STANLEY COLLEGE OF ENGINEERING AND TECHNOLOGY FOR WOMEN
(AUTONOMOUS)

ABIDS, HYDERABAD

PE 515 CS – DATA SCIENCE

Case Study Report (UNIT – V)

Team # 3

TITLE: OBJECT RECOGNITION

S.NO	Roll Number	Name
1.	160620733001	Addi Poojeetha Reddy
2.	160620733002	Advaitha Daduvy
3.	160620733003	B.Neha
4.	160620733006	Badam Rithika
5.	160620733008	Bobbili Hrishitha
6.	160620733010	Chanda Akshara
7.	160620733011	Chinthala Pranathi
8.	160620733018	Janapati Himaja
9.	160620733019	Jangili Madhurima
10.	160620733020	Jetram Aashritha
11.	160620733021	K.Mallika
12.	160620733022	Kadarla Sharvani
13.	160620733023	Kantha Ananya
14.	160620733032	Mudavath Lavanya
15.	160620733043	Pulakurthi Anaghaa Reddy
16.	160620733060	V.Keerthana
17.	160620733302	Boyala Niharika

Table of Contents

List of Figures

Introduction

A computer vision technology called Object Recognition can be used to locate items in photos or movies. Object recognition is the area of artificial intelligence (AI) concerned with the abilities of robots and other AI implementations to recognize various things and entities. It allows robots and AI programs to pick out and identify objects from inputs like video and still camera images. Object recognition is a key output of deep learning and machine learning algorithms. Humans are adept at identifying individuals, objects, settings, and other visual elements when viewing photographs or videos.

The objective is to train a computer to perform something that comes effortlessly to people: grasp what an image conveys. Driverless automobiles use object recognition as a fundamental technology to identify a stop sign or tell a pedestrian from a lamppost. It is also helpful in many different applications, including robotic vision, industrial inspection, and illness identification in bioimaging. Various methods can be used for object recognition. Deep learning and machine learning methods have recently gained popularity as solutions to object recognition issues. Both methods teach users how to recognise items in pictures, but they operate in different ways. The main purpose of object detection is to identify and locate one or more effective targets from still image or video data.

The objective of this case study is to demonstrate Image classification and detection of objects using R language.

Explanation about the topic

Object recognition refers to a collection of related tasks for identifying objects in digital photographs.

- Region-Based Convolutional Neural Networks, or R-CNNs, are a family of techniques for addressing object localization and recognition tasks, designed for model performance.

- **Image classification**: Algorithms produce a list of object categories present in the image.

- **Single-object localization**: Algorithms produce a list of object categories present in the image, along with an axis-aligned bounding box indicating the position and scale of one instance of each object category.

Object detection: Algorithms produce a list of object categories present in the image along with an axis-aligned bounding box indicating the position and scale of every instance of each object category.

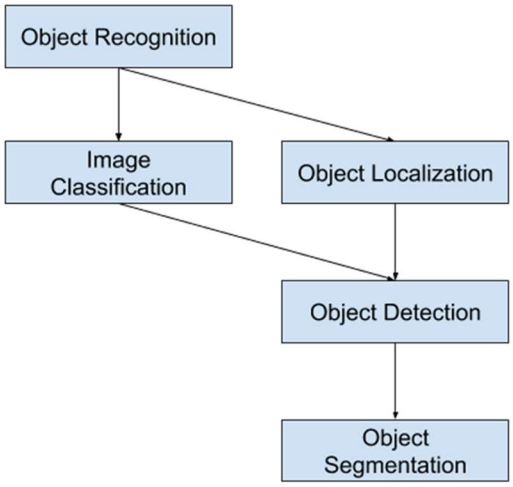


FIG 1 : OBJECT RECOGNITION

The performance of a model for image classification is evaluated using the mean classification error across the predicted class labels. The performance of a model for single-object localization is evaluated using the distance between the expected and predicted bounding box for the expected class. Whereas the performance of a model for object recognition is evaluated using the precision and recall across each of the best matching bounding boxes for the known objects in the image.

