Implementing spot the differences game using Yolo algorithm.

KONDAPANENI, C.S., TEJA, M.V.S., KAVURI, R., TINNAVALLI, D. and BANO, S.

2022

Implementing Spot the Differences Game using Yolo Algorithm

* Charan Sai Kondapaneni[1], Modi Venkata Sai Teja [2], Rohith Kavuri[3], Deepika Tinnavalli[4], Shahana Bano[5,]
Department of CSE,

Koneru Lakshmaiah Educational Foundation
Vaddeswaram, India

[1]kondapanenicharansai@gmail.com
[2]modi.saiteja@gmail.com
[3]convey2rohitkavuri@gmail.com
[4] deepikatinnavalli@gmail.com
[5]shahanabano@icloud.com

Abstract. The requirement for Object Detection has been expanding with computational force. Object identification is a technique that identifies the semantic class of the objects in the image or video. In this work, we talk about implementing an application that connects with the user and distinguishes the various objects present in two generally comparable images utilizing object recognition algorithms (YOLO Algorithm). The user collaborates with the application through speech and recognizes the object. This execution has a precision of 70%.

Keywords: Object Detection, YOLO, Speech Recognition, Object Classification

1 Introduction

Object Detection is one of the most generally utilized tools in computer vision. The Natural eye can recognize objects to a wide reach however carrying that ability to a computer is a difficult errand. Object Detection is directly or indirectly utilized in many fields like self-driving vehicles, video surveillance, robust vision, and many more. The principle thought of object recognition is to recognize objects and arrange them likewise. Object Detection began with customary algorithms like SIFT (scale-invariant element change) [2] which is found in the '90s and the performance of these sorts of calculations are not satisfactory, then, the profound learning methods have arisen which gave goliath jump in the presentation and exactness. Our principal philosophy is to implement an application that distinguishes differences between two images utilizing Yolo Algorithm. Simultaneously it can recognize various objects in the image. There are a few algorithms for object recognition, these are isolated into two kinds like Classification and Regression [8]. CNN and RNN come out under groups under Classification while Yolo groups to regression. Previously, classifiers are used to perform object detection, but by using YOLO algorithm object detection is framed as a regression problem [16].

CNN and RNN are comparatively slow as we have to run these algorithms every time for a selected region in images. But in Yolo, we can directly predict the with bounding boxes and classes. Yolo is one of the pre-trained models. There are various versions of YOLO algorithm each having an advantage over the previous version [8].

1.1 Working Of YOLO Algorithm

YOLO is the short term of "YOU ONLY LOOK ONCE", an object detection algorithm used to distinguish and identify different objects present in images in real-time. As the name recommends, it runs just a single time through the whole image. To comprehend the working of YOLO one should understand what it is predicting. YOLO algorithm predicts the class probabilities and bounding boxes of the objects present in the image.
The algorithm does not find the region of interest, rather it isolates the whole image into networks or cells typically of size SxS [5]. Every cell is then responsible for recognizing K-bounding boxes.

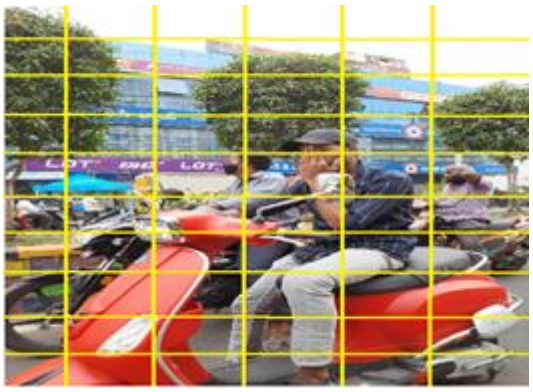

Fig:1

Bounding boxes can be portrayed utilizing four parameters specifically,
Center of the Box (bx, by)
Height of the Box (bh)
Width of the Box (bw) [8]
An object is considered to lie in a cell provided that its center coordinates line inside that cell. The height and width coordinates are constantly determined compared to the image.

