

A
MAJOR PROJECT REPORT
On
Smart Civilization for Detecting Littering on Roads
BACHELOR OF TECHNOLOGY
In
COMPUTER SCIENCE AND ENGINEERING

Submitted by
(MIP-B10)

Student Name:

197Y5A0510 T.RAJU SWAROOP

Under the Guidance

of

Mr.Ch.V.V. NARASIMHA RAJU
Assistant Professor



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
MARRI LAXMAN REDDY
INSTITUTE OF TECHNOLOGY AND MANAGEMENT
(AUTONOMOUS)

(Affiliated to JNTU-H, Approved by AICTE New Delhi and Accredited by NBA & NAAC With 'A' Grade)

June 2022



MARRI LAXMAN REDDY **INSTITUTE OF TECHNOLOGY AND MANAGEMENT**

(AN AUTONOMOUS INSTITUTION)

(Approved by AICTE, New Delhi & Affiliated to JNTUH, Hyderabad)

Accredited by NBA and NAAC with 'A' Grade & Recognized Under Section 2(f) & 12(B) of the UGC act, 1956

CERTIFICATE

This is to certify that the project report titled “**Smart Civilization for Detecting Littering on Roads**” is being submitted by **T.RAJU SWAROOP (197Y5A0510)** in IV B.Tech II Semester **Computer Science & Engineering** is a record bonafide work carried out by him. The results embodied in this report have not been submitted to any other University for the award of any degree.

Internal Guide

HOD

Principal

External Examiner



MARRI LAXMAN REDDY **INSTITUTE OF TECHNOLOGY AND MANAGEMENT**

(AN AUTONOMOUS INSTITUTION)

(Approved by AICTE, New Delhi & Affiliated to JNTUH, Hyderabad)

Accredited by NBA and NAAC with 'A' Grade & Recognized Under Section 2(f) & 12(B) of the UGC act, 1956

DECLARATION

We hereby declare that the Major Project Report entitled, “**Smart Civilization for Detecting Littering on Roads**” submitted for the B. Tech degree is entirely our work and all ideas and references have been duly acknowledged. It does not contain any work for the award of any other degree.

Date:

SWAROOP
(197Y5A0510)



MARRI LAXMAN REDDY **INSTITUTE OF TECHNOLOGY AND MANAGEMENT**

(AN AUTONOMOUS INSTITUTION)

(Approved by AICTE, New Delhi & Affiliated to JNTUH, Hyderabad)

Accredited by NBA and NAAC with 'A' Grade & Recognized Under Section 2(f) & 12(B) of the UGC act, 1956

ACKNOWLEDGEMENT

We are happy to express my deep sense of gratitude to the principal of the college **Dr. K. Venkateswara Reddy**, Professor, Department of Computer Science and Engineering, Marri Laxman Reddy Institute of Technology & Management, for having provided us with adequate facilities to pursue our project.

We would like to thank **Mr. Abdul Basith Khateeb**, Assoc. Professor and Head, Department of Computer Science and Engineering, Marri Laxman Reddy Institute of Technology & Management, for having provided the freedom to use all the facilities available in the department, especially the laboratories and the library.

We are very grateful to my project guide **Mr. Ch. V. V. Narasimha Raju**, Assistant. Prof., Department of Information Technology, Marri Laxman Reddy Institute of Technology & Management, for his extensive patience and guidance throughout our project work.

We sincerely thank our seniors and all the teaching and non-teaching staff of the Department of Computer Science for their timely suggestions, healthy criticism and motivation during the course of this work.

We would also like to thank my classmates for always being there whenever we needed help or moral support. With great respect and obedience, we thank our parents and siblings who were the backbone behind our deeds.

Finally, we express our immense gratitude with pleasure to the other individuals who have either directly or indirectly contributed to our need at right time for the development and success of this work.



MARRI LAXMAN REDDY INSTITUTE OF TECHNOLOGY AND MANAGEMENT

(AN AUTONOMOUS INSTITUTION)

(Approved by AICTE, New Delhi & Affiliated to JNTUH, Hyderabad)

Accredited by NBA and NAAC with 'A' Grade & Recognized Under Section 2(f) & 12(B) of the UGC act, 1956

	CONTENTS	
S NO.	TITLE	PAGE NO.
	ABSTRACT	Vi
	LIST OF FIGURES	Vii
	LIST OF TABLES	Viii
	SYMBOLS & ABBREVIATIONS	IX
1	INTRODUCTION	1
	1.1 Purpose	1
	1.2 Scope	2
	1.3 Features	2
	1.4 Problem Definition	3
	1.5 Organization of Document	3
2	SYSTEM ANALYSIS	4
	2.1 Existing System	4
	2.2 Proposed System	4
	2.3 Advantages	4
	2.4 Modules and Functionalities	5
	2.5 Feasibility Study	6
	2.6 Conclusion	7
3	SYSTEM DESIGN	8
	3.1 System Design	9
	3.2 Software Design	9
	3.3 Input/Output Design	10

	CONTENTS	
S NO.	TITLE	PAGE NO.
	3.4 Software Requirement Specifications (SRS)	10
	3.5 UML Concepts	11
	3.6 Hardware and Software Requirements	22
4	IMPLEMENTATION AND CODING	23
	4.1 Object Detection	23
	4.2 Extreme learning machine	24
	4.3 Haar cascade features	26
	4.4 Python programming	26
	4.5 Coding	29
5	TESTING	30
	5.1 Software Testing	30
	5.2 Testing Design	31
	5.3 Test cases	32
	5.4 Power bi	33
6	LIST OF IMAGES	34
	6.1 India Table	34
	6.2 Rural and urban current population	35
	6.3 Population growth analysis	36
	6.4 Future Data	37
	6.5 Training images from cascade trainer	39
	6.6 Litter detection/garbage detection output	41
7	ADVANCED USE CASES OF OPEN CV	42
	7.1 Open cv in tesla cars	46
	7.2 Real time face recognition with open cv	47
	7.3 How faces are detected	54
8	ADVANCED USE CASES OF POWER BI	55

	CONTENTS	
S NO.	TITLE	PAGE NO.
	8.1 Dash boards	56
	8.2 Advanced analytics with power bi	65
9	Smart civilization planning for future estimation of waste production	66
	9.1 Our dash board	66
	9.2 Selection of year	67
	9.3 Visual for total production	67
	9.4 Graph of future population	68
	9.5 visual for waste production kg/person	68
10	Detecting garbage with computer vision	69
	10.1 validation table	69
11	CONCLUSION & FUTURE ENHANCEMENT	70
12	REFERENCES	71

ABSTRACT

OpenCV is the huge and open-source library for image processing, machine learning and computer vision. It is also playing an important role in real-time operation. With the help of the OpenCV library, we can easily process the images as well as videos to identify the objects, faces or even handwriting of a human present in the file. We will only focus to object detection from images using OpenCV. We will learn about how we can use OpenCV to do object detection from a given image using a Python program. Object detection is a modern computer technology that is related to image processing, deep learning and computer vision to detect the objects present in an image file. All the technologies used in the Object detection technique (as we mentioned earlier) deals with detecting instances of the object in the image or video. We will first import the OpenCV library in the Python program, and then we will use functions to perform object detection on an image file given to us. But, before using and importing the library functions, let's first install the requirements for using the Object detection technique. We will detect the garbage detection based on civilization in Rural and Urban areas. We had created a dashboard where it shows graph about where the garbage is more by this it is easy to clear it. The requirement to perform object detection using the OpenCV library is that the OpenCV library should be present in our device so that we can import it into a Python program and use its object detection functions. If this library is not present in our system, we can use the following command in our command prompt terminal to install it. Matplotlib is very helpful in the opening, closing, reading etc., images in a Python program, and that's why the installation of this library for object detection becomes an important requirement. If the matplotlib library is not present in our system, we have to use the following command in our command prompt terminal to install it.

LIST OF FIGURES

FIG. NO	FIG. NAME	PAGE NO.
3.a	System Architecture Design	9
3.b	Application Architecture Design	9
3.c	Class Diagram	18
3.d	Use Case Diagram	19
3.e	Activity Diagram	20
3.f	Phases in garbage collection	21
5.a	Test cases	32
6.a	India Table	34
6.b	Future Data	37
6.c	Cascade trainer Gui outputs	39
6.d	Output of litter detection code	40
7	Object detection	45
7.1	Detecting objects on roads	46
7.2	Face recognition	48
8.1	Process of prediction	61
8.2	Output of Dax code	63
9.1	Dash board	65

SYMBOLS & ABBREVIATIONS

HTML	: Hyper Text Markup Language
XML	: Extensible Markup Language.
Python	: coding.
OpenCV	: Object detection.
Matplotlib	: Graphical representation.
PNG	: Portable Network Graphics.
ImRead	: Function used to open the image file

CHAPTER 1

INTRODUCTION TO PROJECT

Basically, the Haar cascade technique is an approach based on machine learning where we use a lot of positive and negative images to train the classifier to classify between the images. Haar cascade classifiers are considered as the effective way to do object detection with the OpenCV library. Now, let's understand the concept of positive and negative images.

Positive Images: These are the images that contain the objects which we want to be identified from the classifier.

Negative Images: These are the images that do not contain any object that we want to be detected by the classifier, and these can be images of everything else.

I aim to raise people's awareness and encourage them to recycle so that the oceans, seas, or the environment do not become more polluted and damaged in our increasingly polluted world. This project is aimed to detect recyclable objects such as cardboard, paper, plastic, and metal.

1.1 PURPOSE OF THE PROJECT

The purpose of this project is to help keep the cities clean. And it is also help in reducing the plastic wastage which affects the environment. Reducing solid waste is reducing the amount of trash that goes to landfills. This project shows the detail information regarding how much percentage of trash is present in an area using the number of people living. It also helps in clear the oceans, seas wastage by using garbage detector. In this project we will live capture the images and check whether the place has litter on roads or not.

1.2 SCOPE OF THE PROJECT

Smart Civilization for Detecting Littering on Roads Project is mostly used to reduce the garbage on roads which helps people regarding health and helps in saving environment. Major scope is to reduce the garbage and helps in good environment.

1.3 FEATURES OF THE PROJECT

Object detection, which not only requires accurate classification of objects in images but also needs accurate location of objects is an automatic image detection process based on statistical and geometric features. The accuracy of object classification and object location is important indicators to measure the effectiveness of model detection. Object detection is widely used in intelligent monitoring, military object detection, UAV navigation, unmanned vehicle, and intelligent transportation. However, because of the diversity of the detected objects, the current model fails to detect objects. The changeable light and the complex background increase the difficulty of the object detection especially for the objects that are in the complex environment. The traditional method of image classification and location by multiscale pyramid method needs to extract the statistical features of the image in multiscale and then classify the image by a classifier. Because different types of images are characterized by different features, it is difficult to use one or more features to represent objects, which do not achieve a robust classification model. Those models failed to detect the objects especially that there are more detected objects in an image.

- Identifies the trash items by video capture.
- Use python packages like OpenCV and Matplotlib.
- Follows the Haar Cascade Technique.
- Implementing code for detecting images.
- Use of detect Multiscale function.
- Creating an xml file for cascade the classifier.
- Giving labels to the image like paper, metal, plastic etc.

1.4 PROBLEM DEFINITION

The threat of waste to the environment, health and safety is huge. And so are the financial and social ramifications, waste experts say. Pollution runs into rivers and seeps into ground water. Flooding is caused by garbage clogging drains, and the atmosphere can be poisoned by the toxic discharge from trash. In normal garbage detection system, we only detect the litter by using datasets or video capture to know the type of the item like plastic, paper etc. But in proposed system we used a dashboard using power bi where it is used to predict the percentage of garbage on roads in form of graph where it helps to reduce the garbage on roads by knowing how much level of garbage on roads. It shows the difference in rural and urban. It gives information regarding population.

Problems arising in the existing system:

- 1.It can only detect the garbage on roads?
- 2.How can we find the percentage of garbage and give the graphical representation?

SOLUTION: The solution we created a dashboard where we can find the graphical representation in dashboard and shows the percentage of garbage (or) Litter on roads in a particular area.

1.5 ORGANIZATION OF DOCUMENT

In this project documentation we have initially put the definition and objective of the project as well as the design of the project which is followed by the implementation and testing phases. The project has been concluded successfully and the future enhancements of the project also given in this documentation.

CHAPTER 2

SYSTEM ANALYSIS

2.1 EXISTING SYSTEM

In Existing system, we only detect the litter by using datasets or video capture to know the type of the item like plastic, paper etc.

2.2 PROPOSED SYSTEM

In this proposed system, we not only detect the litter by using datasets or video capture to know the type of the item like plastic, paper etc. But we also created a dashboard which helps in knowing the percentage of garbage in an area. This helps in knowing how negligence the people in that area.

2.3 ADVANTAGES

This proposed system has following advantages:

- Helps in knowing the area which has more litter on roads and taking necessity upon them.
- Knowing the percentage of litter.

2.4 MODULES AND FUNCTIONALITIES

Modules are:

- 1) Imread
- 2) Detect Multiscale
- 3) OpenCV and Matplotlib
- 4) Imshow
- 5) Cascade Classifier

2.4.1 Imread Function in python:

The imread function is used to open the image file (PNG) in cv2. The image which is used for detecting must be opened that is done by imread () function.

2.4.2 Detect Multiscale Function:

We use the detect Multiscale () function with the imported cascade file to detect the object present in the image or not.

2.4.3 OpenCV and Matplotlib:

OpenCV library should be present in our device so that we can import it into a Python program and use its object detection functions. Matplotlib is very helpful in the opening, closing, reading etc., images in a Python program

2.4.4 Imshow Function:

It is used to display the processed image using the plt show () and imshow () function. This function is a part of Matplotlib library. This function helps in visualize the output screen.

2.4.5 Cascade Classifier:

Cascading classifiers are trained with several hundred "positive" sample views of a particular object and arbitrary "negative" images of the same size. After the classifier is trained it can be applied to a region of an image and detect the object in q Cascade classifiers are available in OpenCV, with pre-trained cascades for frontal faces and upper body. Training a new cascade in OpenCV is also possible with either haartraining or train cascades methods. This can be used for rapid object detection of more specific targets, including non-human objects with Haar-like features question.

2.5 FEASIBILITY STUDY

The next step in analysis is to verify the feasibility of the proposed system. “All projects are feasible given unlimited resources and infinite time“. But in reality, both resources and time are scarce. Project should confirm to time bound and should be optimal in their consumption of resources. This place a constant is approval of any project.

