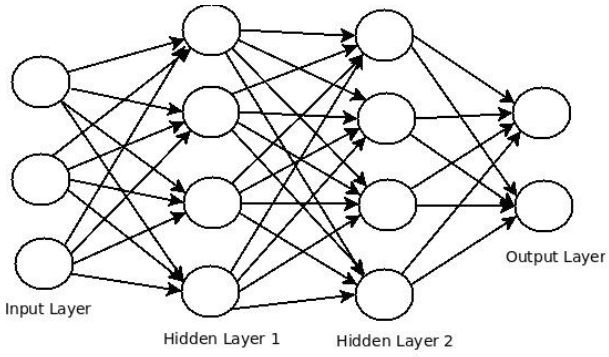


Fig. 5: A typical architecture of a CNN.

The first layer that receive inputs are called convolution filters or kernals. In this classification problem, the object reference signal will be convolved with input signal if

the input looks like previous object image class and the output signal moves to the next layer. The signal strength of output does not depends only whether the feature is present but does not on its location. So that object present at different locations in the image will be detected and recognized by the convolution neural network.

Length or size of the Convolved Feature or feature map is controlled by three variables and are Depth, Stride and Zero-padding.

- Depth: Depth is the number of filters used for the convolution process. Every filters producing different feature maps. We can stack all feature maps in 2D matrices.
- Stride: Stride is the number of units we slide filter metrics over the input image matrix. If stride is 2, then filter moves one pixel to two units forward as well as in the downward direction. Through this we can reduce the feature maps size.
- Zero-padding: This is used for controlling feature map size by adding zeros around the border of input matrix. This method is also called as wide or same convolution, and the one which does not use zero-padding are called as narrow convolution.

Subsampling is the process of smoothing inputs to reduce the noise and variations. Subsampling leads to reduction of image size and as well as the reduction of color contrast on the RGB channels.

Activation controls the signal flow from one layer to next layer. CNN provides several types of activation functions. Here ReLu activation function is been used as shown in Fig. 6 since training process based on ReLu activation process is very fast as compared with the others.

$$f(X) = max(0, X) \quad (1)$$

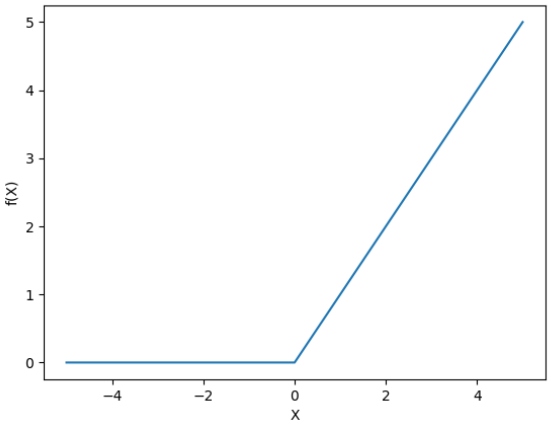


Fig. 6: ReLu activation graph.

Fully connected layer is the last layer of CNN. It is also called as Dense layer. A layer is said to be fully connected layer or dense layer when neurons of preceding layers are connected to each and every neuron in the subsequent layers.

2) Perceptron: A mathematical representation of neurons called a Perceptron.

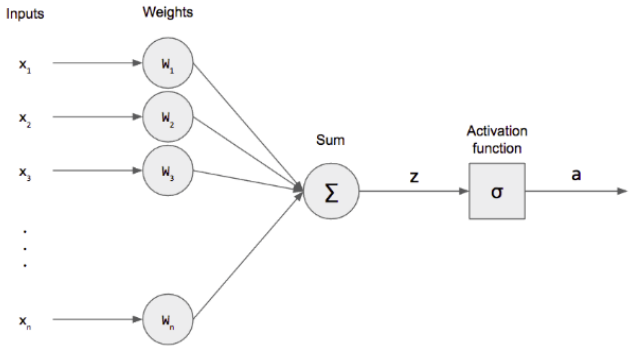


Fig. 7: Graphical representation of a neuron.

From Fig 7, inputs $x1, x2, ...xn$ when multiplied with corresponding weights $w1, w2, ...wn$ and summed together with the bias variable b gives the function f ,

$$f = (x1 \times w1) + (x2 \times w2) + ...(xn \times wn) + b) \quad (2)$$

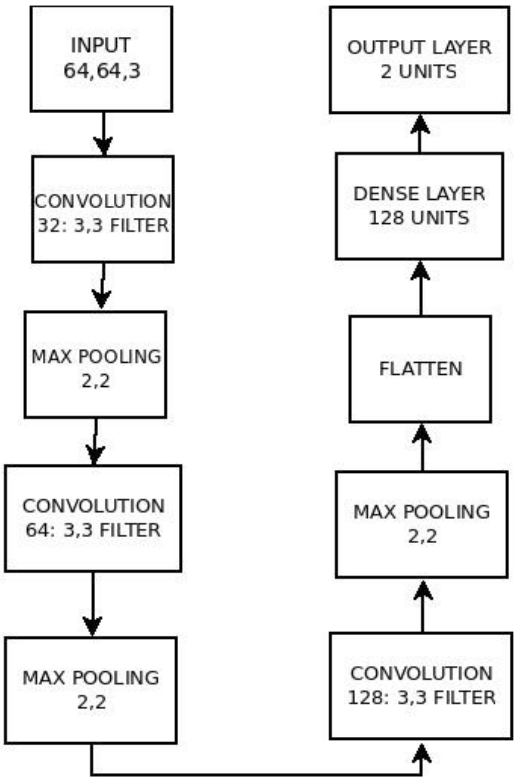


Fig. 8: Flow of operations in different layers in a convolutional neural network.