Object recognition is a computer vision technique for identifying objects in images or videos. Object recognition is a key output of deep learning and machine learning algorithms. When humans look at a photograph or watch a video, we can readily spot people, objects, scenes, and visual details. The goal is to teach a computer to do what comes naturally to humans: to gain a level of understanding of what an image contains.

Object recognition is a key technology behind driverless cars, enabling them to recognize a stop sign or to distinguish a pedestrian from a lamppost. It is also useful in a variety of applications such as disease identification in bioimaging, industrial inspection, and robotic vision.

Problem Statement:

- The development of reliable object recognition system for apt object recognition is still a challenge today.
- Requirement is to recognize the image regardless of position and rotation.
- Objects in images must be distinguished from their backgrounds. The major challenges in object detection are classifying objects and determining their position.
- However for the computer to imitate and detect objects like a human eye is a challenge.

- Some of the problems incurred are variation of view points, illumination, shapes as well as sizes of the goal objects.

- Processed photographs of objects under various orientations are considered.

- The goal is to find an approach for detection of objects with high accuracy and maximum automation for a given input photograph.

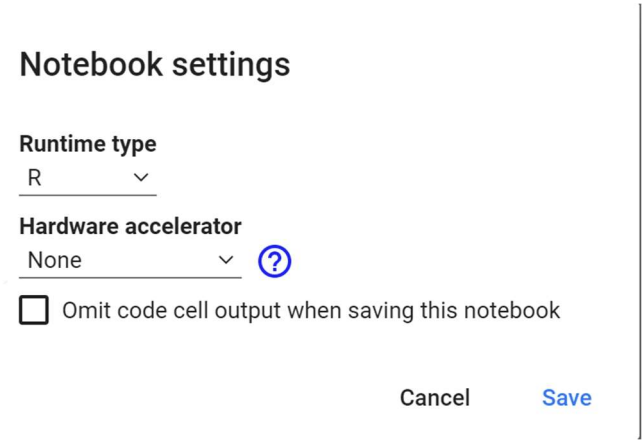
- The language in which the code will be written is R.

Getting things ready, Explaining functions, libraries of the code and Code with Output:

Example 1:

STEP 1: Go to Google Colab and change the runtime environment to R using the below link https://colab.research.google.com/notebook#create=true&language=r .

Then go to Runtime -> Change Runtime Type ->



Click save.

Now, go to Settings icon near your profile and then click on GitHub. Now Click the tick on the checkbox of "Access private repositories and organizations" . Click Save.

STEP 2: Implement the Object Recognition Code using R . Click on Play for each cell of the code to get executed. Then check whether it is green or red tick. If it is green then proceed, or else if it is red ,check where the error occurred and solve it.

1. Install package

```
install.packages("image.darknet", repos = "https://bnosac.github.io/drat")
```

```
Installing package into '/usr/local/lib/R/site-library'
(as 'lib' is unspecified)
```

2. Load package

```
library(image.darknet)
```

3. If you want you can even set a working directory as
   setwd("C:/Users/16138/Documents/YouTube/Image Recognition")
4. Classify : Here image_darknet_model is the function from the image.darknet package and there are 4 arguments in it, called as below and assigning the result to the variable named “darknet_model”
   (i) type : This can be set to either classify or detect
   (ii) model : We’re going to use a pre-trained model from image.darknet package. For classification, we’ll use a model called “tiny.cfg”
   (iii) weights : The weights for this model are stored in a file on your computer, in the same location as where this package was installed or if it the package was by default or in-built in online compilers. The weights we’re interested in for classification for this particular model is called “tiny.weights”.
   (iv) labels : This also is going to be stored in a file on your computer, in the same location as where this package was installed or if it the package was by default or in-built in online compilers. And for this model, the labels are stored in “imagenet.shortnames.list”

```
darknet_model <- image_darknet_model(type = "classify",
                                     model = "tiny.cfg",
                                     weights = system.file(package = "image.darknet", "models", "tiny.weights"),
                                     labels = system.file(package = "image.darknet", "include", "darknet", "data", "imagenet.shortnames.list"))
```

FIG 2.1 : HOME UPLOADED IMAGE

Copy the path of the downloaded image file which needs to be detected of .png , .jpeg or .jpg format and paste the path where it is concerned to be pasted.