- Technical feasibility
- Operational feasibility
- Economic feasibility

2.5.1 Technical Feasibility:

To determine whether the proposed system is technically feasible, we should take into consideration the technical issues involved behind the system. This Application uses the web technologies, which is rampantly employed these days worldwide. The world without the web is incomprehensible today. That goes to proposed system is technically feasible

2.5.2 Operational Feasibility:

To determine the operational feasibility of the system we should take into consideration the awareness level of the users. This system is operational feasible since the users are familiar with the technologies and hence there is no need to gear up the personnel to use system. Also the system is very friendly and to use.

2.5.3 Economic Feasibility:

To decide whether a project is economically feasible, we have to consider various Factors as:

- Cost benefit analysis
- Long-term returns
- Maintenance costs

It requires average computing capabilities and access to internet, which are very basic requirements and can be afforded by any organization hence it doesn't incur additional economic overheads, which renders the system economically feasible.

2.6 CONCLUSION

In this phase, we understand the modules and functionalities of the system. We arrange all the required components to develop the project in this phase itself so that we will have a clear idea regarding the requirements before designing the project. Thus, we will proceed to the design phase followed by the implementation phase of the project. During our case studies, we found some other challenges in the litter object detection and classification. Although we have successfully detected and classified 11 classes of litters, we still faced some challenges like annotation approach for the almost free forms objects and assessment for cleanliness according to the detection results. In the future work, we need more data sets to train the model to detect multi-scale objects on the street and develop a scalable, sustainable, edge-based service system.

CHAPTER 3

SYSTEM DESIGN

3.1 SYSTEM DESIGN

System design is transition from a user-oriented document to programmers or data base personnel. The design is a solution, how to approach to the creation of a new system. This is composed of several steps. It provides the understanding and procedural details necessary for implementing the system recommended in the feasibility study. Designing goes through logical and physical stages of development, logical design reviews the present physical system, prepare input and output specification, details of implementation plan and prepare a logical design walkthrough.

The database tables are designed by analyzing functions involved in the system and format of the fields is also designed. The fields in the database tables should define their role in the system. The unnecessary fields should be avoided because it affects the storage areas of the system. Then in the input and output screen design, the design should be made user friendly. The menu should be precise and compact.

3.2 SOFTWARE DESIGN

In designing the software following principles are followed:

- ❖ **Modularity and partitioning:** software is designed such that, each system should consist of hierarchy of modules and serve to partition into separate function.
- ❖ **Coupling:** modules should have little dependence on other modules of a system.
- ❖ **Cohesion:** modules should carry out in a single processing function.
- ❖ **Shared use:** avoid duplication by allowing a single module be called by other that need the function it provides

3.2.1 System Architecture Design:

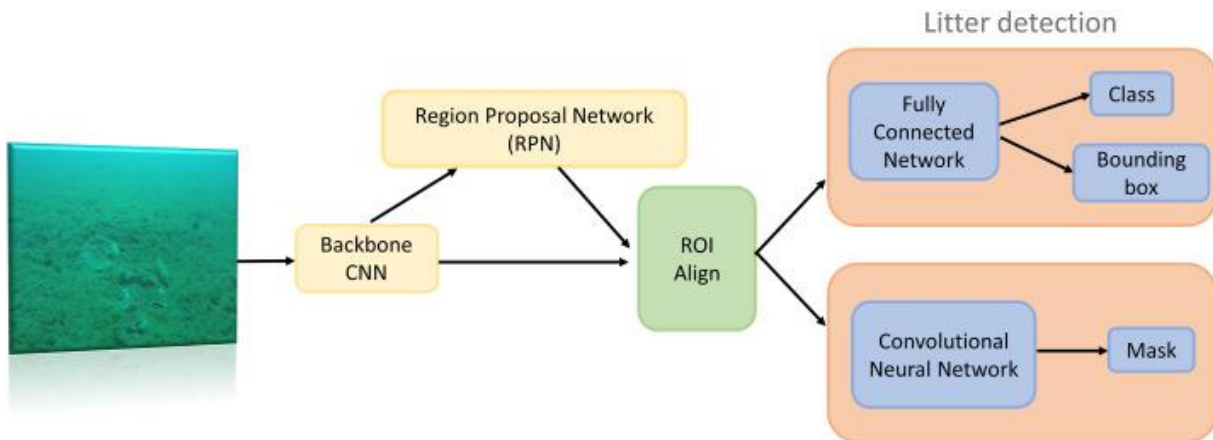


Fig:3. a System Architecture Design

APPLICATION ARCHITECTURE DESIGN

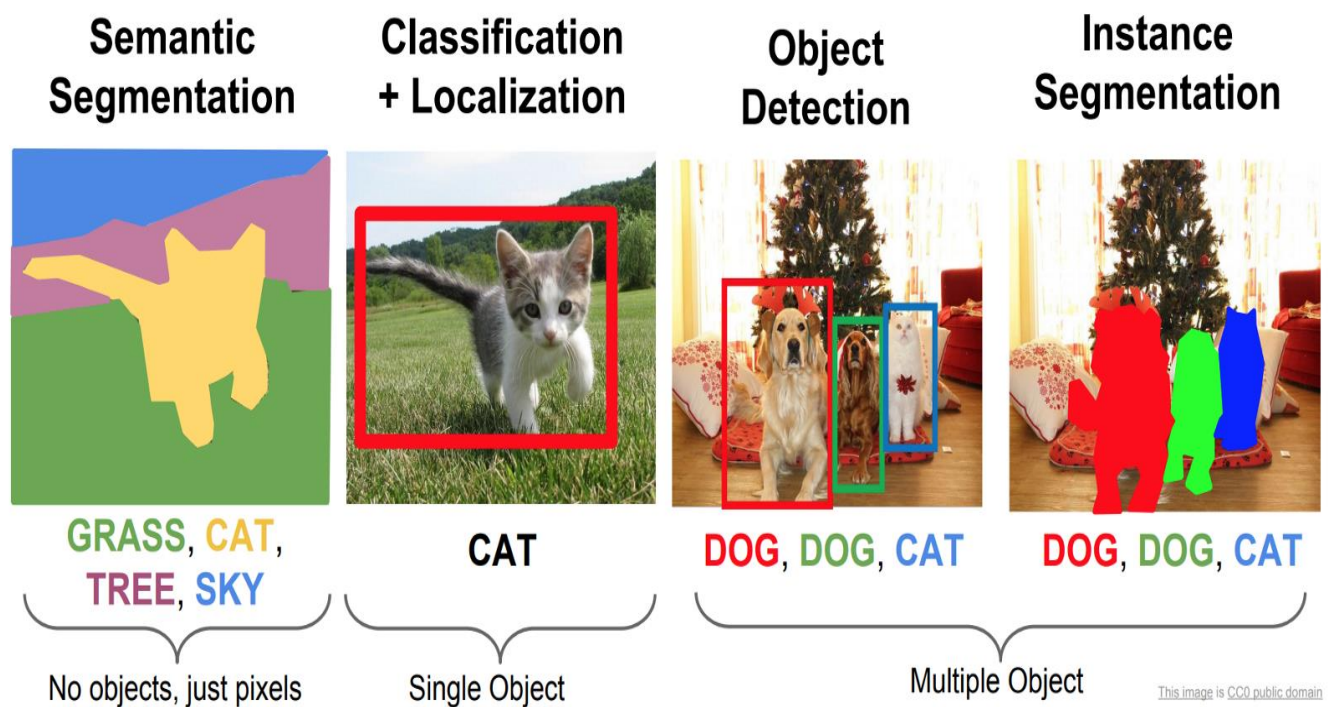


Fig:3. b Application Architecture Design

3.3 INPUT/OUTPUT DESIGN

3.3.1 Input Design:

Considering the requirements, procedures to collect the necessary input data in most efficiently designed. The input design has been done keeping in view that, the interaction of the user with the system being the most effective and simplified way.

Also, the measures are taken for the following

- Avoid unauthorized access
- Eliminating extra steps
- Keeping the process simple
- At this stage the sender and receiver action is designed

3.3.2 Output Design:

All the screens of the system are designed with a view to provide the user with easy operations in simpler and efficient way, minimum key strokes possible. Instructions and important information is emphasized on the screen. Almost every screen is provided with no error and important messages and option selection facilitates. Emphasis is given for speedy processing and speedy transaction between the screens. Each screen assigned to make it as much user friendly as possible by using interactive procedures. So, to say user can operate the system without much help from the operating manual.

3.4 SOFTWARE REQUIREMENT SPECIFICATIONS

3.4.1 What is SRS?

Software Requirement Specification (SRS) is the starting point of the software developing activity. As system grew more complex it became evident that the goal of the entire system cannot be easily comprehended. Hence the need for the requirement phase arose. The software project is initiated by the client needs. The SRS is the means of translating the ideas of the minds of clients (the input) into a formal document (the output of the requirement phase.) The SRS phase consists of two basic activities:

3.4.2 Problem/Requirement Analysis:

The process is order and more nebulous of the two, deals with understand the problem, the goal and constraints.

3.4.3 Requirement Specification:

Here, the focus is on specifying what has been found giving analysis such as representation, specification languages and tools, and checking the specifications are addressed during this activity.

The Requirement phase terminates with the production of the validate SRS document. Producing the SRS document is the basic goal of this phase.

3.4.4 Document Conventions:

We have used Times New Roman (text size 12). Bold Font is used for Main Headings (text size of 16). Normal font is used for sub headings (text size of 14).

Font: Times New Roman

Main Heading: Bold Font

3.5 UML CONCEPTS:

The Unified Modeling Language (UML) is a standard language for writing software blue prints. The UML is a language for

- Visualizing
- Specifying
- Constructing
- Documenting the artifacts of a software intensive system.

The UML is a language which provides vocabulary and the rules for combining words in that vocabulary for the purpose of communication. A modeling language is a language whose vocabulary and the rules focus on the conceptual and physical representation of a system. Modeling yields an understanding of a system.

3.5.1 Building Blocks of the UML

The vocabulary of the UML encompasses three kinds of building blocks:

- ❖ Things
- ❖ Relationships
- ❖ Diagrams

Things are the abstractions that are first-class citizens in a model; relationships tie these things together; diagrams group interesting collections of things.

3.5.2 Things in the UML

There are four kinds of things in the UML:

- Structural things
- Behavioral things
- Grouping things
- Annotational things

❖ **Structural things:** These are the nouns of UML models. The structural things used in the project design are first, a **class** is a description of a set of objects that share the same attributes, operations, relationships and semantics.

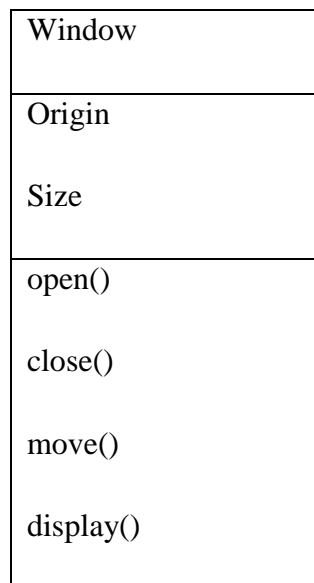
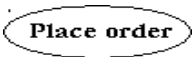


Fig 3.c: Classes

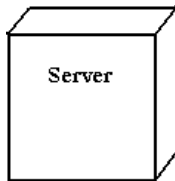
Second, a use case is a description of set of sequence of actions that a system performs that yields an observable result of value to particular actor.



Place order

Use Cases

Third, a node is a physical element that exists at runtime and represents a computational resource, generally having at least some memory and often processing capability.

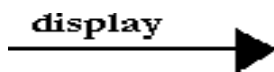


Nodes

Behavioral things are the dynamic parts of UML models. The behavioral thing used is:

Interaction

An interaction is a behavior that comprises a set of messages exchanged among a set of objects within a particular context to accomplish a specific purpose. An interaction involves a number of other elements, including messages, action sequences (the behavior invoked by a message, and links (the connection between objects).



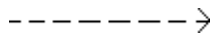
Messages

3.5.3 Relationships in the UML

There are four kinds of relationships in the UML:

- ❖ Dependency
- ❖ Association
- ❖ Generalization
- ❖ Realization

A **dependency** is a semantic relationship between two things in which a change to one thing may affect the semantics of the other thing (the dependent thing).



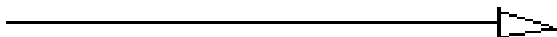
❖ **Dependencies:**

An association is a structural relationship that describes a set links, a link being a connection among objects. Aggregation is a special kind of association, representing a structural relationship between a whole and its parts.



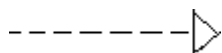
❖ **Association:**

A generalization is a specialization/ generalization relationship in which objects of the specialized element (the child) are substitutable for objects of the generalized element (the parent).



❖ **Generalization:**

A realization is a semantic relationship between classifiers, where in one classifier specifies a contract that another classifier guarantees to carry out.



❖ **Realization:**

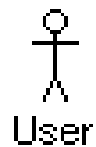
3.5.4 Sequence Diagrams

UML sequence diagrams are used to represent the flow of messages, events and actions between the objects or components of a system. Time is represented in the vertical direction showing the sequence of interactions of the header elements, which are displayed horizontally at the top of the diagram.

Sequence Diagrams are used primarily to design, document and validate the architecture, interfaces and logic of the system by describing the sequence of actions that need to be performed to complete a task or scenario. UML sequence diagrams are useful design tools because they provide a dynamic view of the system behavior which can be difficult to extract from static diagrams or specifications.

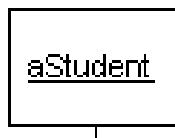
❖ **Actor:**

Represents an external person or entity that interacts with the system



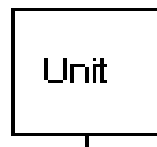
❖ **Object:**

Represents an object in the system or one of its components



❖ **Unit:**

Represents a subsystem, component, unit, or other logical entity in the system (may or may not be implemented by objects)



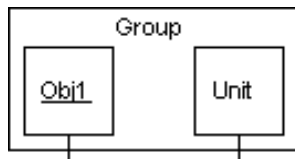
❖ **Separator:**

Represents an interface or boundary between subsystems, components or units (e.g., air interface, Internet, network)



❖ **Group:**

Groups related header elements into subsystems or components.



3.5.5 Sequence Diagram Body Elements

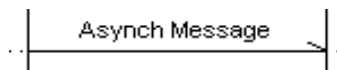
❖ Action:

Represents an action taken by an actor, object or unit.



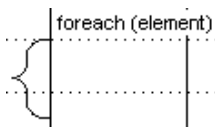
❖ Asynchronous Message:

An asynchronous message between header elements



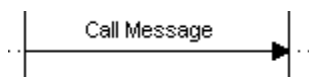
❖ Block:

A block representing a loop or conditional for a particular header element



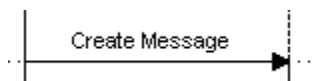
❖ Call Message:

A call (procedure) message between header elements



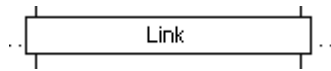
❖ Create Message:

A "create" message that creates a header element (represented by lifeline going from dashed to solid pattern)

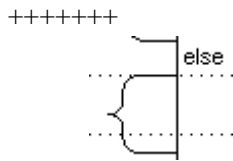


❖ Diagram Link:

Represents a portion of a diagram being treated as a functional block. Similar to a procedure.