3) Layers used to Build ConvNet: Convolutional neural networks or ConvNets is a sequence of layers which passes outputs from a layer to the another layer through a function called differentiable function as shown in Figure 8. Mainly there are three types of layers Convolutional Layer, Pooling Layer, Fully Connected Layer. Format of layers is as shown below.

INPUT - CONV - RELU - POOL - FC

- INPUT Layer : Input Layer $64 \times 64 \times 3$ which represents the raw pixel values of input images of Width-32, Height-32, 3 RGB channels.
- CONV Layer : This layer calculates the output of neurons by multiplying with corresponding weights and adding with biases. In ConvNet the filtering matrix(containing the pixel value)is put on the input image matrix and is convoluted with corresponding pixels and then slide filter window over the input image. The convolutional layer consist of two steps i.e convolution and activation function. In the convolution operation the convolution with the filters or with the kernals is carried out. It include multiplication of image matrix by the corresponding filter matrix which gives an feature map as its ouput. The size of of output feature map depends on the factors like size of input image matrix, filter size, stride and zero padding. For an input layer is accepted at a volume of [$W1 \times H1 \times D1$] where $W1$ is the size of input width and $H1$ is the size of input height and $D1$ is the depth of input, the output volume obtained will be of [$W2 \times H2 \times D2$] where the $W2$, $H2$, $D2$ are the respective output dimensions called as the feature map dimensions and are given by the equations below.

$$W2 = \frac{W1 + 2P - F}{S} + 1 \tag{3}$$

$$H2 = \frac{H1 + 2P - F}{S} + 1 \tag{4}$$

*-

$$D2 = K \tag{5}$$

Where
F : Filter size
K : Number of filters
P : Zero padding,for controlling the output volume
S : Stride,which defines how much units can move filter over the image window
P and S are the hyper parameters
- RELU Layer : It is used to apply activation function and has a range from 0 to x ($max(0, x)$). This layer minimizes the network size and is also easy to train without errors. This layer does not change the size of layer. There are other types of activation functions such as sigmoid, softmax, tanh, elu etc but RELU is the most commonly used activation function especially when used for image classification purposes.
- POOL Layer : This layer performs a downsampling operation in dimensions of width and height. Generally used pooling operation function are MaxPooling and AveragePooling. Binary classification problems mainly uses maxpooling functions. For the downsampling operation along the spacial dimensions of width and height between the successive convolution layers pooling is used. The main purpose of pooling operation in the convolution layer is in reducing the amount of parameters and also optimization of the spacial size. It can update the dimensions [$W2 \times H2 \times D2$] by,

$$W2 = \frac{W1 - F}{S} + 1 \tag{6}$$

$$H2 = \frac{H1 - F}{S} + 1 \tag{7}$$

$$D2 = D1 \tag{8}$$

The output of first pooling layer moves to the input of the second convolution layer [9]. ioThe output of last fully connected layer dimension will [$1 \times 1 \times C$], where C is the number of classes to be identify.

- FC Layer (i.e. fully-connected) : This layer computes the class scores, resulting in volume of size $1 \times 1 \times 2$, where each of the 2 numbers correspond to a class score, such as among the 2 categories of binary classifier. As with ordinary Neural Networks and as the name implies, each neuron in this layer will be connected to all the numbers in the previous volume.

ConvNets transform the original image layer by layer from the original pixel values to the final class scores. Only some of the layers contain parameters. In particular, the CONV/FC layers perform transfiguration that are functions of not only the activations in the input volume, but also of the parameters (the weights and biases of the neurons). On the other hand, the RELU/POOL layers will implement a fixed function. The parameters in the CONV/FC layers will be trained with Adadelta so that the class scores that the ConvNet computes are consistent with the labels in the training set for each image.

Pooling Operation : It is common to periodically insert a Pooling layer in-between successive Conv layers in a ConvNet architecture. Its function is to progressively reduce the spatial size of the representation to reduce the amount of parameters and computation in the network, and hence to also control overfitting. The Pooling Layer operates independently on every depth slice of the input and resizes it spatially, using the MAX operation. The most common form is a pooling layer with filters of size 2×2 applied with a stride of 2 downsamples every depth slice in the input by 2 along both width and height, discarding 75 per of the activations. Every MAX operation would in this case be taking a max over 4 numbers (little 2×2 region in some depth slice). The depth dimension remains

unchanged. An example of pooling operation shown in Figure 4.5

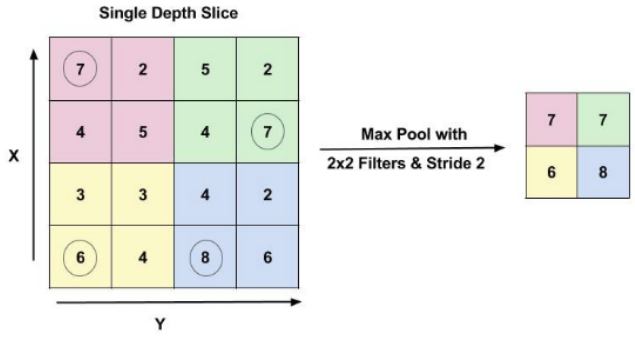


Fig. 9: MaxPooling operation.

Dropout Layer : Dropout layer is used to avoid overfitting problems.