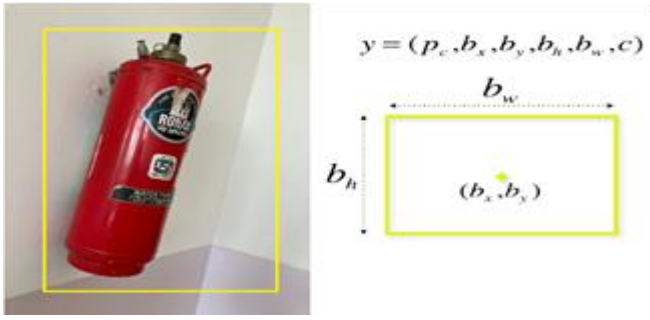

Fig:2

1.2 Non-Maximum Suppression

Objects sometimes tend to get detected more than one time, resulting in more than one bounding box. This can be resolved using Non-Maximum Suppression, which suppresses the bounding boxes with non-maximum probabilities. Hence, the bounding box with maximum class probability is used.

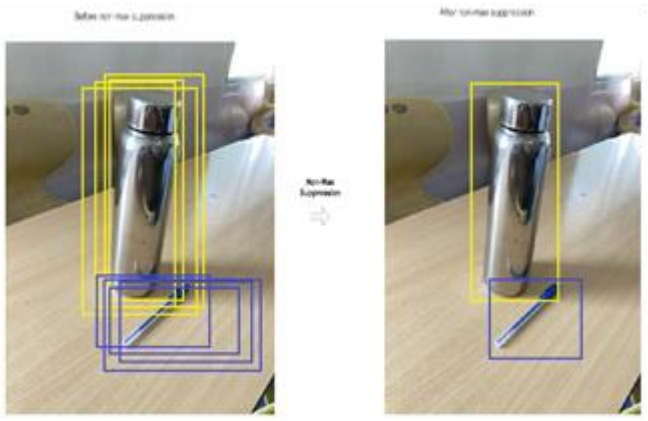

Fig:3

1.3 YOLO Architecture

YOLO architecture is a combination of convolution and max pool layers. The max pool layer picks the maximum element from the feature map which will be having more prominent features than the remaining. Whereas the convolution layer extracts all the features from an image among the available features the max pool layer picks the maximum element out of it.

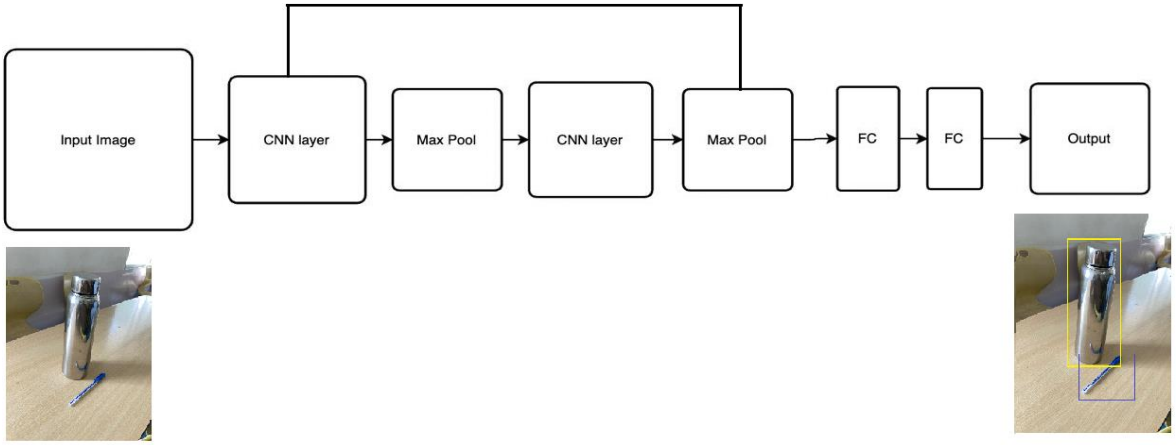

Fig:4

1.4 Comparision Of Various Versions Of YOLO Algorithm

Parameter	YOLO v2	YOLO v3	YOLO v4
Pre- trained objects	80	80	80
Bounding Boxes per Cell (for 416x416 pixel image)	845	10647	10647
Frames per second	40-90	>90	>90
Time Taken for Identifying Objects		439 ms/(16-17) objects	570 ms/ 16 objects
Framework	Darknet – 19. 19 convolutional layers and 5 Max Pool Layers	Darknet – 53. 53 convolutional layers. For detection 53 more are added making 106 layers in total.	
Small Size Object Detection	Smaller objects are detected	Better than v2 [17]	

Table:1

Accuracy of various versions of the YOLO Algorithm on images which contains objects on which these algorithms are trained.