Else Block Represents an "else" block portion of a diagram block



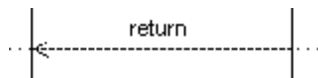
- **Message**

A simple message between header elements



- **Return Message**

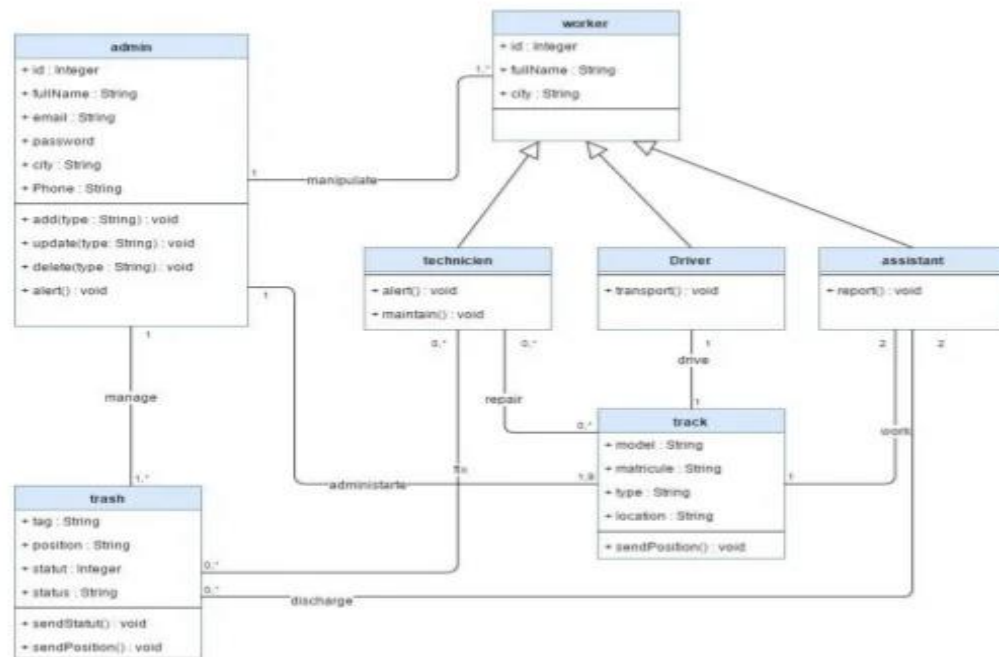
A return message between header elements



3.5.6 Class Diagram:

A class diagram in the Unified Modelling Language is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among objects. Class diagram is a static diagram. It represents the static view of an application. Class diagram describes the attributes and operations of a class and the constraints imposed on the system.

Class diagram:



class diagram

Fig 3.c: Class Diagram

3.5.7 Use Case Diagram:

A use case diagram at its simplest is a representation of a user's interaction with the system that shows the relationship between the user and the different use cases in which the user is involved. A use case diagram can identify the different types of users of a system and the different use cases and will often be accompanied by other types of diagrams as well. The use cases are represented by either circles or ellipses.

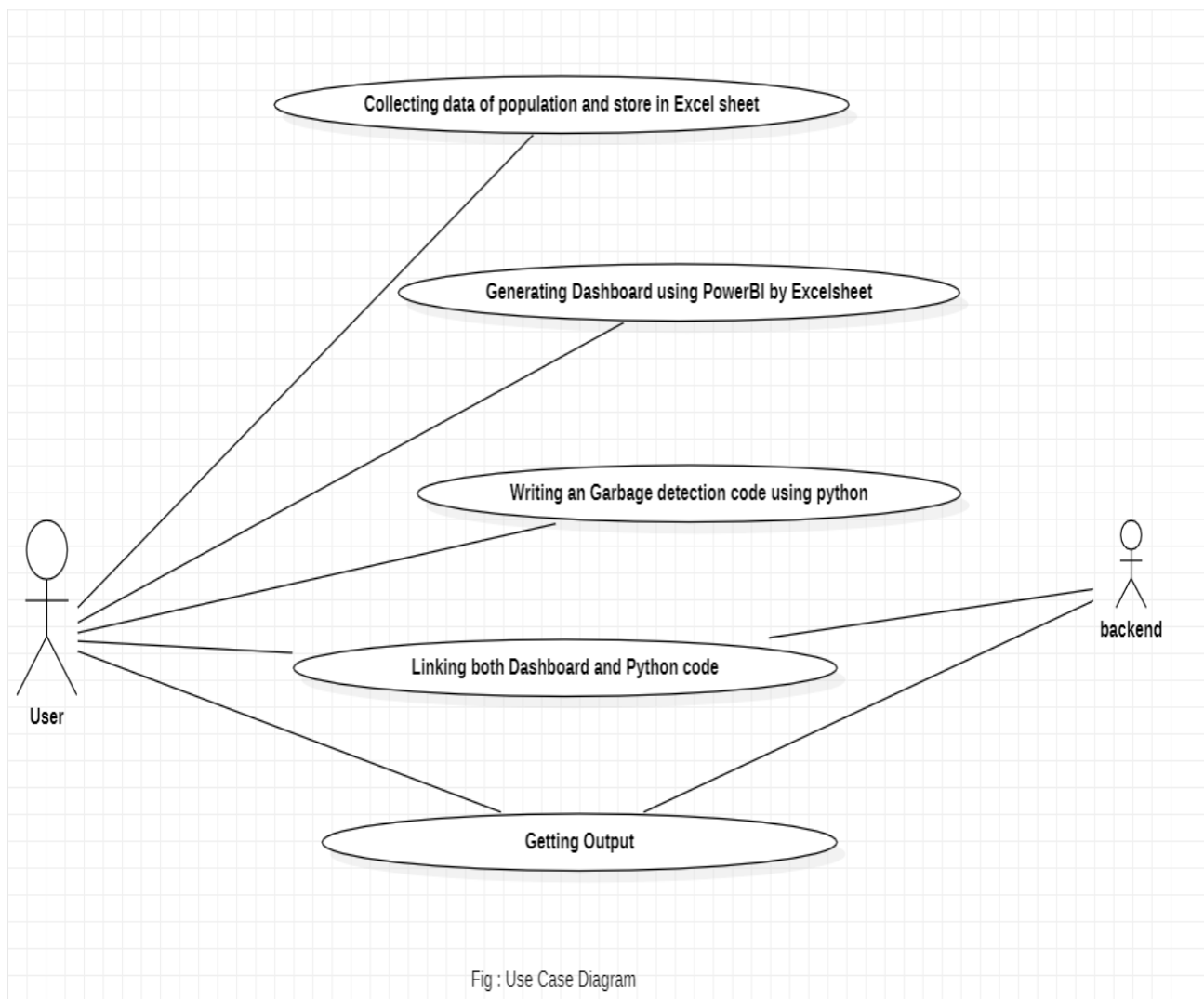


Fig 3.d: Use Case Diagram

3.5.8 Activity diagram:

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modeling Language, activity diagrams are intended to model both computational and organizational processes (i.e., workflows) as well as the data flows intersecting with the related activities.

Activity diagram:

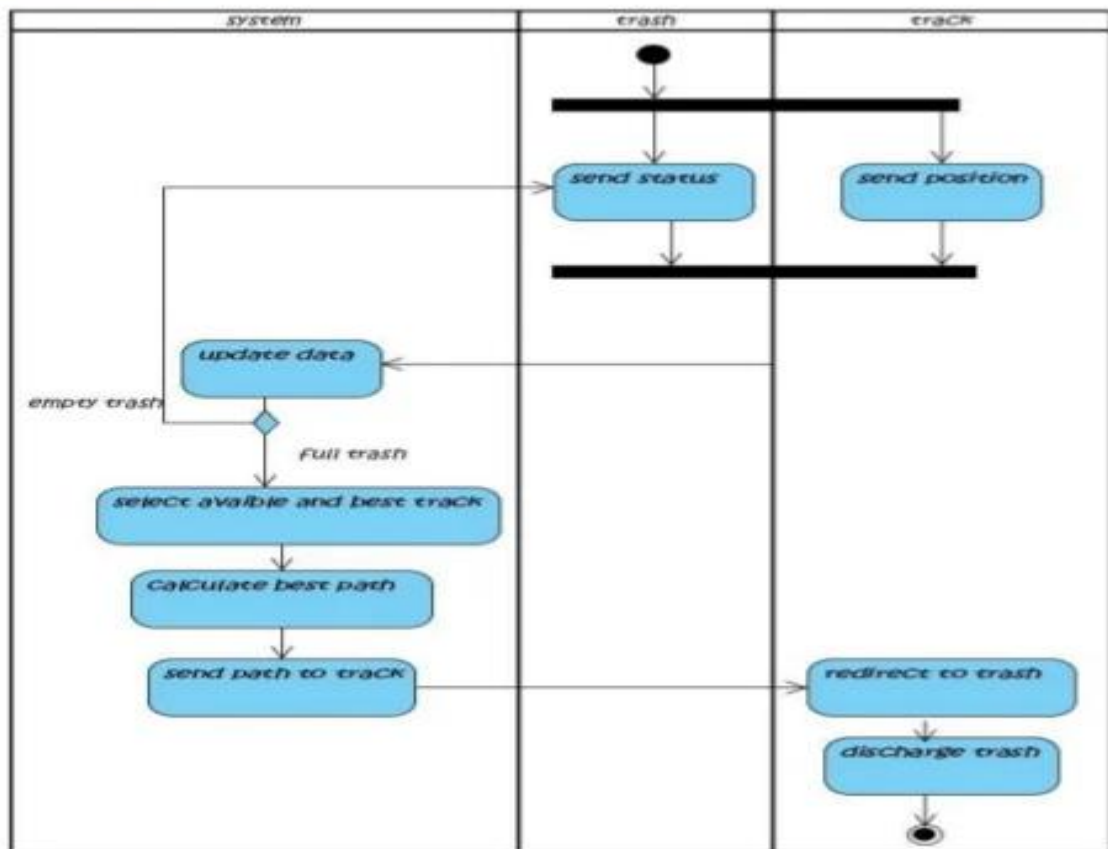


Fig 3.e: Activity Diagram

3.5.9 Phases in Garbage collection:

Marking Phase: A list of all the live objects is created during the marking phase. This is done by following the references from all the root objects. All of the objects that are not on the list of live objects are potentially deleted from the heap memory.

Relocating Phase: The references of all the objects that were on the list of all the live objects are updated in the relocating phase so that they point to the new location where the objects will be relocated to in the compacting phase.

Compacting Phase: The heap gets compacted in the compacting phase as the space occupied by the dead objects is released and the live objects remaining are moved. All the live objects that remain after the garbage collection are moved towards the older end of the heap memory in their original order.

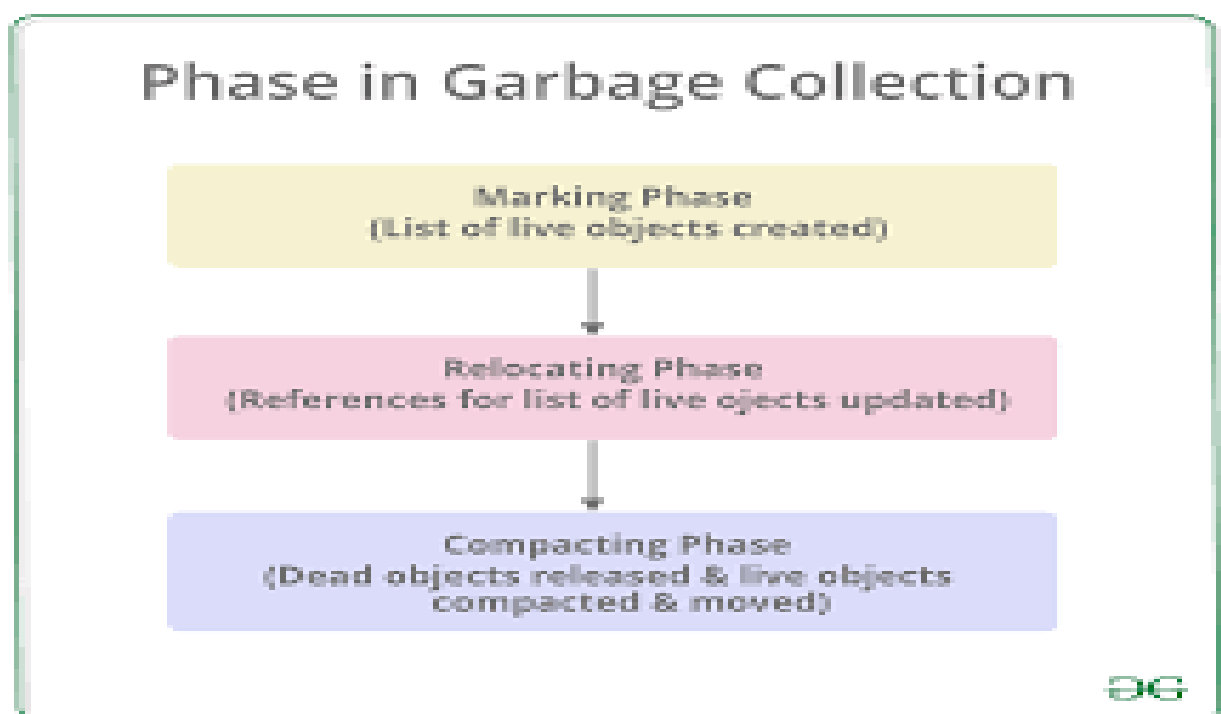


Fig 3.f: Phases in Garbage detection

3.6 HARDWARE AND SOFTWARE REQUIREMENTS:

3.6.1 Software Interfaces:

We use Python IDE and PyCharm for executing Python code which we have writing for Garbage Detection. We collect Data of past and Future and store in Excel. And Creating a Dashboard using PowerBI for Detail information of garbage present in an area.

3.6.2 Software Requirements:

- Python PyCharm developer edition.
- OpenCV and Matplotlib libraries.
- Python IDE.
- Operating System - Windows XP/7/8/10.
- PowerBI Application.
- MS Excel.