B. Object Classification Using YOLO

You Only Look Once (YOLO) real time object detection algorithm used for marking bounding boxes of objects in images for training process to classify different objects in images. In this method, detection systems re-task classifiers or localizers to perform detection. Thus model apply to an image at multiple locations and scales. The region of high scoring in the image are considered detections. YOLO use a completely different approach. By applying a single neural network to the full image, this network divides the image into regions and predicts bounding boxes and probabilities for each region. These bounding boxes are weighted by the predicted probabilities. Our model has several advantages over classifier-based systems. It looks at the whole image at test time so its predictions are informed by global context in the image. It also makes predictions with a single network evaluation unlike systems like R-CNN which require thousands for a single image. This makes it extremely fast, more than 1000 $\times$ faster than R-CNN and 100 $\times$ faster than Fast R-CNN.

Now-a-days YOLO updates new versions of YOLO v2, YOLO v3 from YOLO v1. YOLO v1 termed as YOLO itself. Moving through the updated versions they are little slow on real time detection as compared with the previous but can more accurate and detect multiple objects as well as the small objects. Hence the updated versions are better, stronger but not faster. We need a fast detection for real time object detection so choose the oldest. Every Yolo versions can executed with or trained with Neural Network framework of Darknet.

1) Network Architecture: Convolutional Networks (ConvNets) are currently the most efficient deep models for classifying images data. In this networks first identify the low level features and try to learn then combine these features try to learn complicated patterns similar to the previous method of keras. Each layers represents number of neurons and each neurons presents with three dimensions of height, width and depth. ConvNets provides special kinds of architectures designed to recognize directly from different pixel images. These are LeNet, AlexNet, VGG, GoogLeNet, ResNet etc. One of the example for ConvNet model using AlexNet architecture shown in Figure 10

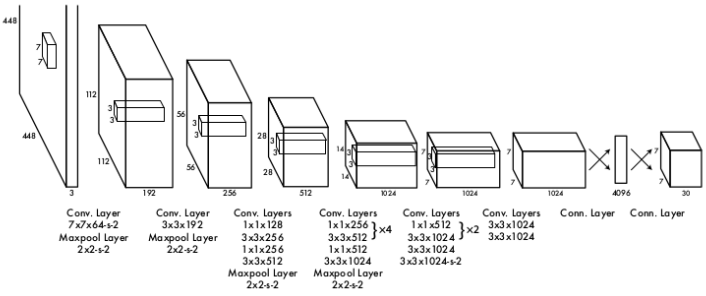


Fig. 10: Architecture of AlexNet.

From architecture model, first we set input data or images before moving to first convolution layer. In the first convolution layer provides two parts, they are convolutive part and Nonlinear activation function operation. Then the other part followed by Pooling operation [11].

2) Selection of Input Parameters during the Training: There is some parameters depending on training accuracy and training losses. They are

- Epoch: An epoch defines how much times the algorithm sees the entire data set. So, each time the algorithm has seen all images in the dataset, an epoch has completed.
- Iteration: An iteration defines how much times a batch of data or image passed through the modeled algorithm. In the case of our neural networks, which means the forward pass as well as the backward pass. So, every time pass a batch of data or image through our neural network that which completed an iteration.
- Batch Size: Total amount or number of examples for training present in a single batch. Batch size and number of batches are two different things. I set Batch=64 for this training process.
- Intersection of Union (IoU): Intersection over Union is an estimation metric used to find the accuracy of an object detector on a dataset for a particular task. Generally Intersection over Union used to evaluate the performance of object detectors such as HOG + Linear SVM and Convolutional Neural Network detectors such as YOLO, R-CNN, Faster R-CNN, etc. Intersection over Union is simply an estimation metric. Intersection over Union based evaluations are mainly used to get predicted bounding boxes in the output. To apply our Intersection over Union to a detector of object we must need two thinks. They are The ground-truth bounding boxes or the hand labeled bounding boxes from the testing set that define where our object in the image is and the predicted bounding boxes. Computing intersection over union for different can be determined in Figure 11

$$IoU = \frac{Area\ of\ Overlap}{Area\ of\ Union} \tag{9}$$

Fig. 11: An example of computing intersection over unions for various bounding boxes.

IV. EXPERIMENTS

This chapter discusses the implementation of the methodologies (previously discussed chapter) in my own classification problem. This follows Dataset creations for the keras model and YOLO mark. After creating dataset, we have to train or learn the dataset and classes using different methods of deep learning.

A. Experimental Dataset

Dataset creation or image collection is a difficult task in these experiments. With 13 MP RGB Camera, the videos of waste particles including obstacles are captured from different area or different environments. Then using Python tool, the video is converted into number of frames. The first method or keras model based approach uses 30000 input images (one class 15000 for object contained ground and other 15000 for ground alone) whereas YOLO uses 2000 marked images (mark every object classes in all input images by bounding box) which are marked manually by drawing bounding boxes.

B. Keras Model

This model collects huge number of images from different locations. This dataset provides two classes of objects. They are waste particles and ground alone images. Waste particles includes waste papers, covers, leafs etc as shown in Figure 12.

The ground alone images includes the outdoor roads,college campuses,college ground etc as shown in Figure 13

This class of ground alone images includes all types of outdoors that are free from any waste particles. It used 15000 images of ground alone from different environments. Addition of more environment images will improve the accuracy of results.