Here in this case , it is “/content/download.jpg”

5. Here, we’re going to actually apply this model onto an image. To do that we’re going to use image_darknet_classify function of 2 arguments specified as
   (i) file : Copy and paste the path of the image file which needs to classified
   (ii) object : Pass the model which we created in the previous step

   Now run the code to see the outputs

```
image_darknet_classify(file = "/content/download.jpg",
                       object = darknet_model)
```

$file
'/content/download.jpg'
$type
A data.frame: 5 × 2

label	probability
<chr>	<dbl>
studio couch	0.73350757
window shade	0.08380695
four-poster	0.05149113
dining table	0.03175243
plate rack	0.01800213

6. Detect : Here image_darknet_model is the function from the image.darknet package and there are 4 arguments in it, called as below and assigning the result to the variable named “darknet_model”
   (i) type : This can be set to either classify or detect
   (ii) model : We’re going to use a pre-trained model from image.darknet package. For detection, we’ll use a model called “tiny-yolo-voc.cfg”
   (iii) weights : The weights for this model are stored in a file on your computer, in the same location as where this package was installed or if it the package was by default or in-built in online compilers. The weights we’re interested in for classification for this particular model is called “tiny-yolo-voc.weights”.
   (iv) labels : This also is going to be stored in a file on your computer, in the same location as where this package was installed or if it the package was by default or in-built in online compilers. And for this model, the labels are stored in “voc.names”

7.System.file():Finds the full file names of files in packages etc.

system.file(…, package = "base", lib.loc = NULL, mustWork = FALSE)

Arguments:

character vectors, specifying subdirectory and file(s) within some package.

default, none, returns the root of the package. Wildcards are not supported.

(i) package: a character string with the name of a single package. An error occurs if more than one package name is given.

(ii)lib.loc: a character vector with path names of R libraries.

(iii)mustWork: logical. If TRUE, an error is given if there are no matching files.

```
darknet_model <- image_darknet_model(type = "detect",
                                     model = "tiny-yolo-voc.cfg",
                                     weights = system.file(package = "image.darknet", "models", "tiny-yolo-voc.weights"),
                                     labels = system.file(package = "image.darknet", "include", "darknet", "data", "voc.names"))
```

8.We're using image_darknet_detect function with 2 arguments passed into it

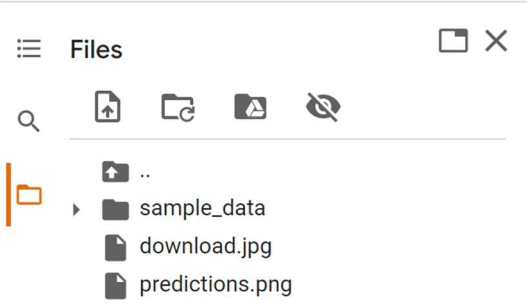
(i) file : Copy and paste the path of the image file which needs to detected

(ii) object : Pass the model which we created in the previous step

Now run the code to see the outputs

```
image_darknet_detect(file = "/content/download.jpg",
                     object = darknet_model)
```

STEP 3: After executing the above code cell, then we can observe the new file named "predictions.png" , which is the output.



Open predictions.png file.