3.6.3 Hardware Requirements:

- Ram: : 1GB Ram and above
- Hard Disk : 50GB and above
- Processor : Dual core and above

CHAPTER 4

IMPLEMENTATION AND CODING

4.1 Object Detection

4.1.1 OpenCV:

OpenCV (Open-source Computer Vision Library) is an open-source computer vision and machine learning software library. OpenCV was built to provide a common infrastructure for computer vision applications and to accelerate the use of machine perception in the commercial products. Being a BSD-licensed product, OpenCV makes it easy for businesses to utilize and modify the code.

4.2.1.1 Matplotlib:

Matplotlib is a plotting library for the Python programming language and its numerical mathematics extension NumPy. It provides an object-oriented API for embedding plots into applications using general-purpose GUI toolkits like Tkinter, wxPython, Qt, or GTK. There is also a procedural "pylab" interface based on a state machine (like OpenGL), designed to closely resemble that of MATLAB, though its use is discouraged. SciPy makes use of Matplotlib. Matplotlib was originally written by "John D. Hunter".

4.2.1.2 Video capture using OpenCV:

Python provides various libraries for image and video processing. One of them is OpenCV. OpenCV is a vast library that helps in providing various functions for image and video operations. With OpenCV, we can capture a video from the camera. It lets you create a video capture object which is helpful to capture videos through webcam and then you may perform desired operations on that video.

4.2.1.3 Steps to Capture a video:

- Use `cv2.VideoCapture()` to get a video capture object for the camera.
- Set up an infinite while loop and use the `read ()` method to read the frames using the above created object.
- Use `cv2.imshow()` method to show the frames in the video.
- Breaks the loop when the user clicks a specific key.

4.2.1.4 wait key () function:

wait key () function of Python OpenCV allows users to display a window for given milliseconds or until any key is pressed. It takes time in milliseconds as a parameter and waits for the given time to destroy the window, if 0 is passed in the argument it waits till any key is pressed.

Syntax: cv2.waitKey(1)

4.2.1.5 Result:

It gives the detection output and get detail analysis in dashboard.

4.2 EXTREME LEARNING MACHINE, CSS, HAAR CASCADE CLASSIFIER

4.2.1 Extreme Learning Machine:

Extreme Learning Machine (ELM) is a single-layer forward feedback neural network proposed by “Huang et al”. The input weights and hidden layer biases are all generated by program randomization instead of setting artificially. The output weights are determined by least squares regression. Therefore, the whole training process only needs to specify the number of hidden layer neurons and the activation function. Compared with the traditional neural network algorithm, the training speed is faster and the parameter selection is more flexible. At present, ELM has demonstrated its superior performance in many fields,

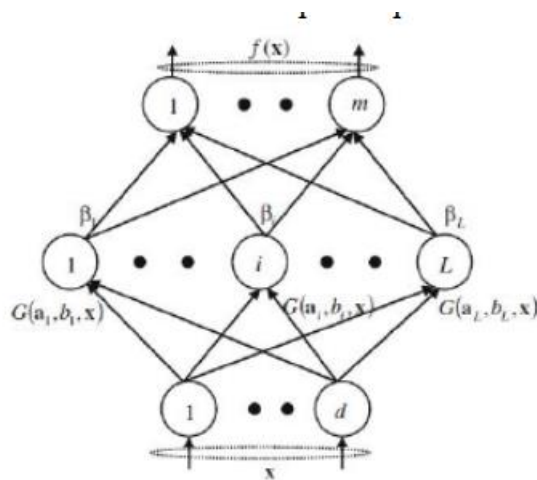


Fig.2. Single-layer ELM structure. ($m=2$, $L=20$, $d=28$).

4.2.2 Detect Multiscale Function:

detect Multiscale function is used to detect the faces. This function will return a rectangle with coordinates (x, y, w, h) around the detected face.

It takes 3 common arguments — the input image, scale Factor, and minNeighbours.

- **Scale Factor:** It specifies how much the image size is reduced with each scale. In a group photo, there may be some faces which are near the camera than others. Naturally, such faces would appear more prominent than the ones behind. This factor compensates for that.
- **minNeighbours:** It specifies how many neighbours each candidate rectangle should have to retain it. You can read about it in detail [here](#). You may have to tweak these values to get the best results. This parameter specifies the number of neighbours a rectangle should have to be called a face. We obtain these values after trial and test over a specific range.

4.2.3 Haar Cascade:

Haar-like feature is a simple rectangular feature introduced in the face detection system by Viola et al. and named after the Haar wavelet. Lienhart R et al. extended it further by adding rectangular features with a rotation of 45° . The extended features are roughly divided into three types: edge features, line features, center-surround features (as shown in Fig. 1). The Haar-like feature can effectively reflect the local gray change information of the image, and can also be quickly calculated through the integral image. In this paper, we select linear features and center-surround features as templates. The linear features Fig.1 (2a) -(2b) are computed in the dimensions of 2×3 and 2×4 , respectively. They are rotated at 45° and 90° to get new features Fig.1 (2c) -(2h). The feature Fig.1 (3a) uses a 3×3 window, which is also rotated through 45° to get feature Fig.1 (3b). Finally, it is calculated using the integral image and 10 feature values are extracted for each pixel.

4.3.3.1 Haar Cascade features:

1. Edge features



2. Line features



3. Center-surround features



4.3 PYTHON PROGRAMMING:

Python is an interpreted high-level general-purpose programming language. Its design philosophy emphasizes code readability with its use of significant indentation. Its language constructs as well as its object-oriented approach aim to help programmers write clear, logical code for small and large-scale projects. Python is dynamically-typed and garbage-collected. It supports multiple programming paradigms, including structured (particularly, procedural), object-oriented and functional programming. It is often described as a "batteries included" language due to its comprehensive standard library. Guido van Rossum began working on Python in the late 1980s, as a successor to the ABC programming language, and first released it in 1991 as Python 0.9.0.[33] Python 2.0 was released in 2000 and introduced new features, such as list comprehensions and a garbage collection system using reference counting. Python 3.0 was released in 2008 and was a major revision of the language that is not completely backward-compatible. Python 2 was discontinued with version 2.7.18 in 2020. Python consistently ranks as one of the most popular programming languages.

4.3.1 cv2.imread() method:

OpenCV-Python is a library of Python bindings designed to solve computer vision problems. cv2.imread() method loads an image from the specified file. If the image cannot be read (because of missing file, improper permissions, unsupported or invalid format) then this method returns an empty matrix.

Syntax: cv2.imread(path, flag)

4.4.2 Parameters:

- **Path:** A string representing the path of the image to be read.
- **Flag:** It specifies the way in which image should be read. The default value is cv2.IMREAD_COLOR
- **Return Value:** This method returns an image that is loaded from the specified file.

4.4.3 Types of flags:

- **cv2.IMREAD_COLOR:** It specifies to load a color image. Any transparency of image will be neglected. It is the default flag. Alternatively, we can pass integer value 1 for this flag.
- **cv2.IMREAD_GRAYSCALE:** It specifies to load an image in grayscale mode. Alternatively, we can pass integer value 0 for this flag.
- **cv2.IMREAD_UNCHANGED:** It specifies to load an image as such including alpha channel. Alternatively, we can pass integer value -1 for this flag.

4.6 CODING:

4.6.1 Capturing.py:

```
import cv2
import numpy as np
import imutils
import os
Datos = 'p'
if not os.path.exists(Datos):
    print('Carpeta creada: ', Datos)
    os.makedirs(Datos)
cap = cv2.VideoCapture(1,cv2.CAP_DSHOW)
x1, y1 = 190, 80
x2, y2 = 450, 398
count = 0
while True:
    ret, frame = cap.read()
    if ret == False: break
    imAux = frame.copy()
    cv2.rectangle(frame,(x1,y1),(x2,y2),(255,0,0),2)
    objeto = imAux[y1:y2,x1:x2]
    objeto = imutils.resize(objeto, width=38)
    # print(objeto.shape)
    k = cv2.waitKey(1)
    if k == 27:
        break
    if k == ord('s'):
        cv2.imwrite(Datos+'/objeto_{}.jpg'.format(count),objeto)
        print('Imagen almacenada: ', 'objeto_{}.jpg'.format(count))
        count = count + 1
    cv2.imshow('frame',frame)
    cv2.imshow('objeto',objeto)
cap.release()
cv2.destroyAllWindows()
```

4.6.2 Detector.py:

```
import cv2
cap = cv2.VideoCapture(1,cv2.CAP_DSHOW)
Classif = cv2.CascadeClassifier('garbage.xml')
while True:
    ret,frame = cap.read()
    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
    waste = Classif.detectMultiScale(gray,
    scaleFactor = 10,
    minNeighbors = 150,
    minSize=(95,95))
    for (x,y,w,h) in waste:
        cv2.rectangle(frame, (x,y),(x+w,y+h),(0,255,0),2)
        cv2.putText(frame,'litter',(x,y-10),2,0.7,(0,255,0),2,cv2.LINE_AA)
    cv2.imshow('frame',frame)
    if cv2.waitKey(1) == 27:
        break
cap.release()
cv2.destroyAllWindows()
```

4.6.3 CapturingObject.py:

```
import cv2
import numpy as np
import imutils
import os
Datos = 'n'
if not os.path.exists(Datos):
    print('folder created: ', Datos)
    os.makedirs(Datos)
cap = cv2.VideoCapture(1,cv2.CAP_DSHOW)
x1, y1 = 190, 80
x2, y2 = 450, 398
count = 0
while True:
```

```

ret, frame = cap.read()
if ret == False: break
imAux = frame.copy()
cv2.rectangle(frame,(x1,y1),(x2,y2),(255,0,0),2)
objeto = imAux[y1:y2,x1:x2]
objeto = imutils.resize(objeto, width=38)
# print(objeto.shape)
k = cv2.waitKey(1)
if k == ord('s'):
    cv2.imwrite(Datos+'/object_{ }.jpg'.format(count),objeto)
    print('Image captured: ', 'object_{ }.jpg'.format(count))
    count = count + 1
if k == 27:
    break
cv2.imshow('frame',frame)
cv2.imshow('objeto',objeto)
cap.release()
cv2.destroyAllWindows()

```

CHAPTER 5

TESTING

5.1 SOFTWARE TESTING

Software testing is a critical element of software quality assurance and represents the ultimate review of specification, design, and code generation.

5.1.1 Testing Objectives:

- To ensure that during operation the system will perform as per specification.
- To make sure that system meets the user requirements during operation.
- To make sure that during the operation, incorrect input, processing and output will be detected.
- To see that when correct inputs are fed to the system the outputs are correct.
- To verify that the controls incorporated in the same system as intended.
- Testing is a process of executing a program with the intent of finding an error.
- A good test case is one that has a high probability of finding an as yet undiscovered error.

The software developed has been tested successfully using the following testing strategies and any errors that are encountered are corrected and again the part of the program or the procedure or function is put to testing until all the errors are removed. A successful test is one that uncovers an as yet undiscovered error.

Note that the result of the system testing will prove that the system is working correctly. It will give confidence to system designer, users of the system, prevent frustration during implementation process etc.

5.2 TESTING DESIGN

5.2.1 White box testing:

White box testing is a testing case design method that uses the control structure of the procedure design to derive test cases. All independent paths in a module are exercised at least once, all logical decisions are exercised at once, execute all loops at boundaries and within their operational bounds exercise internal data structure to ensure their validity. Here the customer is given three chances to enter a valid choice out of the given menu. After which the control exits the current menu.

5.2.2 Black Box Testing:

Black Box Testing attempts to find errors in following areas or categories, incorrect or missing functions, interface error, errors in data structures, performance error and initialization and termination error. Here all the input data must match the data type to become a valid entry.

The following are the different tests at various levels:

5.2.3 Unit Testing:

Unit testing is essentially for the verification of the code produced during the coding phase and the goal is to test the internal logic of the module/program. In the Generic code project, the unit testing is done during the coding phase of data entry forms whether the functions are working properly or not. In this phase all the drivers are tested to see if they are rightly connected or not.

5.2.4 Integration Testing:

All the tested modules are combined into sub systems, which are then tested. The goal is to see if the modules are properly integrated, and the emphasis is on the testing interfaces between the modules. In the generic code integration testing is done mainly on table creation module and insertion module.

5.2.5 Validation Testing:

This testing concentrates on confirming that the software is error-free in all respects.

All the specified validations are verified and the software is subjected to

hard-core testing. It also aims at determining the degree of deviation that exists in the software designed from the specification; they are listed out and are corrected.

5.2.6 System Testing:

This testing is a series of different tests whose primary is to fully exercise the computer-based system. This involves:

- Implementing the system in a simulated production environment and testing it.
- Introducing errors and testing for error handling.

5.2.7 Test Cases:

Test Case	Valid	Invalid
Garbage in image	Yes	No
No Garbage in image	No	Yes

5.3 POWER BI

5.3.1 Power BI Model Dashboard

❖ Power BI Model Dashboard

Power BI is a Data Visualization and Business Intelligence tool that converts data from different data sources to interactive dashboards and BI reports. Power BI suite provides multiple software, connector, and services - Power BI desktop, Power BI service based on Saas, and mobile Power BI apps available for different platforms. These set of services are used by business users to consume data and build BI reports.

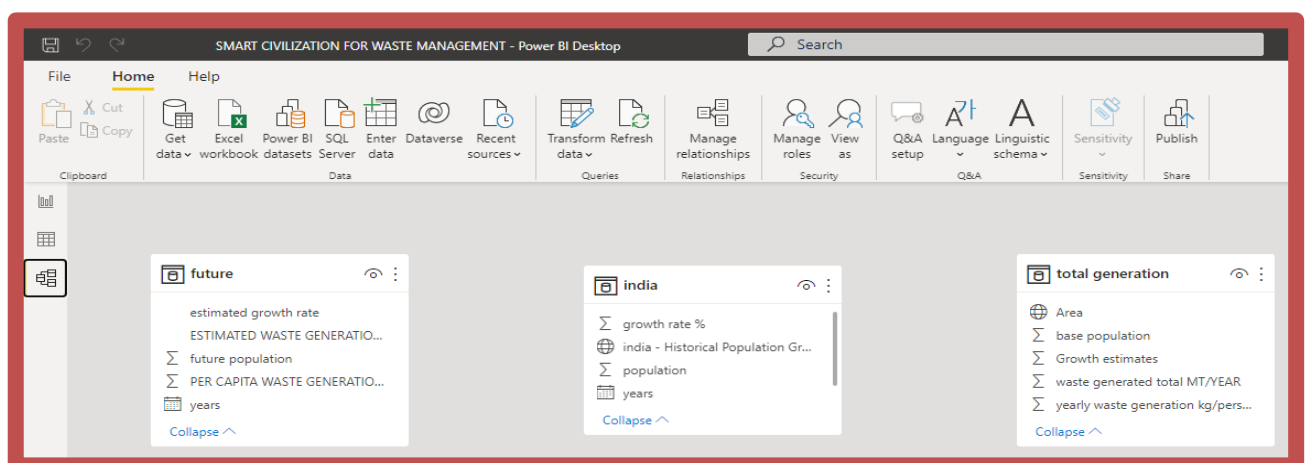


Fig 5 Power BI Model Dashboard

5.3.2 Uses of Power BI:

- A quick start.
- Streamlined publication and distribution.
- Real-time information.
- Ability to customize Power BI app navigation.
- Ability to customize security features.
- Cortana integration.
- Artificial Intelligence.