Figure 14 shows the flow chart illustrating different steps in waste identification using Keras-Tensorflow model. After collecting image or dataset, learning process or trained using convolutional neural network is performed. we need to predict new image classes including waste particles or not. Keras based approach provides probabilties of each classes for captured images. Using this probability, setting a threshold value to identify the captured class of image is done. set a threshold value of 0.5 to check the prediction results to identify the classes which are over the threshold or not. When the probability of ground class is greater than the threshold or $P(G) > 0.5$, the captured image shows the ground alone. Hence we can move forword for the cleaning areas. When the probability of waste particles greater than the threshold or $P(O) > 0.5$, image shows waste particles. Hence thus information can be passed to the controller for further actions of cleaning tasks.

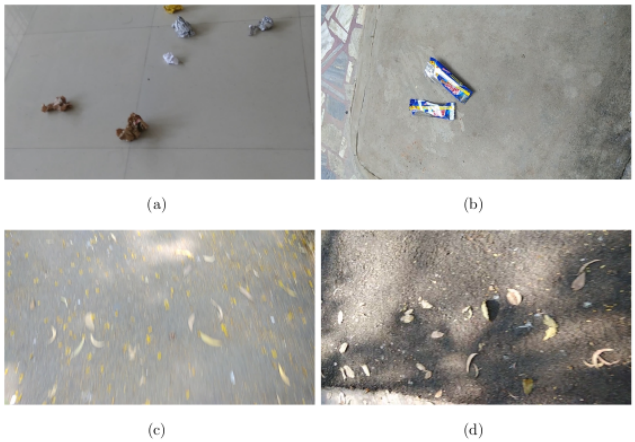


Fig. 12: Images of different waste particles used to train the system. (a) papers, (b) covers, (c) flowers, (d) leafs

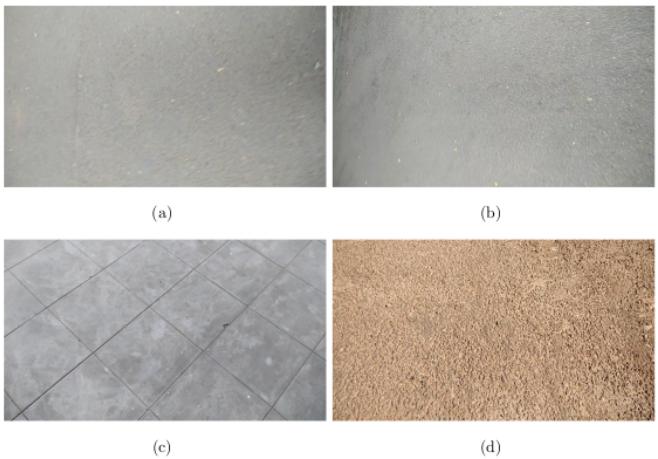


Fig. 13: Ground images from different locations. (a) Road, (b) college campus , (c) varanda, (d) college ground

C. YOLO Object Mark

In YOLO object marking process, we initially created a dataset of 1000 images of different objects and obstacles. We need to train all the images with marked images of objects and obstacles (Figure 15). So we need a text file for every image which contains all the details of location of objects and obstacles for the training process. This helps in understanding or easy learning of the object places or the detection of the objects[10].

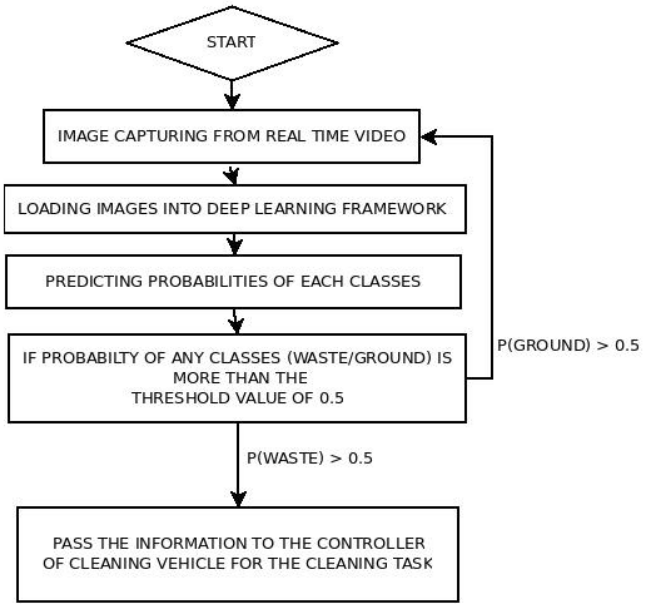


Fig. 14: Flow chart illustrating the prediction of results.

This work used Yolo mark master for the marking process of images with corresponding object classes. In the marking folder of yolo provides a marking command to run. Using this command we can mark every images to create the text file for further processes.

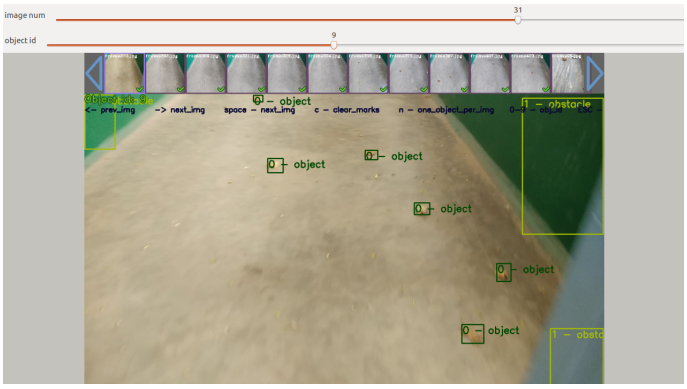


Fig. 15: Illustration of marking objects using YOLO.