OUTPUT:

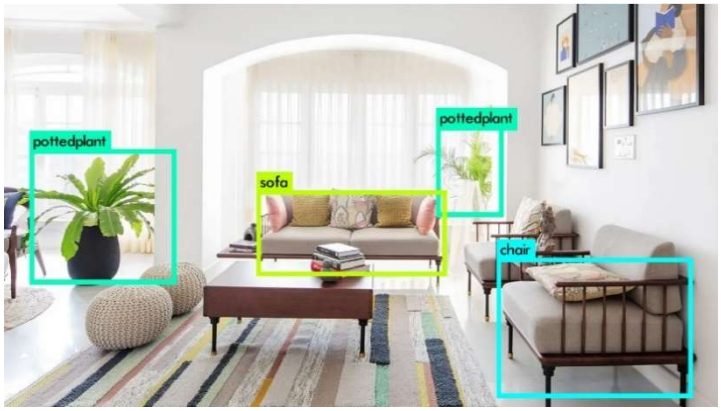


FIG 2.2 : HOME DETECTED/RECOGNIZED IMAGE

Example 2:

```
[2] install.packages("image.darknet", repos = "https://bnosac.github.io/drat")
```

```
Installing package into '/usr/local/lib/R/site-library'
(as 'lib' is unspecified)
```

```
[3] library(image.darknet)
```

```
[4] darknet_model <- image_darknet_model(type = "classify",
                                         model = "tiny.cfg",
                                         weights = system.file(package = "image.darknet", "models", "tiny.weights"),
                                         labels = system.file(package = "image.darknet", "include", "darknet", "data", "imagenet.shortnames.list"))
```

```
image_darknet_classify(file = "/content/maxresdefault.jpg",
                       object = darknet_model)
```

$file
'/content/maxresdefault.jpg'
$type

A data.frame: 5 × 2

label	probability
<chr>	<dbl>
home theater	0.3406775
television	0.1273764
screen	0.1212719
monitor	0.1203282
desktop computer	0.0723283

```
[6] darknet_model <- image_darknet_model(type = "detect",
                                         model = "tiny-yolo-voc.cfg",
                                         weights = system.file(package = "image.darknet", "models", "tiny-yolo-voc.weights"),
                                         labels = system.file(package = "image.darknet", "include", "darknet", "data", "voc.names"))
```

```
[7] image_darknet_detect(file = "/content/maxresdefault.jpg",
                         object = darknet_model)
```

Uploaded image:

FIG 3.1 : UPLOADED IMAGE

Detected image or Output:

FIG 3.2 : DETECTED IMAGE

Example 3:

It is same as the Example 1 , but only one change must be done i.e you have to copy the path of the other image which have to be detected or recognized and paste it in the concerned areas , i.e near image_darknet_classify and image_darknet_detect .

Uploaded image:

FIG 4.1 : UPLOADED IMAGE

Detected image or Output:

FIG 4.2 : DETECTED IMAGE

EXAMPLE 4:

Uploaded image:

FIG 5.1 : UPLOADED IMAGE

Detected image or Output:

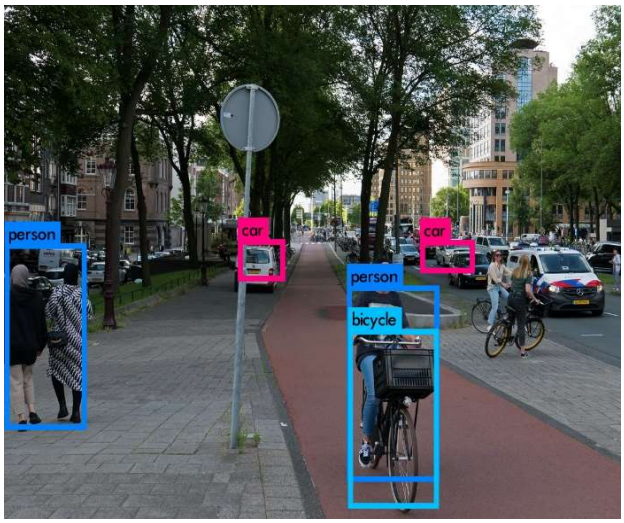


FIG 5.2 : DETECTED IMAGE

EXAMPLE 5:

```
image_darknet_classify(file = "/content/cat-dog-26409253.jpg",
                       object = darknet_model)
```