5.3.3 Power BI Dashboard:

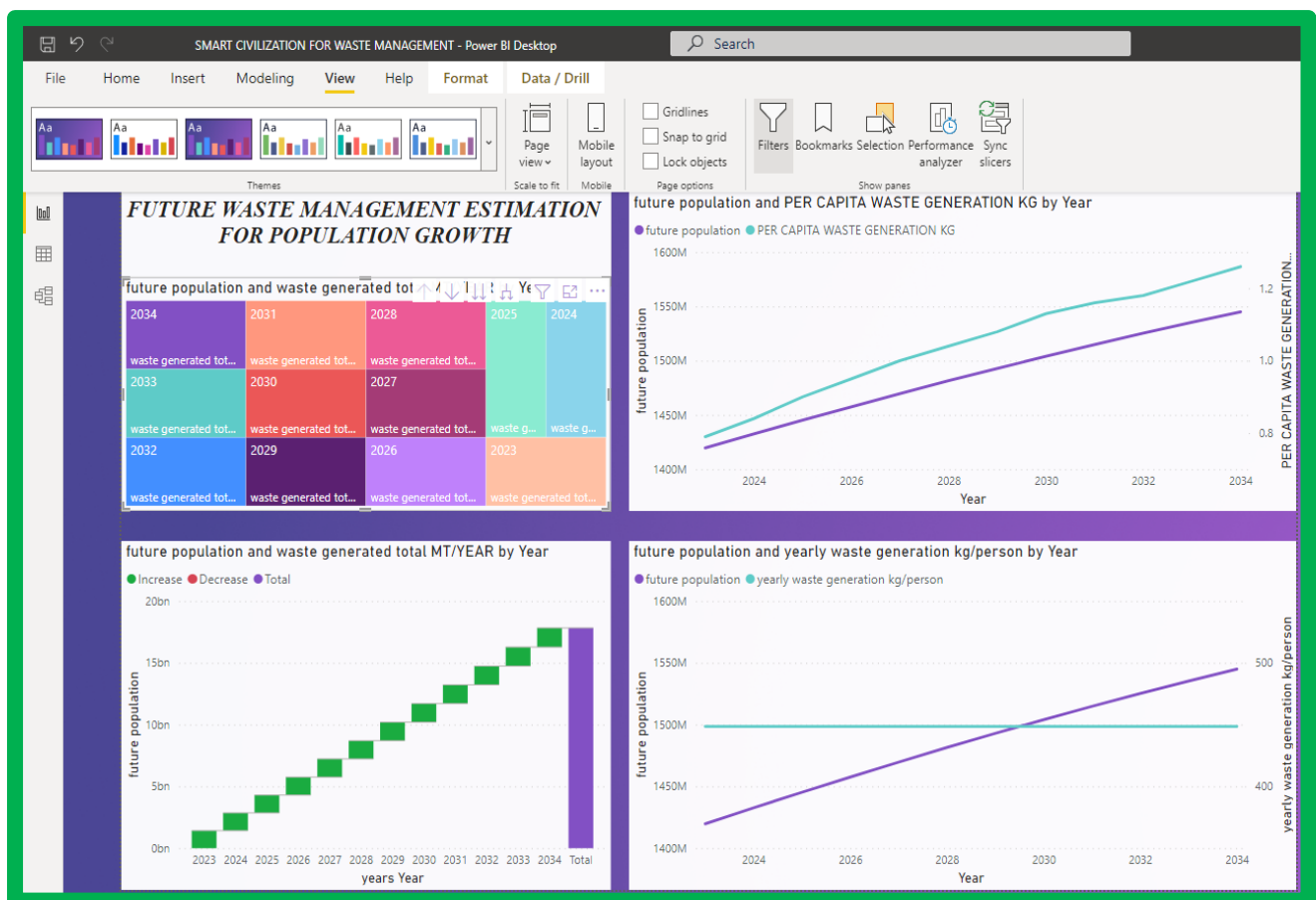


Fig 5.3.3: Power BI Dashboard.

CHAPTER 6

LIST OF IMAGES

6.1 INDIA TABLE.png

The population of India has grown rapidly over the last sixty years, from about 350 million in 1947 to approximately 1.16 billion today. The statistic shows the population growth in India from 2010 to 2020. In 2020, the population of India grew by about 0.99 percent compared to the previous year. We collected the growth rate data is taken from 1998 to 2022. The growth rate percentage is also taken according to the population. The current population of World in 2022 is 7,953,952,567, a 1% increase from 2021. The current population of India is 1,405,754,557 as of Saturday, May 28, 2022, based on World meter website elaboration of the latest United Nations data.

Population growth rate is the percentage change in the size of the population in a year. It is calculated by dividing the number of people added to a population in a year (Natural Increase + Net In-Migration) by the population size at the start of the year.

India - Historical Population Growth Rate Data	years	population	growth rate %
India - Historical Population Growth Rate Data	Friday, April 29, 2022	1406631776	0.95%
India - Historical Population Growth Rate Data	Friday, April 30, 2021	1393409038	0.97%
India - Historical Population Growth Rate Data	Friday, May 1, 2020	1380004385	0.99%
India - Historical Population Growth Rate Data	Thursday, May 2, 2019	1366417754	1.02%
India - Historical Population Growth Rate Data	Thursday, May 3, 2018	1352642280	1.04%
India - Historical Population Growth Rate Data	Thursday, May 4, 2017	1338676785	1.07%
India - Historical Population Growth Rate Data	Thursday, May 5, 2016	1324517249	1.10%
India - Historical Population Growth Rate Data	Wednesday, May 6, 2015	1310152403	1.12%
India - Historical Population Growth Rate Data	Wednesday, May 7, 2014	1295600772	1.15%
India - Historical Population Growth Rate Data	Wednesday, May 8, 2013	1280842125	1.19%
India - Historical Population Growth Rate Data	Wednesday, May 9, 2012	1265780247	1.24%
India - Historical Population Growth Rate Data	Tuesday, May 10, 2011	1250287943	1.30%
India - Historical Population Growth Rate Data	Tuesday, May 11, 2010	1234281170	1.36%
India - Historical Population Growth Rate Data	Tuesday, May 12, 2009	1217726215	1.42%
India - Historical Population Growth Rate Data	Tuesday, May 13, 2008	1200669765	1.48%
India - Historical Population Growth Rate Data	Monday, May 14, 2007	1183209472	1.52%
India - Historical Population Growth Rate Data	Monday, May 15, 2006	1165486291	1.56%
India - Historical Population Growth Rate Data	Monday, May 16, 2005	1147609927	1.59%
India - Historical Population Growth Rate Data	Monday, May 17, 2004	1129623456	1.63%
India - Historical Population Growth Rate Data	Sunday, May 18, 2003	1111523144	1.67%
India - Historical Population Growth Rate Data	Sunday, May 19, 2002	1093317189	1.70%
India - Historical Population Growth Rate Data	Sunday, May 20, 2001	1075000085	1.74%
India - Historical Population Growth Rate Data	Sunday, May 21, 2000	1056575549	1.78%
India - Historical Population Growth Rate Data	Saturday, May 22, 1999	1038058156	1.82%
India - Historical Population Growth Rate Data	Saturday, May 23, 1998	1019483581	1.86%

Fig:6.1 India table

6. b Table data: India population of this year rural and urban population

SMART CIVILIZATION FOR WASTE MANAGEMENT - Power BI Desktop

File Home Help **Table tools**

Name: total generation

Mark as date table | Manage relationships | New measure | Quick measure | New measure column | New table

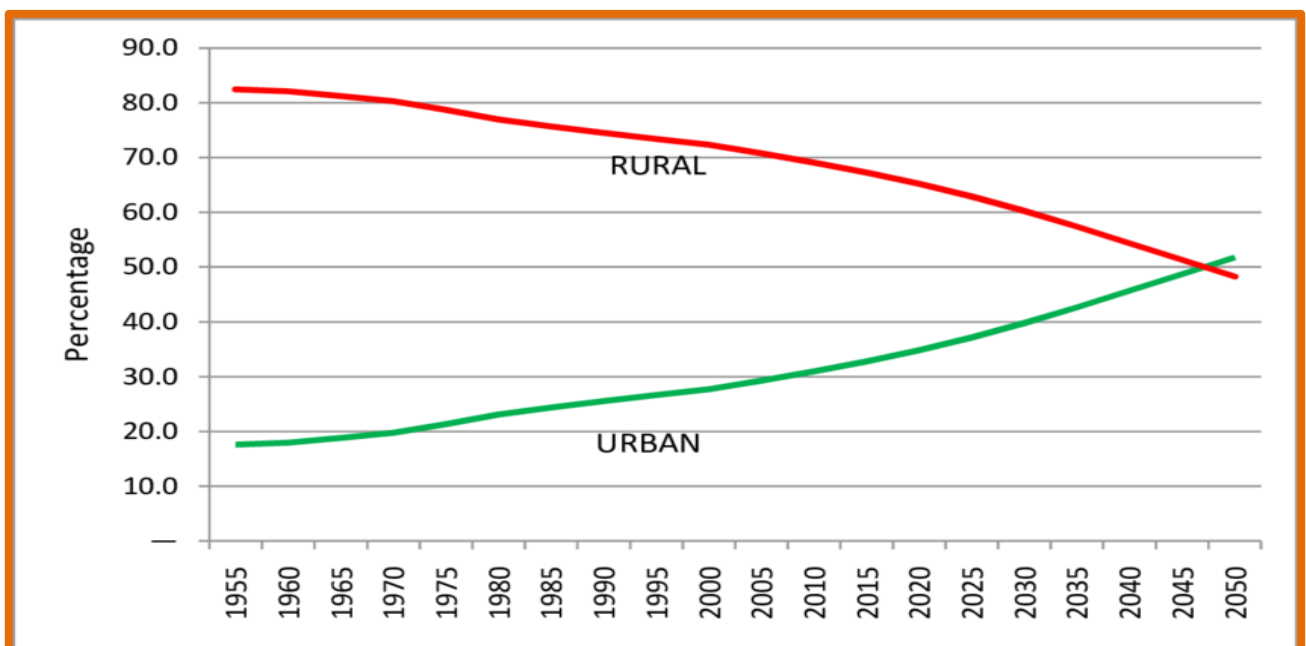
Structure | Calendars | Relationships | Calculations

Area	base population	Growth estimates	yearly waste generation kg/person	waste generated total MT/YEAR
Rural and Urban	1419713452	93.00%	281.24	399280211105
Rural and Urban	1432632844	91.00%	299.04	428414525648
Rural and Urban	1445240013	88.00%	320.4	463054900150
Rural and Urban	1457524553	85.00%	338.2	492934803846
Rural and Urban	1469622007	83.00%	356	523185434440
Rural and Urban	1481525945	81.00%	370.24	548520165917
Rural and Urban	1492933695	77.00%	384.48	574003147010
Rural and Urban	1503981404	74.00%	402.28	605021639293
Rural and Urban	1514810070	72.00%	412.96	625555966647
Rural and Urban	1525262260	69.00%	420.08	640732170107
Rural and Urban	1535328991	66.00%	434.32	666824087257
Rural and Urban	1545001563	63.00%	448.56	693025901270

Fig 6.b: Table Data

6.b.1 Description:

Rural and Urban Population is combined to find the growth rate in a country. Rural Population is decreasing rapidly and increasing Urban Population.



6.b India Population Growth from 1998 to 2028:

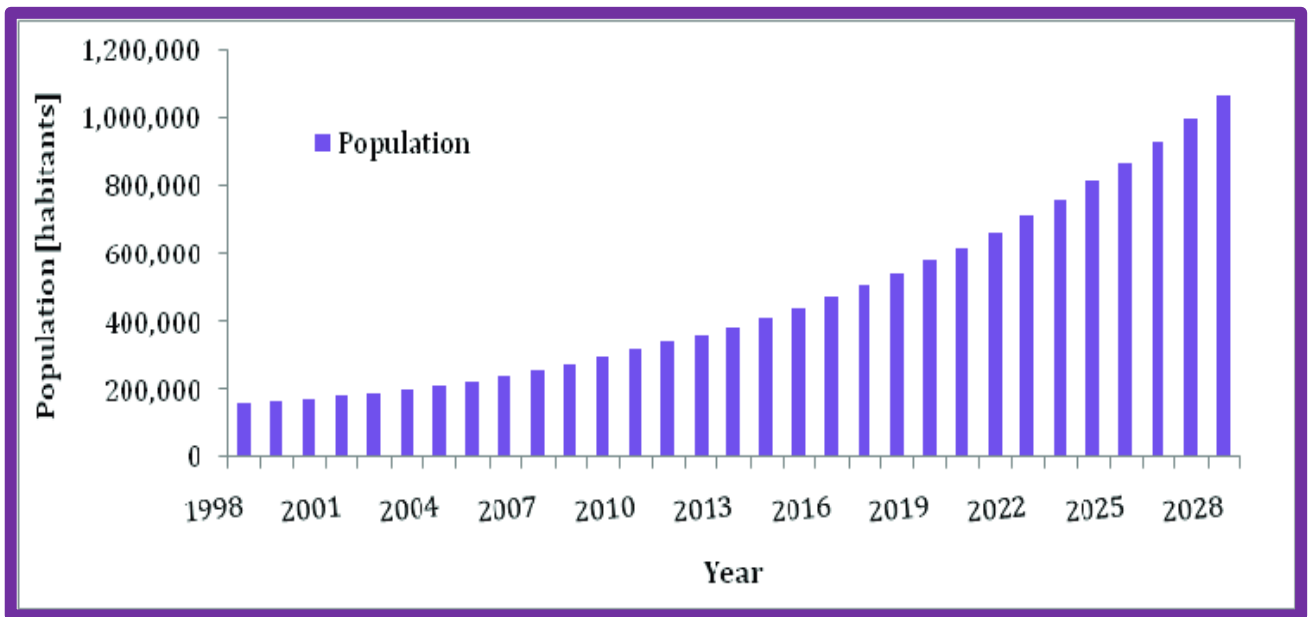


Fig 6.c: Population Growth.