After marking objects in every images which creates a text file automatically, defines locations of objects and obstacles. Using this informations from YOLO mark, we need to train our Deep learning framework. Flow chart of object detection using YOLO is shown in Figure 16.

D. Training using Darknet

Darknet is one of the open source Deep learning framework to do classification problems generally. Here Darknet is used to learn or understand the objects and obstacles from the images using convolutional neural networks.

After annotations of every images, we need to train the data to predict or detect unseen data. We need a copy of data

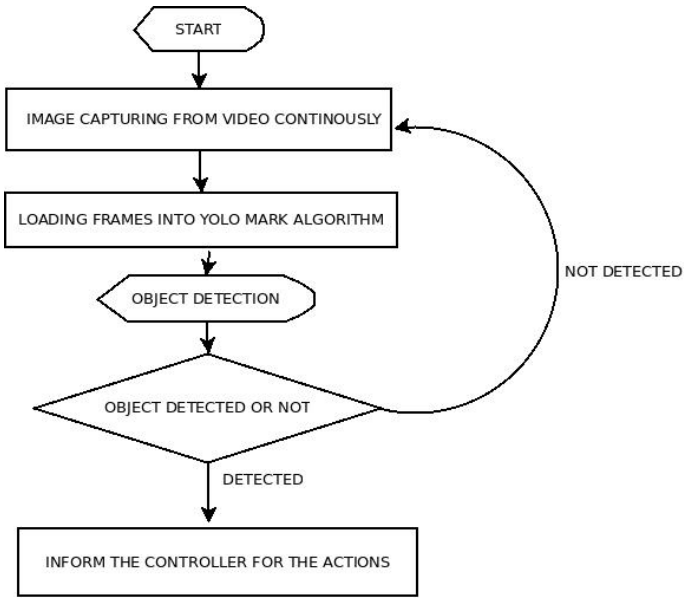


Fig. 16: Algorithm to get bounding box in predicted results.

from yolo mark master files to train the network. After that, a command is used to train the CNN with the darknet. The training process then carried out is shown Figure 17.

```
layer     filters    size              input                output
    0 conv     32  3 x 3 / 1   416 x 416 x   3   ->   416 x 416 x  32
    1 max          2 x 2 / 2   416 x 416 x  32   ->   208 x 208 x  32
    2 conv     64  3 x 3 / 1   208 x 208 x  32   ->   208 x 208 x  64
    3 max          2 x 2 / 2   208 x 208 x  64   ->   104 x 104 x  64
    4 conv    128  3 x 3 / 1   104 x 104 x  64   ->   104 x 104 x 128
    5 conv     64  1 x 1 / 1   104 x 104 x 128   ->   104 x 104 x  64
    6 conv    128  3 x 3 / 1   104 x 104 x  64   ->   104 x 104 x 128
    7 max          2 x 2 / 2   104 x 104 x 128   ->    52 x  52 x 128
    8 conv    256  3 x 3 / 1    52 x  52 x 128   ->    52 x  52 x 256
    9 conv    128  1 x 1 / 1    52 x  52 x 256   ->    52 x  52 x 128
   10 conv    256  3 x 3 / 1    52 x  52 x 128   ->    52 x  52 x 256
   11 max          2 x 2 / 2    52 x  52 x 256   ->    26 x  26 x 256
   12 conv    512  3 x 3 / 1    26 x  26 x 256   ->    26 x  26 x 512
   13 conv    256  1 x 1 / 1    26 x  26 x 512   ->    26 x  26 x 256
   14 conv    512  3 x 3 / 1    26 x  26 x 256   ->    26 x  26 x 512
   15 conv    256  1 x 1 / 1    26 x  26 x 512   ->    26 x  26 x 256
   16 conv    512  3 x 3 / 1    26 x  26 x 256   ->    26 x  26 x 512
   17 max          2 x 2 / 2    26 x  26 x 512   ->    13 x  13 x 512
   18 conv   1024  3 x 3 / 1    13 x  13 x 512   ->    13 x  13 x1024
   19 conv    512  1 x 1 / 1    13 x  13 x1024   ->    13 x  13 x 512
   20 conv   1024  3 x 3 / 1    13 x  13 x 512   ->    13 x  13 x1024
   21 conv    512  1 x 1 / 1    13 x  13 x1024   ->    13 x  13 x 512
   22 conv   1024  3 x 3 / 1    13 x  13 x 512   ->    13 x  13 x1024
   23 conv   1024  3 x 3 / 1    13 x  13 x1024   ->    13 x  13 x1024
   24 conv   1024  3 x 3 / 1    13 x  13 x1024   ->    13 x  13 x1024
   25 route  16
   26 conv     64  1 x 1 / 1    26 x  26 x 512   ->    26 x  26 x  64
   27 reorg              / 2    26 x  26 x  64   ->    13 x  13 x 256
   28 route  27 24
   29 conv   1024  3 x 3 / 1    13 x  13 x1280   ->    13 x  13 x1024
   30 conv    125  1 x 1 / 1    13 x  13 x1024   ->    13 x  13 x 125
   31 detection
```

Fig. 17: Screen shot of the computer during the training process.

During the training process using convolutional neural network architecture we can see in Figure 17 that a screen shot during the training process with layers used, type of layers, number of filters, size of filters, size of inputs, size of outputs etc.

V. Comparison of two Methodologies

Table I shows comparisons of techniques between Keras and Yolo based object detection.