$file
'/content/cat-dog-26409253.jpg'
$type
A data.frame: 5 × 2

label	probability
<chr>	<dbl>
golden retriever	0.61046904
cocker spaniel	0.20111656
English setter	0.05069079
Labrador retriever	0.02421339
clumber	0.02402302

FIG 6.1 : UPLOADED IMAGE

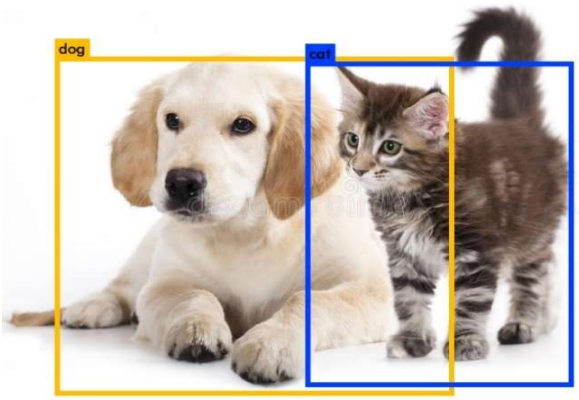


FIG 6.2 : DETECTED IMAGE

Other Considerations and Applications :

Considerations:

- Support vector machine (SVM) has been adapted to deal with several feature vectors per image.
- Different clustering configurations of SIFT key points in relation with their pose parameters: coordinates location, scale and orientation have been evaluated.
- ILF have been used for the description of the acrosome of boar spermatozoa heads yielding a successful classification of spermatozoa heads as intact or damaged.
- A novel method has been introduced for the effective description and classification of inserts, as broken or unbroken, with respect to the state of their cutting edges.
- An early fusion of ILF with global texture descriptors has been proposed for the classification of the integrity of the head acrosomes, demonstrating that some of the combinations of global and local features improve the accuracy obtained when using them separately.
- Another, more versatile and generic, method for the localisation of inserts has been presented.
- Colour COSFIRE filters have been proposed, adding colour description and discrimination power to COSFIRE filters as well as providing invariance to background intensity.
- A highly effective and efficient method for the localisation of cutting edges in milling machines has been presented.

Applications:

- Image retrievals
- Security
- Surveillance
- Automated vehicle parking system
- Machine inspection
- Improving iris recognition
- Augmented reality applications and gaming
- Powering self-driving cars
- Creating city guides

Conclusion:

Object recognition is a computer vision technique for identifying objects in images or videos. Object recognition technology is a basic research in the field of computer vision. Object recognition is a key output of deep learning and machine learning algorithms. When humans look at a photograph or watch a video, we can readily spot people, objects, scenes, and visual details.

Its main function is to recognize what object is in the image and give the position and direction of the object in the image. The recognition effect is more and more accurate, and the algorithm is more robust. But the better the effect and efficiency of the algorithm means the improvement of the network and hardware requirements, which is a relatively large restriction for the application of the algorithm. It has been evaluated for the particular task of object detection in a mobile robot platform. Furthermore, a number of variations or improvements to the selected methods are being actively produced and evaluated.

Object recognition can help predict the regions of attention and vice versa .Main tasks includes Tagging, Detection, Segmentation.

Object recognition is a fundamental process that serves as a gateway from vision to cognitive processes such as categorization, language and reasoning.

The goal is to teach a computer to do what comes naturally to humans: to gain a level of understanding of what an image contains. In addition, the current object recognition almost always requires network transmission of images, and then transmission of recognition feedback results through the network, which is much less efficient in environments with relatively poor network conditions, such as the wild and crowded indoor environments.