Version	Correct Prediction	Wrong Prediction	Accuracy
YOLO v2	60	20	75.00 %
YOLO v3	65	15	81.25 %
YOLO v4	64	16	80.00 %

Table:2

There are many metrics to evaluate performance of these algorithms and one of the popular metrics being Average precision (AP) [12].
YOLO v3 is one of the most widely used object detection method and it also uses k-means cluster method [7].

2 Procedure

The point of our work is to implement an application that recognizes the different objects present in two comparative images. The user communicates with the model through the speech module. This speech module is utilized to change speech over to message [22].
First, the system arbitrarily shows two comparable images with a couple of differences. It additionally distinguishes the objects present in the images, yet won't be shown to the user. Now the user recognizes the various objects present in the images. Then, at that point, by utilizing the speech module (speech to message converter), the user inputs the distinctive object present in the first image and not in

the second image. In the event that the information given by the user is available in the rundown of objects distinguished by the model, then, at that point, the model will return "Right Answer" as output and draw a bounding box to indicate the area of the object. In the event that the info, given by the user isn't recognized by the model or then again assuming that information is absent in the main image, the model returns "Wrong Answer" as output and won't draw a bounding box.

2.1 Data For Implementation

The data utilized for testing and execution of the application are modified images that contain objects that are now pre-trained by the model. That is, few of the individual images of the objects that are pre-trained are consolidated together to frame one new image. Such produced images are kept in folders. Now, the model chooses two images arbitrarily and recognizes the distinction between these two images.

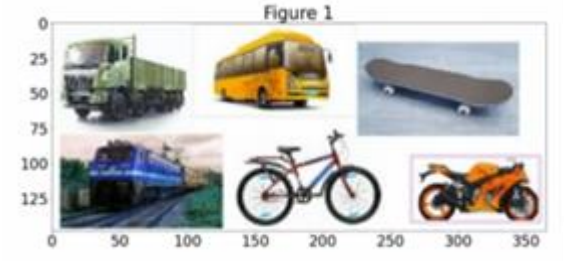

Fig :5

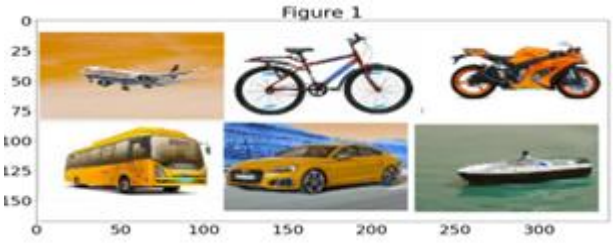

Fig:6

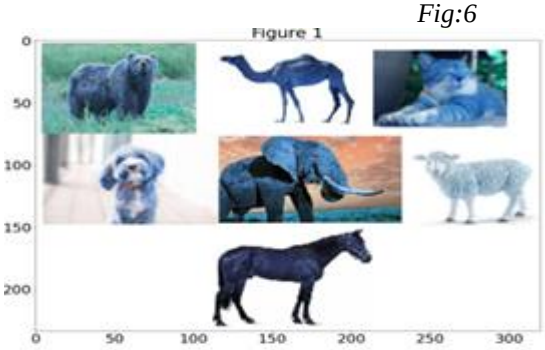

Fig:7

In the above images, we collected different images of objects and combined them into one single image. Now, the model detects the objects present in this new image.

3 Pseudo Code

Step1: Start.
Step2: Provide the input of 2 images.
Step3: Prepare the images for the image processing tasks (resizing the images by specifying the resized image sizes).
Step4: It returns the resized images with the specified size.
Step5: Obtain the Histogram of image. The histogram's values sum to 1.
Step6: Get the sum of values accumulated by each position in hist.
Step7: Determine the normalization values for each unit of the CDF.
Step8: Normalize each position in the output image
Step9: Measures the Earth Mover's distance (Wasserstein distance) between two images by taking the paths of image files as arguments and returns the Wasserstein distance.
Step10: Measures the structural similarity between two images by taking the paths of image files as arguments.
Step11: Measures the pixel similarity between two images by taking paths of image files as arguments.
Step12: It initializes the ORB (Oriented Fast Rotated Brief) Feature detector to detect and extract the features and obtain measure of similarities.
Step13: Get the images that are normalized and resized and generates the key points and descriptors with ORB.
Step14: Find the matches between the images using the Brute Force (BF) Matcher and store all the matches of the descriptors in an array.
Step15: Measure the score of similarity of all possible matches using the above-mentioned steps 6,7,8.
Step16: Output the similarity scores of all measures used.
Step17: Stop.