Comparison between two methods	
Keras-Tensorflow model	YOLO mark
Software based approach	Graphical based approach
Deep Learning algorithm used	Deep Learning algorithm used
Keras needs backend to execute	No need any backends
Runs in both CPUs and GPUs	Runs in both CPUs and GPUs
Reduce the cognitive load	Reduce the number of inputs
Need more images to learn less accurate	Need less images to learn more accurate
Only gives prediction probabilities	Gives bounding box of each classes
Need a threshold value to identify the classes	No need any threshold value to get the results
Cannot locate the objects	Which can locate the objects
Real time detection accuracy is less as compared with YOLO	High accuracy for real time detection compared with Keras
Difficult to interface with Raspberry Pi	Easy to interface with Raspberry Pi

TABLE I: Comparison of techniques used in keras tensorflow model and yolo.

First method or Keras using Tensorflow backend method provides a software model that contains a generally used sequential model and creates a convolution neural network to learn the dataset using Tensorflow backend. But in second method or YOLO object detection method whprovides a graphical input by manually marked objects are entirely different from first method (Table I).

Both methods are using with deep learning algorithms of convolutionl neural network. Keras based model giving eight hidden layers which contains 5 convolutional layer and 3 fully connected layer. Because of high number of hidden layer used thus becomes more accurate as compared with the less number of hidden layers used. Whereas YOLO uses AlexNet architecture (similar pattern) that consists same layers used in the keras model.

Graphically giving inputs by manually marked objects in YOLO get more perfection as compared with the keras based model because YOLO based approach gives object classes from the real time image as well as the locations of objects in that frame. YOLO which gives bounding box inputs and gets bounding box outputs. Keras needs more image classes to train to get better accuracy whereas the YOLO no needs more number of input images to train. For the input creation of YOLO mark detection which is difficult to mark every objects to learn the networks.

VI. RESULTS AND DISCUSSION

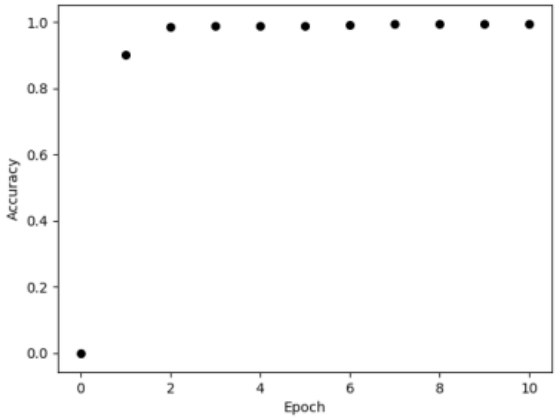
This chapter consists of results from both Keras based software model and from the graphical YOLO object detection models seperately. Finally both the methods are compared to apply the best method for waste identification .

Nvidia Quadro M5000 GPU WITH 8GB RAM is used for training both models. This device is very fast as compared with the ordinary CPUs. Four days are taken for training the keras model of 30000 images at the 500 epochs. But YOLO method of training is very short duration as compared with keras, which used two days for the training of 2000 images with 50000 iterations. And both the methods possess good accuracy for the identification of objects in the real time video capturing process. YOLO based approach finds much more usefull for the required tasks as compared with the software based approach.

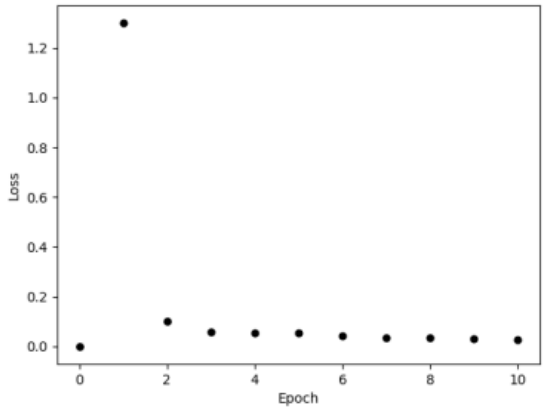
A. Detection Results from Keras Model

This section discusses the results from the training processes and prediction stages. Image datasets are used for training the images that defines how to use image datasets to create updated weight files with maximum accuracy and minimum loss function.

1) Training Results for Accuracy and Loss function: The training process showed good accuracy with an increase in the number of epochs and obtained a decreasing losses with increasing epochs as well as Shown in Figure 18.



(a)



(b)

Fig. 18: Accuracy and Loss function during training process.(a) Accuracy vs Epochs during training process,(b) Losses vs Epochs during training process

Here we can see that accuracy increases with epochs and losses decreases with epochs.

2) Prediction Results: Obtained a good result for the prediction of unseen image from the real time video. But there is lots of errors getting from the results due to imperfections of the input images. So the input image perfections is one

of the important parameter that depends on the probability of prediction accuracy. Accuracy increases with increasing number of input images as well as the hidden layers used in the convolutional network layers. After finishing the training process through the deep learning framework, they creates some weight files. By using these weight files, we want to detect objects and obstacles for unseen data environment. There is no difficult in predicting seen data environment. Seen data environment means, we used pictures or images for training process related to the same environment.

A screen shot taken from the captured real time video corresponds to the waste obtained as well as the ground alone as shown in Figure 19. Here we can see a good probability prediction value for seen environment images and for unseen environment images as in Table II. We need to detect the waste particle classes as objects from the captured real time video. In this work, when the probability value of the objects from the captured frame in video is higher than the threshold value (set as 0.5), then it is considered as a waste particle (Figure 19 a). Hence the controller is informed for further actions of cleaning tasks. Otherwise (for ground class), when probability of ground is higher than the threshold, it is considerd as a no waste as shown in Figure 19(b). In such cases, our vehicle move forward without cleaning the current area.