In summary, the future development prospects of object recognition technology should not be limited to the recognition accuracy, recognition efficiency, algorithm robustness, etc., but also specific issues should be considered.

For objects with different characteristics and the environment in which the objects are located, designing corresponding recognition algorithms and considering the degree of dependence on network and equipment performance should also be a focus of future research.

References:

1. Link Reference(usage of image.darknet) :
   https://github.com/melissavanbussel/YouTube-Tutorials/commit/5e5cc1dd4f8de69f7b09e7fb96225589eba7d4f4

2. Link Reference(familiarizing with r tools) :
   https://cran.r-project.org/bin/windows/Rtools/rtools40.html

3. Link Reference(for image.darknet package) :
   https://pjreddie.com/darknet/install/

4. Link Reference(An introduction to Object recognition using deep learning) :
   https://machinelearningmastery.com/object-recognition-with-deep-learning/#:~:text=Object%20recognition%20is%20refers%20to,tasks%2C%20designed%20for%20model%20performance

5. Link Reference :
   https://www.mathworks.com/solutions/image-video-processing/object-recognition.html

6. Link Reference(Of all the uploaded images) :
   - https://assets.architecturaldigest.in/photos/62d66f2df8e893c06cec9970/master/pass/This%20Hyderabad%20home%20marries%20French-style%20architecture%20with%20vernacular%20interiors.jpg

   - https://encrypted-tbn2.gstatic.com/images?q=tbn:ANd9GcS6Xs7RWoPFCMgeWEGymHbNLOvukc2L39i_28Bg-2zxkhhqNqDR

   - https://encrypted-tbn0.gstatic.com/images?q=tbn:ANd9GcSaf_J39GGHQYx1BRInMDQq-xYS4Hnf0lfFZXVE6atQsYadbfp0

   - https://encrypted-tbn0.gstatic.com/images?q=tbn:ANd9GcTVGXDczi-5Y0FK6RGPoOZAz3gss5xwbmsibdL_ZrRgPNlDBC4m

   - https://encrypted-tbn1.gstatic.com/images?q=tbn:ANd9GcSNtaw7R07VMQf357V4IMaAowAhHvNr2IxejhYLTQvXGztG-ccA

Presented By:

S.NO	NAME	ROLLNUMBER	CONTRIBUTION	PERCENTAGE(%)
1.	Poojeetha	160620733001	Explanation about the topic	5.885
2.	Advaitha	160620733002	Problem Statement, References, Presented By	5.885
3.	Neha	160620733003	Introduction and Table of Contents, Installed 4.2.0v of R and checked for installation of package “image.darknet”	5.885
4.	Rithika	160620733006	Introduction and Table of Contents, Installed 4.2.2v of R and checked for installation of package “image.darknet”	5.885
5.	Hrishitha	160620733008	Other considerations and Applications	5.885
6.	Akshara	160620733010	Explanation about the topic	5.885
7.	Pranathi	160620733011	Introduction and Table of Contents, Installed 4.1.3v of R and checked for installation of package “image.darknet”	5.885
8.	Himaja	160620733018	Code with output, Installed 4.0.1v of R and checked for installation of package “image.darknet”	5.885
9.	Madhurima	160620733019	Explanation about the topic	5.885
10.	Aashritha	160620733020	Getting things ready, Finding errors and debugging	5.885
11.	Mallika	160620733021	Conclusion, Installed 4.0.3v of R and checked for installation of package “image.darknet”	5.885
12.	Sharvani	160620733022	Code with Output and Formatting of the whole document	5.885

13.	Ananya	160620733023	Explaining functions, libraries of the code	5.885
14.	Lavanya	160620733032	Conclusion	5.885
15.	Anaghaa	160620733043	Getting things ready, Finding errors and debugging	5.885
16.	Keerthana	160620733060	Other considerations and Applications	5.885
17.	Niharika	160620733302	Explaining functions, libraries of the code	5.885