The Below Figure represents the workflow representation of the whole process.
The randomly generated images are displayed to the user. The user identifies the differences from the images.

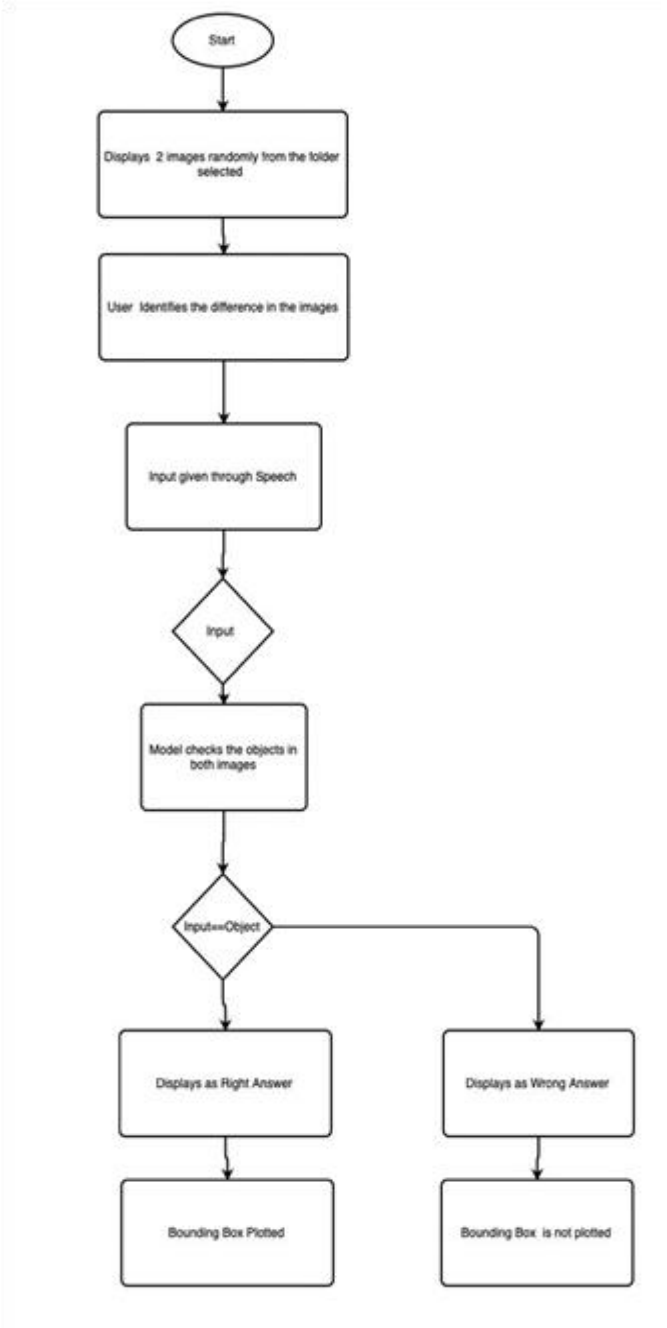

Fig:8
workflow representation

3.1 Data Flow Diagram

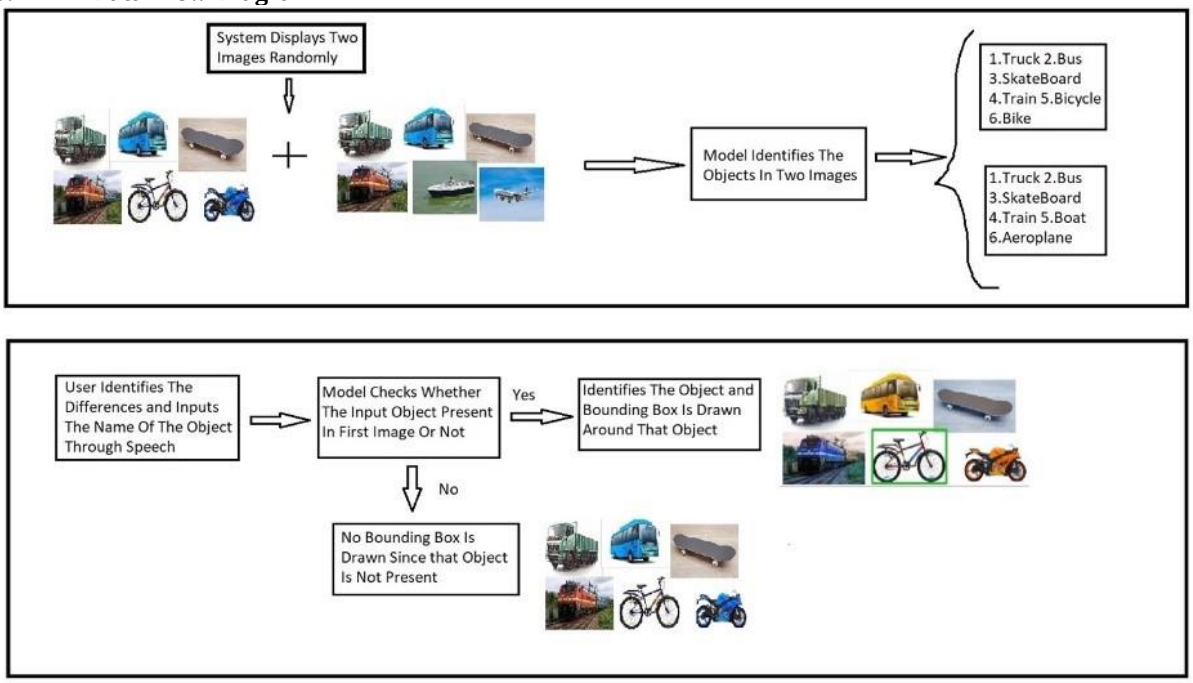

Fig:9

4 Results

The model randomly selects these two images randomly and displays them to the user.

Input:

Fig:10

Output:

```
Detection_Of_Object("motorbike")
Right Answer