$P(O) > 0.5$ where $P(O)$ represents probability of getting object class and $P(G) > 0.5$ where $P(G)$ represents probability of getting groung class.

From the real time video analysis				
Input data Environment	Total prediction	Wrong prediction	Total input images used	Accuracy
Seen environment	74	26	100	74%
Unseen environment	58	42	100	58%

TABLE II: Detection accuracy for different environmental images.

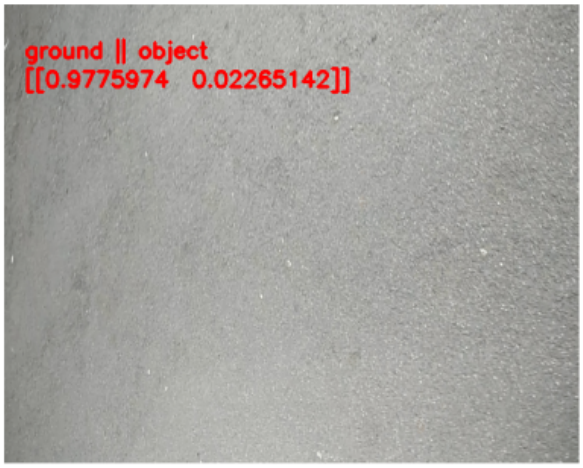
B. Results from YOLO

This section presents the results obtained from YOLO mark detection method during training process as well as the bounding box predictions in the seen environments and unseen environments. And finally compared the accuracy from different environments.

1) Training Losses for Different Number of Iterations : We need to optimize the losses during the training process to get good prediction results. Here, losses decreases with increasing iterations. Firstly, training has been done with 10000 iterations (Figure 20 a) which gets the average loss of 0.26 . We need to optimize the losses more, so trained with 50000 iterations (Figure 20 b) which gets the average loss of 0.02.

Hence Loss function is decreasing with increasing number of iterations . Simultanously accuracies increases with decreasing loss functions, because accuracies and loss functions are inversly proportional.

(a)



(b)

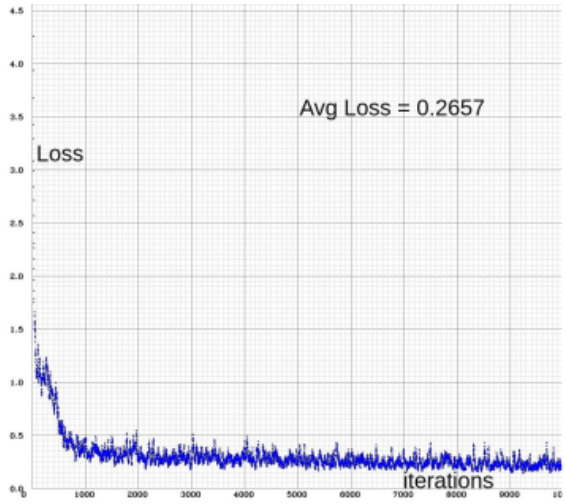
Fig. 19: Prediction results for different classes.(a) For an object class,(b) For a ground class

2) Prediction Results: The training process creates weight files and using this weight files, bounding box of objects in the unseen environments is predicted. Figure 21 represents bounding box predictions for seen environment.

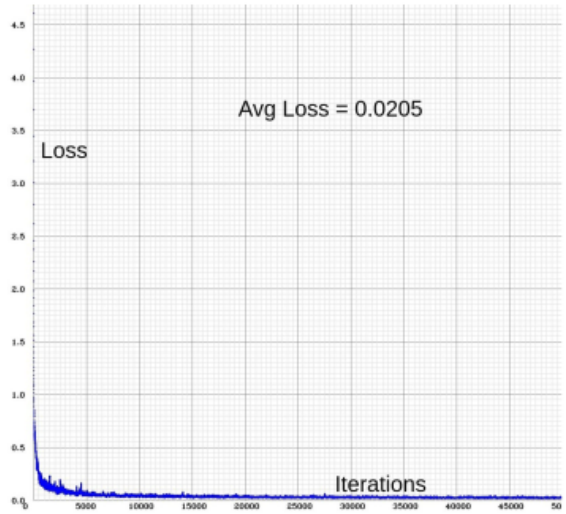
The unseen environment image is shown in figure 22 Detecting different environments that is never trained with the same background. That means to detect from the real time video, that video capturing area should not be used to create the training dataset. This type of detection is of less accuracy compared with the seen environment and comparison is done in the next section.

3) Accuracy from Different Environment Data: In this comparison study, 20 different images of both seen environment and unseen environment is used and the prediction is done. This gives 19 images of correct detection of objects for same environment where as 17 correct in the unseen environment. YOLO based detection method give good accuracy as compared with others like RCNN.

The below table represents comparison between different



(a)



(b)

Fig. 20: A screen shot of loss variations in different number of iterations during training process.(a) For 10000 iterations, (b) For 50000 iterations

data detection accuracies Table III.

Detection accuracy comparison				
Input image environment	Total detected	Wrong detection	Total input images used	Accuracy
Seen environment	19	1	20	95%
Unseen environment	17	3	20	85%

TABLE III: Detection accuracy for different environmental images.