6.2 MS EXCEL FILE IN SMART CIVILIZATION:

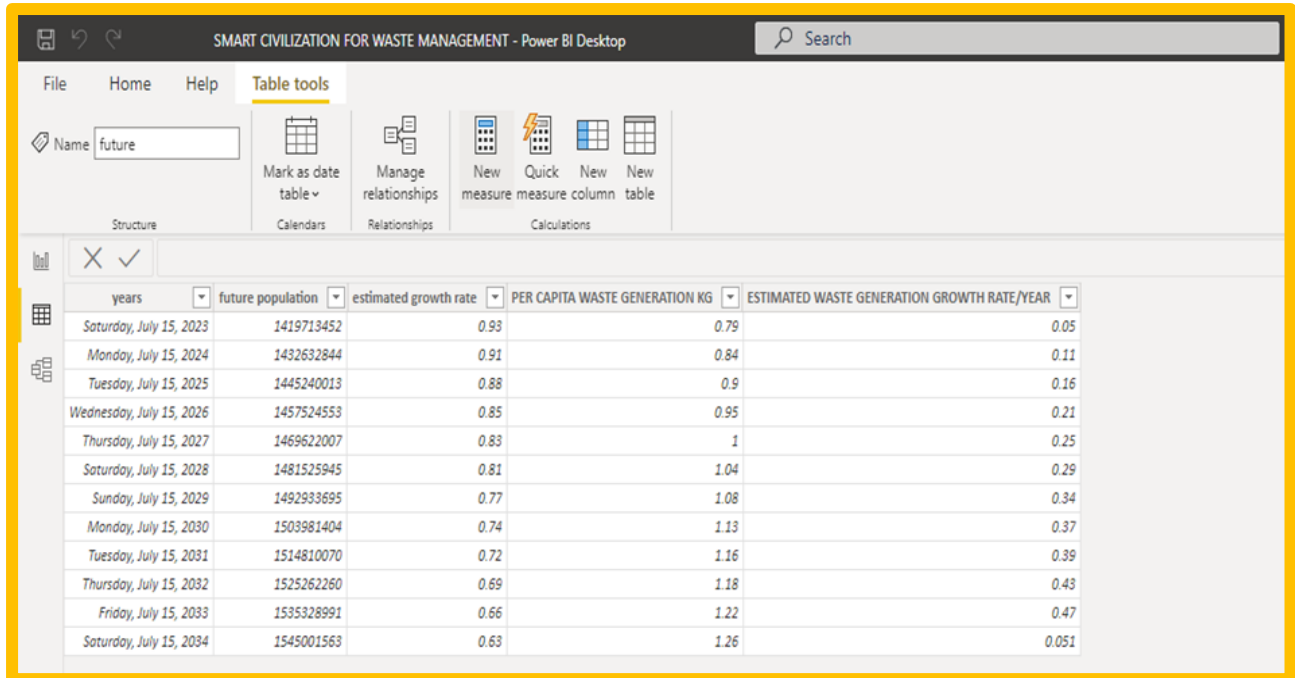
MS Excel tutorial provides basic and advanced concepts of Excel. Our Excel tutorial is designed for beginners and professionals by keeping their requirements in mind.

Microsoft Excel is a computer application program written by Microsoft. It mainly comprises tabs, groups of commands, and worksheets. It stores the data in tabular form and allows the users to perform manipulation operations on them.

6.3 SMART CIVILIZATION MS EXCEL FILE:

1						Area
2						Rural and Urban
3	years	future population	estimated growth r	PER CAPITA WAST	ESTIMATED WASTE GENERATION GROWTH RATE/YEAR	Rural and Urban
4	15-07-2023	1419713452	93.00%	0.79	0.05	Rural and Urban
5	15-07-2024	1432632844	91.00%	0.84	0.11	Rural and Urban
6	15-07-2025	1445240013	88.00%	0.9	0.16	Rural and Urban
7	15-07-2026	1457524553	85.00%	0.95	0.21	Rural and Urban
8	15-07-2027	1469622007	83.00%	1	0.25	Rural and Urban
9	15-07-2028	1481525945	81.00%	1.04	0.29	Rural and Urban
10	15-07-2029	1492933695	77.00%	1.08	0.34	Rural and Urban
11	15-07-2030	1503981404	74.00%	1.13	0.37	Rural and Urban
12	15-07-2031	1514810070	72.00%	1.16	0.39	Rural and Urban
13	15-07-2032	1525262260	69.00%	1.18	0.43	Rural and Urban
14	15-07-2033	1535328991	66.00%	1.22	0.47	
15	15-07-2034	1545001563	63.00%	1.26	0.051	MSW generation
16	Population growth rate = ((Natural Increase + Net in Migration)/Starting population)) * 100					in small towns t
17						
18						
19	Population Growth and Population Decrease : Formula					
20						
21	The population decrease and growth formula are as below:					0.17kg+0.62kg=0.79kg
22	1. If a constant rate of growth be R% per annum, then population after n years = $P \times (1+R/100)^n$.					
23	2. if the growth be R% during first year and Q% during second year the population after 2 years = $P \times (1+R/100) \times (1+Q/100)$					
24	3. if the constant decrease in population be R% per annum, then the population after n years = $P \times (1-R/100)^n$.					
25						
26						
27						

6.3 FUTURE DATA: Table having data regarding the future population which is represented in dash board

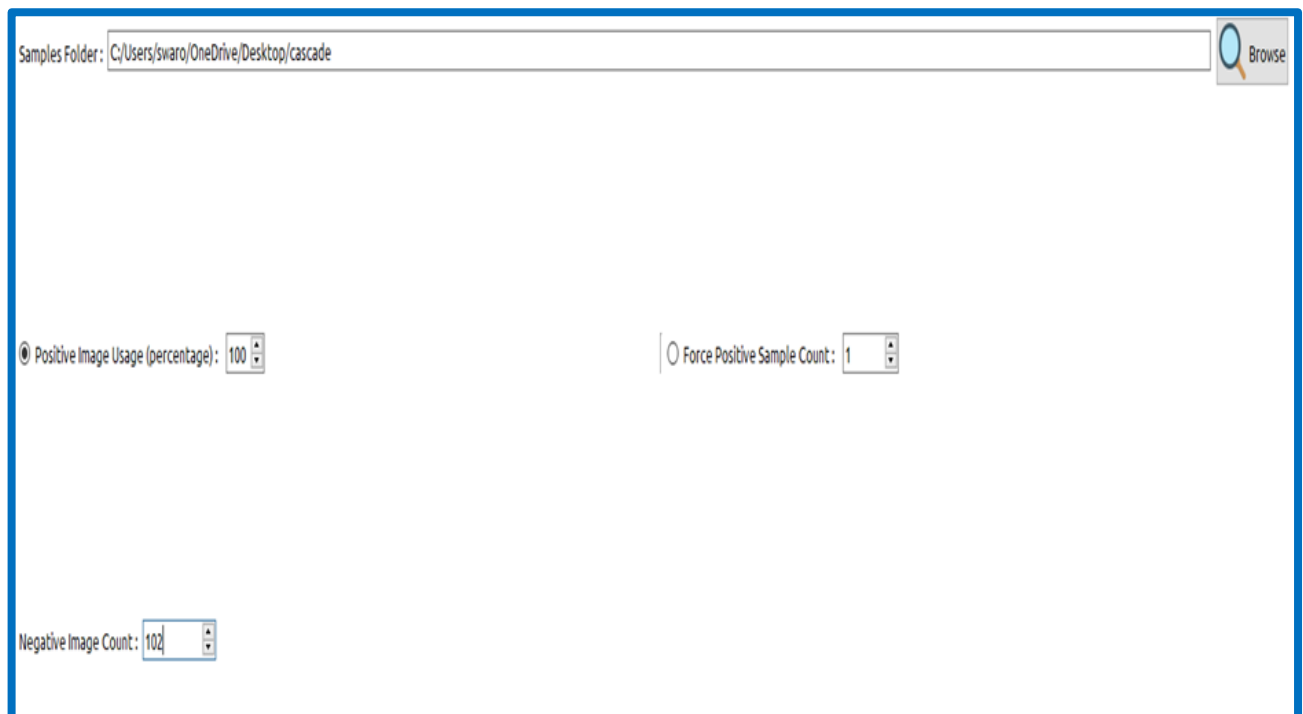


years	future population	estimated growth rate	PER CAPITA WASTE GENERATION KG	ESTIMATED WASTE GENERATION GROWTH RATE/YEAR
Saturday, July 15, 2023	1419713452	0.93	0.79	0.05
Monday, July 15, 2024	1432632844	0.91	0.84	0.11
Tuesday, July 15, 2025	1445240013	0.88	0.9	0.16
Wednesday, July 15, 2026	1457524553	0.85	0.95	0.21
Thursday, July 15, 2027	1469622007	0.83	1	0.25
Saturday, July 15, 2028	1481525945	0.81	1.04	0.29
Sunday, July 15, 2029	1492933695	0.77	1.08	0.34
Monday, July 15, 2030	1503981404	0.74	1.13	0.37
Tuesday, July 15, 2031	1514810070	0.72	1.16	0.39
Thursday, July 15, 2032	1525262260	0.69	1.18	0.43
Friday, July 15, 2033	1535328991	0.66	1.22	0.47
Saturday, July 15, 2034	1545001563	0.63	1.26	0.051

Fig 6.d: Future Data Image.

6.4 TRAINING THE IMAGES:

6.4.1 Input: Our cascade is trained by giving the file containing negative and positive samples



Samples Folder: C:/Users/swaro/OneDrive/Desktop/cascade

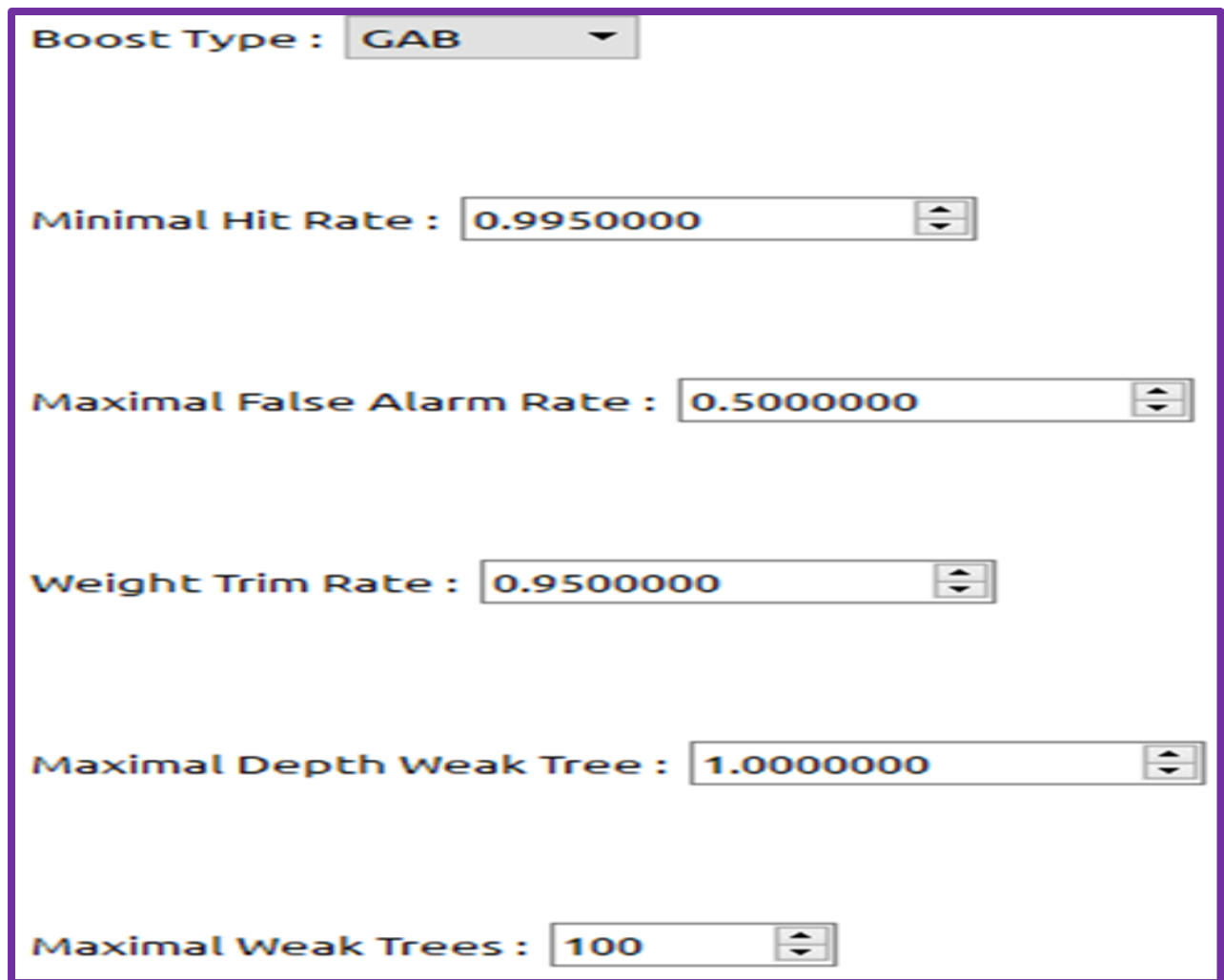
Positive Image Usage (percentage): 100

Force Positive Sample Count: 1

Negative Image Count: 102

Fig 6.4.1 Input.png

6.4.2 Boost: Application defaults for training the positive and negative samples



The image shows a configuration window for the Boost classifier. It contains several parameters, each with a text label, a numeric input field, and a small up/down arrow button. The parameters and their values are:

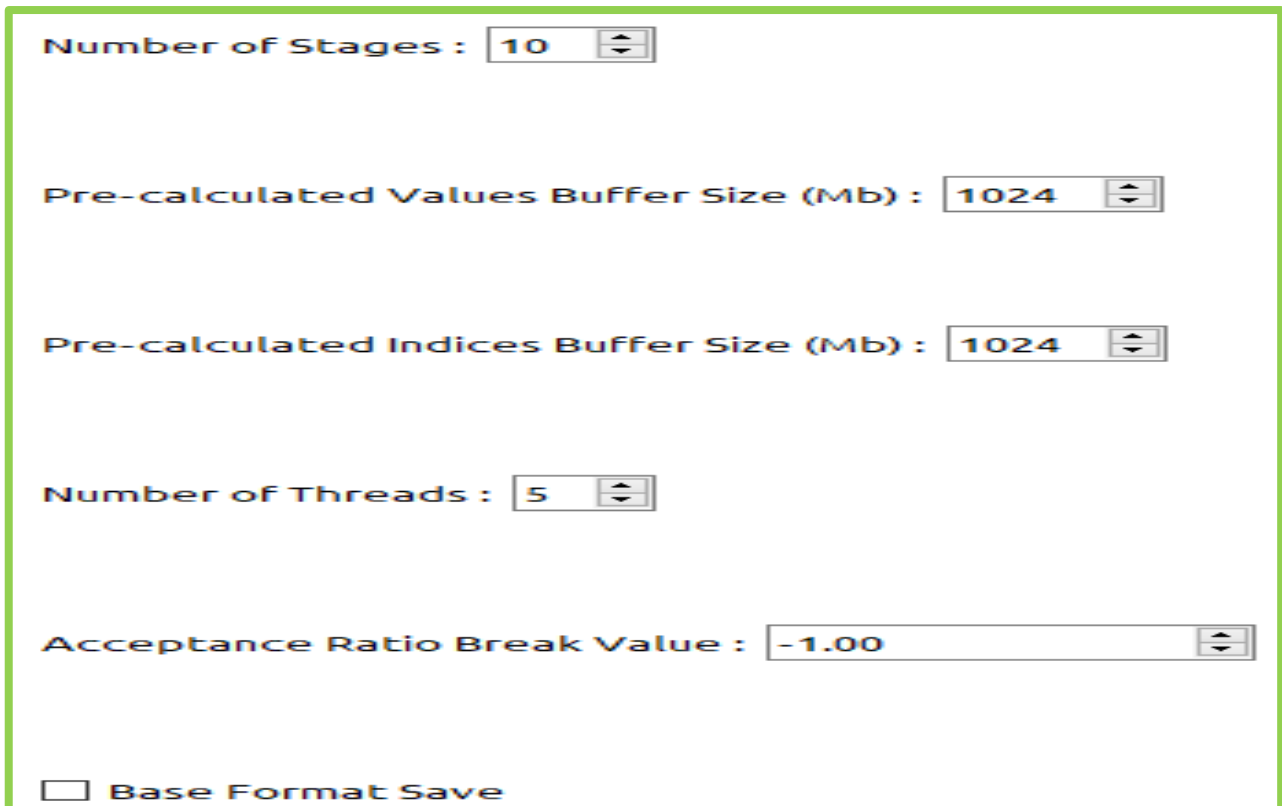
- Boost Type : GAB (selected from a dropdown)
- Minimal Hit Rate : 0.9950000
- Maximal False Alarm Rate : 0.5000000
- Weight Trim Rate : 0.9500000
- Maximal Depth Weak Tree : 1.0000000
- Maximal Weak Trees : 100

Fig 6.4.2 Boost.png

Common, Cascade and Boost tabs can be used for setting numerous parameters for customizing the classifier training. Cascade Trainer GUI sets the most optimized and recommended settings for these parameters by default, still some parameters need to be modified for each training. Note that detailed description of all these parameters are beyond the scope of this help page and require deep knowledge about cascade classification techniques.

You can set pre-calculation buffer size to help with the speed of training process. You can assign as much memory as you can for these but be careful to not assign too much or too low. For example if you have 8 GB of RAM on your computer then you can safely set both of the buffer sizes below to 2048.

6.4.3 Common: stages are specified for training where the accuracy is controlled



Number of Stages : 10

Pre-calculated Values Buffer Size (Mb) : 1024

Pre-calculated Indices Buffer Size (Mb) : 1024

Number of Threads : 5

Acceptance Ratio Break Value : -1.00

☐ Base Format Save

Fig 6.4.3 Common.png

6.4.4 Cascade Trainer: log of cascade training after giving negative and positive samples

```
*****  
***** TRAINING CLASSIFIER *****  
*****  
Running : opencv_traincascade  
PARAMETERS:  
cascadeDirName: C:\Users\swaro\OneDrive\Desktop\cascade\classifier  
vecFileName: C:\Users\swaro\OneDrive\Desktop\cascade\pos_samples.vec  
bgFileName: C:\Users\swaro\OneDrive\Desktop\cascade\neg.lst  
numPos: 51  
numNeg: 102  
numStages: 10  
precalcValBufSize[Mb] : 1024  
precalcIdxBufSize[Mb] : 1024  
  
acceptanceRatioBreakValue : -1  
stageType: BOOST  
featureType: HAAR  
sampleWidth: 46  
sampleHeight: 38  
boostType: GAB  
minHitRate: 0.995  
maxFalseAlarmRate: 0.5  
weightTrimRate: 0.95  
maxDepth: 1  
maxWeakCount: 100  
mode: BASIC  
Number of unique features given windowSize [46,38] : 1481798
```

6.4.5 Litter / Garbage Detection Output (No Garbage):

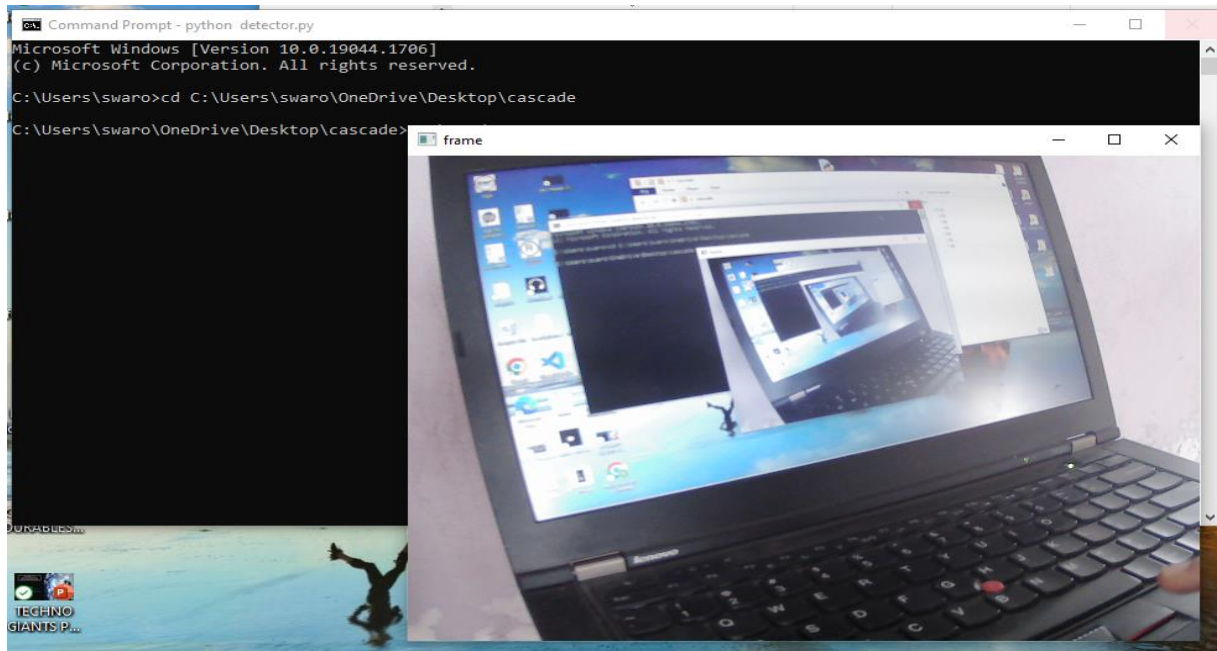


Fig 6.4.5 No Garbage Detection

Description of 6.4.5: web cam detects nothing when there is laptop in its sight it only detects garbage like 6.5 output.

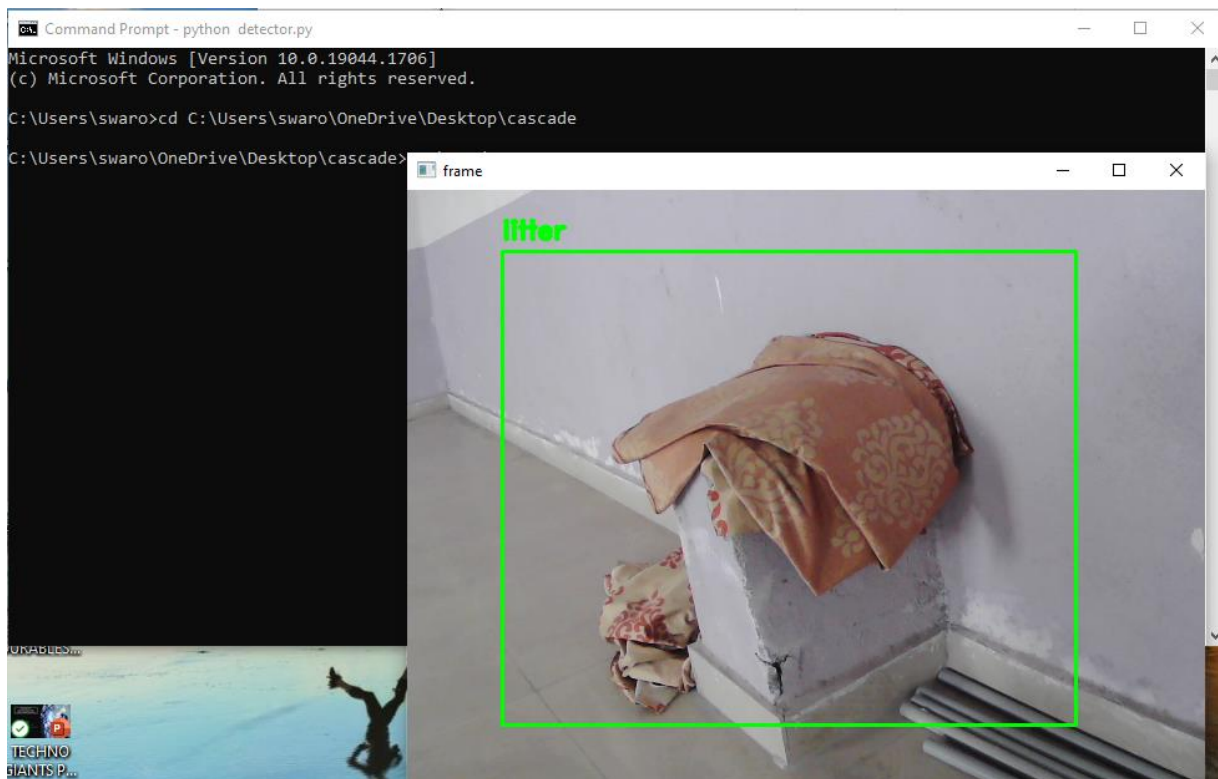
How does a human mind apprehend an image? When you see the image below, what do you actually see and how do you say what is in the Image?

You most probably look for different shapes and colours in the Image and that might help you decide that this is an image of a dog. But does a computer also see it in the same way? The answer is no.

A digital image is an image composed of picture elements, also known as pixels, each with finite, discrete quantities of numeric representation for its intensity or grey level. So, the computer sees an image as numerical values of these pixels and in order to recognise a certain image, it has to recognise the patterns and regularities in this numerical data.

Here is a hypothetical example of how pixels form an image. The darker pixels are represented by a number closer to the zero and lighter pixels are represented by numbers approaching one. All other colours are represented by the numbers between 0 and 1.

6.5 Output of Litter Detection code:



6. 5 Output of Litter Detection code

Description of 6.5

The web cam is now capable of detecting the garbage which is there at its sight we are using our detecting code from python.

6.5.1 Garbage Detection and Power BI Dashboard:

In this we will link both Dashboard and Detection code and ensure that both will work simultaneously. We create a URL and connect URL with Dashboard to get the increase of wastage in future and the percentage of wastage present in a particular area.

6.6 Litter Detection.png

- Identify Litter on Roads easily.
- Reduce the wastage on roads

CHAPTER 7

ADVANCED USE CASES OF OPEN CV

OpenCV is the leading open-source library for computer vision, image processing and machine learning, and now features GPU acceleration for real-time operation.

OpenCV is released under a BSD license and hence it's free for both academic and commercial use. It has C++, C, Python and Java interfaces and supports Windows, Linux, Mac OS, iOS and Android. OpenCV was designed for computational efficiency and with a strong focus on real-time applications. Written in optimized C/C++, the library can take advantage of multi-core processing. Adopted all around the world, OpenCV has more than 47 thousand people of user community and estimated number of downloads exceeding 6 million. Usage ranges from interactive art, to mines inspection, stitching maps on the web or through advanced robotics.

OpenCV Applications

OpenCV is being used for a very wide range of applications which include:

- Street view image stitching
- Automated inspection and surveillance
- Robot and driver-less car navigation and control
- Medical image analysis
- Video/image search and retrieval
- Movies - 3D structure from motion
- Interactive art installations

OpenCV Functionality

- Image/video I/O, processing, display
- Object/feature detection (obj detect, features2d, non-free)
- Geometry-based monocular or stereo computer vision (calib3d, stitching, video stab)
- Computational photography (photo, video, superres)
- Machine learning & clustering (ml, Flann)
- CUDA acceleration (gpu)

Application of Computer Vision

- **Automobiles:**

Companies like [Tesla](#) and **Waymo** invested in **self-driving cars**, that will be assumed to be the future of the automobile industry.

- **Retail Stores:**

Amazon Go, which overcame the work of store owners by eliminating the need for a cashier and checkout counters, as a customer can directly pay their bills utilizing, which proves to save the time of the customer and disappear the endless queues of people waiting at the checkout counters for their turn.

- **Face Recognition:**

Using [machine learning](#) and computer vision, it's now possible to recognize the faces of a particular person either from videos or photos. In particular, this application can also be utilized in **surveillance**.

7.1 Open cv in Tesla cars

In recent years we keep witnessing a major advance in AI, which brings about an ascending implementation of computer vision (CV) in real life. Putting all the current hype around autonomous vehicles aside, the autopilot system is indeed one of the major accomplishments in ML that projects the future reality. It is expected that self-driving cars will become commonplace in the span of 10 years. Chris Gerdes, a professor of Mechanical Engineering at Stanford University and co-director of the Center for Automotive Research at Stanford, is confident in his [statement](#) that “we can soon give cars the skills of the very best human drivers, and maybe even better than that.” Yet, with the growing demand, there are also emerging challenges for CV in autonomous vehicles. In this blog, we will cover the following:

- [Gathering the training data](#)
- [Data labeling](#)
- [Object detection for autonomous vehicles](#)
- [Semantic segmentation and instance segmentation](#)
- [Multi-camera vision and depth estimation](#)
- [Key takeaways](#)

- **Gathering the training data**

Cars without a human operator require rigorous pattern recognition and a ton of computing power to drive independently. One of the main challenges for AI-powered self-driving cars is the acquisition of training datasets. An AI solution is as good as the data it is trained on. Given that, quality datasets and pixel-perfect labeling are of incremental value for the model.

One of the better options of data collection to be used for computer vision in autonomous vehicles is driving around and capturing shots, which can be done either through semi-autonomous driving or by using an artificial model such as the computer game engine. The model has to undergo multiple iterations of camera-generated images for sufficient detection. Keep in mind that the training process will mostly require object images to be recognized by your CV model: things that may appear on the road, street signs, road lanes, humans, buildings, other cars, etc.