```

Fig:11

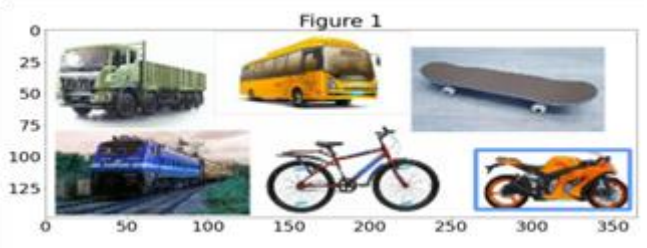

Fig:12

As shown, when the input given by the user is present in the first image and not in second the image, the model displays “Right Answer” draws a bounding box ar ound that object.

```
Detection_Of_Object("train")
```

```
Wrong Answer
```

Fig 13

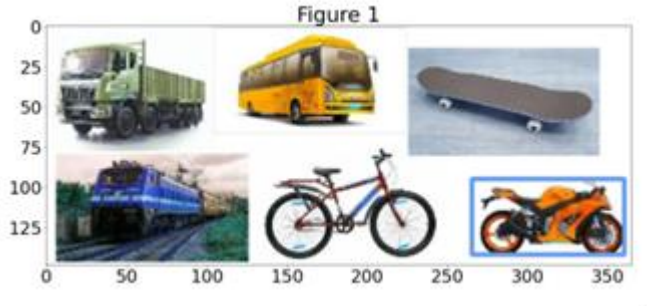

Fig:14

As shown, when the input given by the user is present is both the images, the mod el displays “Wrong Answer” and does not draw a bounding box.