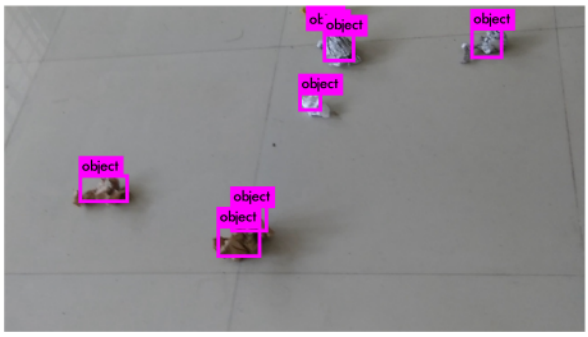
Using YOLO based image classification, we can predict or bound every objects from the captured image as bounded by training all the objects in the input images. The main advantages of this method is we can detect more environment

(a)

(b)

Fig. 21: Detection of different objects in captured image for seen environment.(a) Input Image,(b) Detected output showing obstacles and objects

(a)



(b)

Fig. 22: Detection of different objects in captured image for unseen environment.(a) Input image,(b) Objects detected

objects by providing more input images of objects trained from the different area of background or different environments. For example, in this work, it used more images of ground objects from the different environments such as college campus, roads, home court, college ground etc for the training process. We need to predict the classes of objects from the different environment like hospital court. Thus we can detect the objects by using the weight file getting after training process. This method represents an unseen environment detection. But for detection of same environment, we are using other images of seen environment like college campus that is trained before. This gets more accurated prediction results as compared with the unseen environment detection.

C. Output Comparison between Keras and YOLO

Finally compared two different deep learning frameworks. It shows that the accuracy of Yolo based approach is higher than the Keras-Tensorflow based software model.

Accuracy comparison			
Method used	Number of input images used	Number of Iterations	Accuracy
Keras	30000 30000	20 100	68% 72%
YOLO	2000 2000	10000 50000	81% 89%

TABLE IV: Accuracy results of Keras and YOLO for different iterations.

Table IV shows the comparison between Keras-Tensorflow based approach and YOLO based approach. In keras model, initially trained with 30000 images and 20 iterations in which weight file obtained 68 percentage for the real time capturing detection accuracy. This accuracy can be increased by changing the iterations. Keras based approach depends more on the type of environments that are trained. But YOLO approach gets good accuracy with less number of input images.

Finally we can say that the output prediction results much more depends on the types of images, number of input images and the number of iterations used for the training process.

CONCLUSION

The work proposed in this paper compares the performance of YOLO based approach with that of Keras based approach in object identification related problems. Accuracy increases with increase in epoch value and also on the type of input images. Results shows that Keras based approach gives prediction probabilities, but it cannot provide any location information about the objects. Whereas YOLO based approach provides the exact location information about the objects present in the scene. YOLO annotation of each image in the dataset has to be carried out which makes the dataset preparation much more difficult when compared to the Keras based approach. In case of Keras based approach, more number of images have to be present in the dataset for accurate detection. Results indicate that the prediction accuracy of YOLO based approach is better than that of Keras based approach.

REFERENCES

[1] Felzenszwalb, Pedro F., et al. "Object detection with discriminatively trained part-based models." IEEE transactions on pattern analysis and machine intelligence 32.9 (2010): 1627-1645.

[2] Girshick, Ross. "Fast r-cnn." arXiv preprint arXiv:1504.08083 (2015).

[3] Redmon, Joseph, et al. "You only look once: Unified, real-time object detection." Proceedings of the IEEE conference on computer vision and pattern recognition. 2016.

[4] Redmon, Joseph, and Ali Farhadi. "YOLO9000: better, faster, stronger." arXiv preprint (2017).

[5] Prabhakar, Gowdham, et al. "Obstacle detection and classification using deep learning for tracking in high-speed autonomous driving." IEEE Region 10 Symposium (TENSYMP), 2017. IEEE, 2017.

[6] Jmour, Nadia, Sehla Zayen, and Afef Abdelkrim. "Convolutional neural networks for image classification." 2018 International Conference on Advanced Systems and Electric Technologies (ICASET). IEEE, 2018.

[7] Shiddieqy, Hasbi Ash, Farkhad Ihsan Hariadi, and Trio Adiono. "Implementation of deep-learning based image classification on single board computer." Electronics and Smart Devices (ISESD), 2017 International Symposium on. IEEE, 2017.

[8] Gallardo, Nicolas, et al. "Autonomous decision making for a driver-less car." System of Systems Engineering Conference (SoSE), 2017 12th. IEEE, 2017.

[9] Murugan P. Implementation of Deep Convolutional Neural Network in Multi-class Categorical Image Classification. arXiv preprint arXiv:1801.01397. 2018 Jan 3.

[10] Cai, Zhaowei, et al. "A unified multi-scale deep convolutional neural network for fast object detection." European Conference on Computer Vision. Springer, Cham, 2016.

[11] Airola, Rasmus, and Kristoffer Hager. "Image Classification, Deep Learning and Convolutional Neural Networks: A Comparative Study of Machine Learning Frameworks." (2017).

[12] Hnoohom, Narit, and Sumeth Yuenyong. "Thai fast food image classification using deep learning." Electrical, Electronics, Computer and Telecommunications Engineering (ECTI-NCON), 2018 International ECTI Northern Section Conference on. IEEE, 2018.

[13] Bhatnagar, Shobhit, Deepanway Ghosal, and Maheshkumar H. Kolekar. "Classification of fashion article images using convolutional neural networks." Image Information Processing (ICIIP), 2017 Fourth International Conference on. IEEE, 2017.

[14] Yu, Zhenxia, Chengxuan Shen, and Lin Chen. "Gender classification of full body images based on the convolutional neural network." Security, Pattern Analysis, and Cybernetics (SPAC), 2017 International Conference on. IEEE, 2017.