Input:

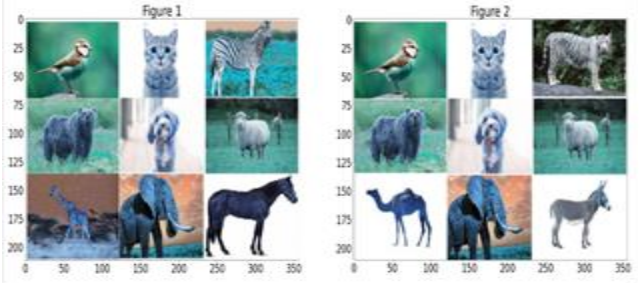

Fig:15

Two images were randomly selected and displayed to the user by the model.
Output:

```
Detection_Of_Object("giraffe")
```

```
Right Answer
```

Fig:16

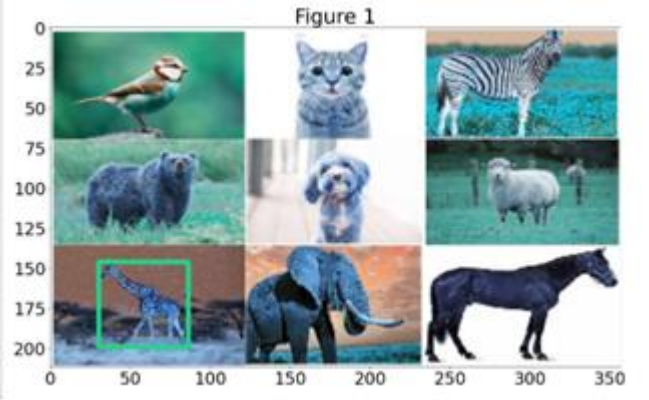

Fig:17

The input given by the user has been identified and a bounding box is drawn aroun d that object.

Input:

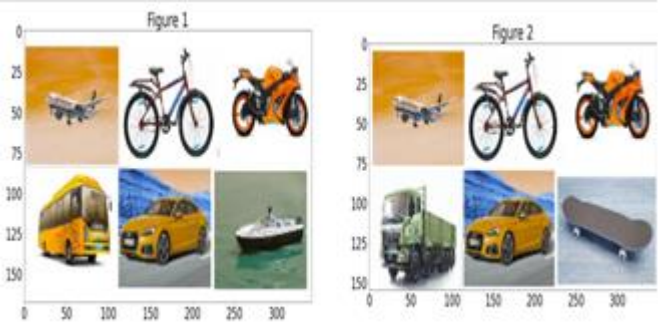

Fig:18

Output:

```
Detection_Of_Object("motorbike")
```

```
Wrong Answer
```

Fig:19

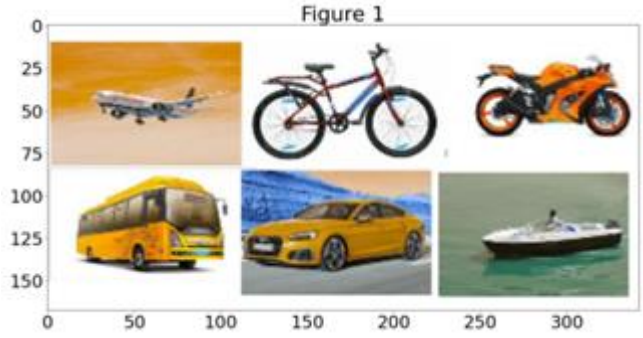

Fig:20

The input given by the user is
not present in the first image. Hence no bounding box is drawn.

5 Drawback

This application is confined to recognizing just 80 pre-trained class objects. The fundamental disadvantage of this application is that it wrongly recognizes an object of undeveloped classes as one of the pre-trained class objects. For example, it recognizes an image of a Donkey as a Horse and an image of a White Tiger as Cat. To beat this downside, we can train the model with more custom datasets and classes.

6 Conclusion

Recognizing the objects in images is one of the crucial assignments in computer vision on account of its expansive scope of uses. This paper extends an outline of the working of the Yolo algorithm for object recognition. This paper likewise centers around executing an application for finding various objects present in two generally comparative images utilizing Yolo v3 object location calculation. The diverse object present in the main image will be represented by the bounding box. By utilizing Non-Max Suppression work this model recognizes the bounding box with the greatest confidence values.

7 Future Scope

The application examined in this paper can be utilized for distinguishing the differences between two images. This can likewise be created to distinguish similitudes among images [1].and it additionally can be created as a gaming application for the “Spot the Differences” game. This can additionally be created to recognize objects in crime location examination. YOLOv3 algorithm can be utilized in different fields to tackle some genuine issues like security, observing traffic,

parking assistance [17], port administration [21], and visually impaired people. This execution can additionally be scaled to distinguish different custom objects.

References

[1] Varshini Appana, Tulasi Manasa Guttikonda, Divya Shree, Shahana Bano, Hi masri Kurra, "Similarity Score of Two Images Using Different Measures", 6th International Conference On Inventive Computation Technologies,2021.

[2] Himasri Kurra, Tulasi Manasa Guttikonda, Guntaka Greeshmanth Reddy, Shah ana Bano, Vempati Biswas Trinadh, "Detecting Images Similarity Using SIFT", International Conference On Expert Clouds and Applications, 2021.

[3] Dietrich Van der Weken. Mike Nachtegael. Etienne E. Kerre, "AN OVERVIEW OF SIMILARITY MEASURES FOR IMAGES", IEEE International Conference on Acoustics, Speech, and Signal Processing, 2002.

[4] Tanvir Ahmad, Yinglong ma, Muhammad Yahya, Belal Ahmad, Shah Nazir, Amin Ul Haq, Rahman Ali, "Object Detection through Modified YOLO Neural Network", 2020.

[5] Geethapriya. S, N. Duraimurugan, S.P. Chokkalingam, "Real-Time Object Detection With Yolo",
International Journal Of Engineering and Advanced Technology, 2019.

[6] Omkar Masurekar, Omkar Jadhav, Prateek Kulkarni, Shubham Patil, "Real Time Object Detection Using YOLOv3", International Journal Of Engineering and Advanced Technology, 2020.

[7] Liquan Zhao, Shuaing Li,
"Object Detection Algorithm Based on Improved YOLOv3", 2020.

[8] Upulie Handalage, Lakshini Kugunandamurthy, "Real- Time Object Detection Using YOLO: A review", 2021.

[9] Hatem Ibrahem, Ahmad Diefy Ahmad Salem, Hyun-Soo Kang, "Real-Time Weakly Supervised Object Detection Using Center-of-Features Localization", 2021.

[10] Wei Fang, Lin Wang, Peiming Ren, "Tinier-YOLO: A Real-Time Object Detection Method for Constrained Environments", 2019

[11] Rachel Huang, Jonathan Pedoeem, Cuixian Chen, "YOLO-LITE: A Real-Time Object Detection Algorithm Optimized for Non-GPU Computers",2019.

[12] Rafael Padilla, Sergio L. Netto, Eduardo A. B. da Silva, "A Survey on Performance Metrics for Object-Detection Algorithms", 2020.

[13] Zhiqing Sun, Shengcao Cao, Yiming Yang, Kris M. Kitani,
"Rethinking Transformer-Based Set Prediction for Object Detection", 2021.

[14] Mate Kristo, Marina Ivasic-Kos, Miran Pobar, "Thermal Object Detection in Difficult Weather Conditions Using YOLO", 2020.

[15] Yunhua Yin, Huifang Li, Wei Fu, "Faster-YOLO: An accurate and faster object detection method", 2020.

[16] Joseph Redmon, Santosh Divvala, Ross Girshick, Ali Farhadi, "You Only Look Once :Unified, Real-Time Object Detection", 2016.

[17] Qiwei Xu, Runzi Lin, Han Yue, Hong Huang, Yun Yang, Zhigang Yao, ”Research on
Small Target Detection in Driving Scenarios Based on Improved Yolo Network”, 2020.
[18] Ashwani Kumar, Zuopeng Justin Zhang, Hongbo Lyu,”Object detection in re al time based on improved single shot multi-box detector algorithm”,
EURASIP Journal on Wireless Communications and Networking, 2020.
[19] Shenlu Jiang, Wei Yao, Man Sing Wong, Gen Li, Zhongua Hong, Tae-Yong Kuc, Xiaohua Tong,
”An Optimized Deep Neural Network Detecting Small and Narrow Rectangular O bjects in Google Earth”, 2020.
[20] Changqing Cao, Bo Wang, Wenrui Zhang, Xiadong Zeng, Xu Yan, Zhejun F eng, Yutao Liu, Zengyan Wu, “An Improved Faster R-CNN for Small Object Detection”, 2019.
[21] Hao Li, Lianbing Deng, Cheng Yang, Jianbo Liu, Zhauquan Gu,
“Enhanced YOLO v3 Tiny Network for Real-Time Ship Detection From Visual Image”, 2021.
[22] Sameer Bansal, Herman Kamper, Adam Lopez, Sharon Goldwater, “Towards speech-to-text translation without speech recognition“, 2017.