Each of these elements is labeled through an individual [annotation](#) type: polyline for lane detection, 3D point annotation for LiDARs, and so forth. A similar variety points at the complexity and vast amounts of data needed to train a model.

- **Data labeling**

[Data labeling](#) requires heavy manual labor. For datasets as massive as self-driving cars, data labeling is especially dependent on human effort to identify unlabeled elements in raw images. In the meantime, the labeled data has to be accurate to run successful ML projects. Maintaining high levels of precision for large-scale projects is especially challenging. With the increased workforce, there comes increased responsibility for keeping communication open and setting up an effective feedback system so that annotation teams or members within the teams operate in cohesion. For that, we recommend setting up an annotation guideline that roadmaps the annotation process and provides concise instruction to avoid further mistakes and imbalance.



Fig: Object Detection

Cohesion, however, should not be confused with the data type. A CV model has to be able to make accurate predictions and estimations based on what it “sees” on the road and beyond, which comes down to the need for diverse data input when training a model.

There are multiple ways you could go with data labeling, including in-house, through [outsourcing](#), or crowdsourcing. Whichever you end up choosing, make sure to set up a robust management process to [develop a scalable annotation pipeline](#).

- **Object detection for autonomous vehicles**

Self-driving cars use CV to detect objects. Object detection takes two steps: [image classification](#) and image localization.

Image classification is done by training the [convolutional neural network](#) (CNN) to recognize and classify objects. The problem with CNN is that it’s not the best solution for images with multiple objects, as the model is likely not to capture all objects. This is where sliding windows come into play.

As the window slides over the image, it runs each part of the CNN and checks if it resembles any object the model is trained to recognize. If there are objects considerably larger or smaller than the window size, the model won’t detect them. To get that covered, you can use different window sizes for sliding purposes or apply the You Look Only Once ([YOLO](#)) algorithm. In this case, the image is run through the CNN only once, as you split it into grids. In the end, YOLO provides predictions based on the probability of each grid cell containing an object: so, no need for several run-throughs.

Now, to point out where the object is positioned on an image, we use the so-called non-max suppression (NMS) algorithm. The NMS algorithm selects the best bounding box for an object based on the highest objectiveness score and the overlap or intersection over union (IoU: calculated by dividing the area of overlap by the area of union) of the bounding boxes while omitting the rest. The objectiveness score provides the probability of an object being present in the bounding box. The selection process is repeated and reiterated until there is no room for box reduction. In short, NMS can be described as taking the boxes with the lowest probability score and suppressing them.

Let's use the example below to illustrate how NMS works. Suppose you want your model to detect the car vs. the truck on an image. Here is how you should proceed with box selection using NMS.



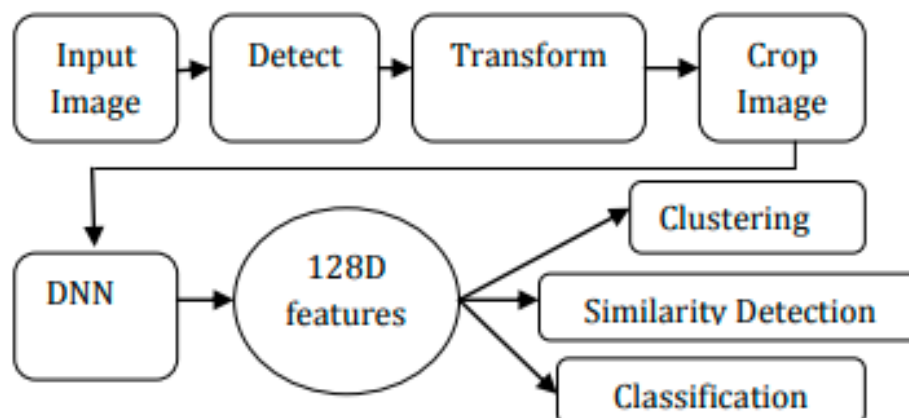
Fig: detecting objects on roads

- Pick the box with the highest objectiveness score
- Compare the IoU of the selected box with other boxes
- Suppress the bounding box with the IoU over 50%
- Move the next highest objectiveness score
- Repeat steps 2-4 all over

The end result, in our case, will be the green boxes with the highest objectiveness score, which is what you want your CV model to identify.

7.2 Real Time Face Recognition System Using OpenCV

This real time GUI based face recognition system is developed using Open source tool Open face. Open Face is the face recognition tool developed by Carnegie Mellon University, using OpenCV. Open Face, consists in a broader Prospective, three phases: Detection, Feature extraction, and Recognition. The dimensionality of face image is reduced by the Histogram of Oriented Gradients (HOG) and this algorithm is developed to detect frontal views of faces. After detecting the face part of image, extract the 128 face features for the given image by using a Deep Neural Network algorithm and the recognition is done by the Support Vector machine (SVM) classifier. HOG is one of the most popular representation methods for a face image. It not only reduces the dimensionality of the image, but also extracting the facial features of the given images, and retains some of the variations in the image data. So dimensionality of face image reduced by HOG using deep learning algorithm and recognition is done by SVM approach Face Recognition is used to recognize a person by using some features of that particular person's face, by matching with stored models of each individual face in a group of people. Face recognition is the natural way of identification and authenticating a person. Face Recognition plays an important role in human's day-to-day communication and daily lives. Security and authentication of a person is crucial in any industry or establishment. So, in today's environment, there is a great deal of interest in automatic face recognition using computers or devices for verification of identities round the clock and sometimes remotely. Face recognition has become one of the most challenging and interesting problems in the pattern recognition and image processing.



Training classifier

Training a classifier Open face is core provides a feature extraction method to obtain a low-dimensional representations of any face. Demos/classifier.py shows a demo of how these representations can be use to create a face classifier. There is a distinction between training the DNN model for feature representation and training a model for classifying people with the DNN model.

Create a classification model:

1. Create raw image directory Create a directory for your raw images so that images from different people are in different subdirectories. The names of the labels or images do not matter, and each person can have a different number of images. The images should be formatted as jpg or png.
2. Preprocess the raw images Here for the given input image, it will detect the face part of the given image and apply the affine transform for the given image. Then extract the features of given image, finally it will align (crop) the face part of that particular image.
3. Generate Representations In this we are generating the 128 facial features of each and every input image. And it will create the labels and representations of all images.
4. Classification Model After generating the face features, the system will classify how many people present in the data base. Suppose in my system I am trained 20 persons each of 20 images, then the system will be recognized as 20 classifiers are present in the classification model. After Generating classifier, The Test images are compared with the database. If the given image is in the training, then it will recognize the particular person's name with confidence. Otherwise, it will recognize as unable to find a face image.



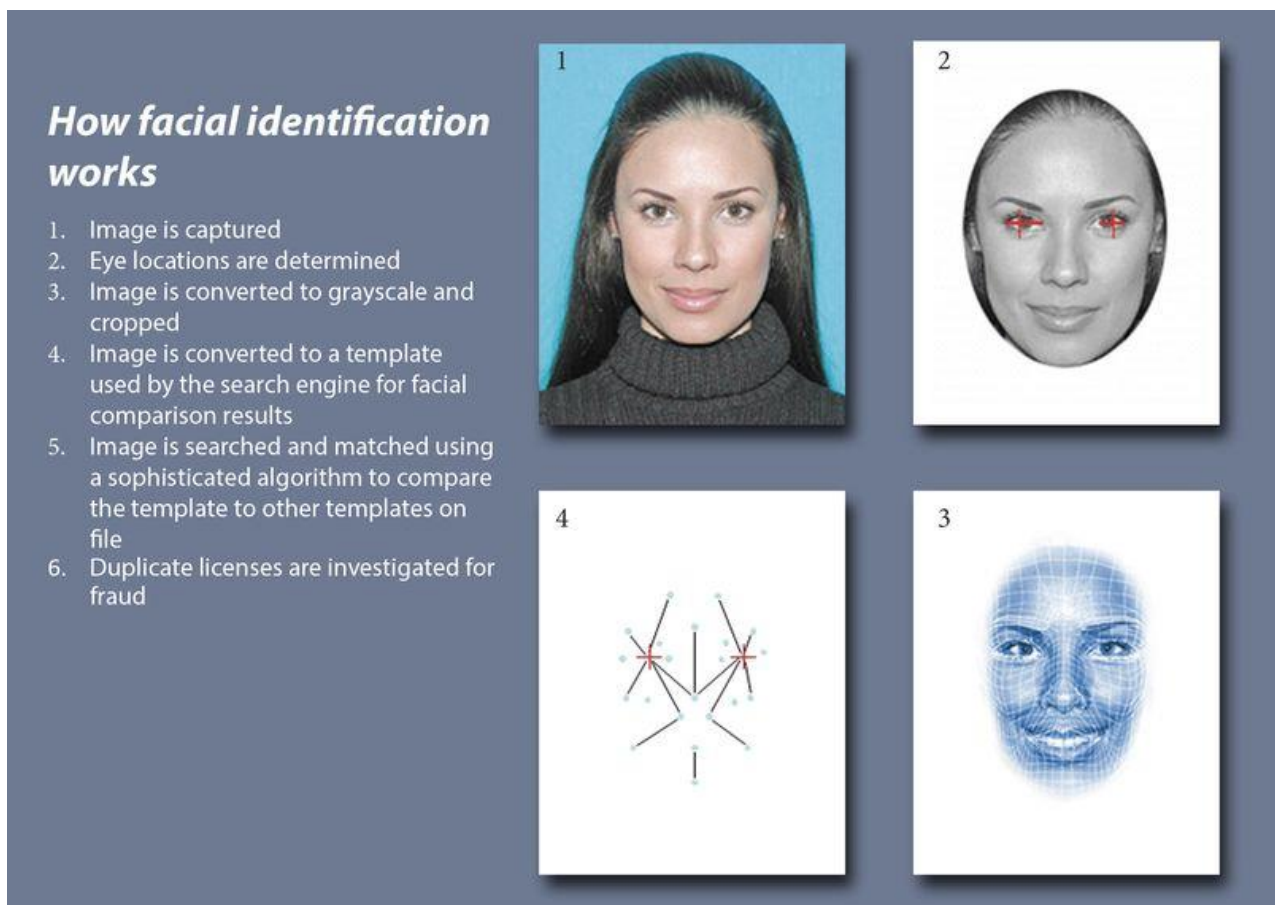
Fig: face recognition

The potential for building Face Recognition systems using Open Face is shown by the examples given at CMU website only. Now how to build a face recognition system, suitable for our institution or for any other application, using open Face will be explained in this chapter with the example of building a face recognition system built for our unit (NERTU, OU). Once open face is installed and ready to run, the following steps are required to build a face recognition system.

1. Building database by registering the faces of the people to recognized.
 2. Detection of Faces from the images of registered people and extraction of facial features from the detected faces.
 3. Recognition face in the test image with SVM classifier.
- Once the open face is installed the required modules or software's will be readily available to complete these three tasks. To check whether the system is working properly or not, initially a face recognition system for NERTU having 15 people (staff and students) is built and the same procedure is explained below. Then after that we have built a real time system with GUI. Twenty images having the face of each person, for 15 people are collected using CHEESE software, which is an open source tool developed by Daniel G. Siegel So, this database contains 300 face images. It consists of each person with 20 different poses like smiling, looking serious, head bend, eyes moving and closing eyes with very small variations in the database. Totally 300 images are used for training and 60 images are used for testing. Here I will show the result of 15 persons with 20 images. Here I am not taking any real time pictures. So, the accuracies will be different. To improve the accuracy, increase the number of training images of each person.

7.3 How faces are detected using open cv

Face recognition systems use computer algorithms to pick out specific, distinctive details about a person's face. These details, such as distance between the eyes or shape of the chin, are then converted into a mathematical representation and compared to data on other faces collected in a face recognition database. The data about a particular face is often called a face template and is distinct from a photograph because it's designed to only include certain details that can be used to distinguish one face from another.



Some face recognition systems, instead of positively identifying an unknown person, are designed to calculate a probability match score between the unknown person and specific face templates stored in the database. These systems will offer up several potential matches, ranked in order of likelihood of correct identification, instead of just returning a single result.

Face recognition systems vary in their ability to identify people under challenging conditions such as poor lighting, low quality image resolution, and suboptimal angle of view (such as in a photograph taken from above looking down on an unknown person).

When it comes to errors, there are two key concepts to understand:

A “false negative” is when the face recognition system fails to match a person’s face to an image that is, in fact, contained in a database. In other words, the system will erroneously return zero results in response to a query. A “false positive” is when the face recognition system does match a person’s face to an image in a database, but that match is actually incorrect. This is when a police officer submits an image of “Joe,” but the system erroneously tells the officer that the photo is of “Jack.”

When researching a face recognition system, it is important to look closely at the “false positive” rate and the “false negative” rate, since there is almost always a trade-off. For example, if you are using face recognition to unlock your phone, it is better if the system fails to identify you a few times (false negative) than it is for the system to misidentify other people as you and lets those people unlock your phone (false positive). If the result of a misidentification is that an innocent person goes to jail (like a misidentification in a mugshot database), then the system should be designed to have as few false positives as possible.

- **How Law Enforcement Uses Face Recognition**



Source: Arizona Department of Transportation

Law enforcement agencies are using face recognition more and more frequently in routine policing. Police collect mugshots from arrestees and compare them against local, state, and federal face recognition databases. Once an arrestee's photo has been taken, the mugshot will live on in one or more databases to be scanned every time the police do another criminal search.

Law enforcement can then query these vast mugshot databases to identify people in photos taken from social media, CCTV, traffic cameras, or even photographs they've taken themselves in the field. Faces may also be compared in real-time against "hot lists" of people suspected of illegal activity.

Mobile face recognition allows officers to use smartphones, tablets or other portable devices to take a photo of a driver or pedestrian in the field and immediately compare that photo against one or more face recognition databases to attempt an identification.

Face recognition has been used in airports, at border crossings, and during events such as the Olympic Games. Face recognition may also be used in private spaces like stores and sports stadiums, but different rules may apply to private sector face recognition.

Supporting these uses of face recognition are scores of databases at the local, state and federal level. Estimates indicate that 25% or more of all state and local law enforcement agencies in the U.S. can run face recognition searches on their own databases or those of another agency.

According to *Governing* magazine, as of 2015, at least 39 states used face recognition software with their Department of Motor Vehicles (DMV) databases to detect fraud. The *Washington Post* reported in 2013 that 26 of these states allow law enforcement to search or request searches of driver license databases, however it is likely this number has increased over time.

Databases are also found at the local level, and these databases can be very large. For example, the Pinellas County Sheriff's Office in Florida may have one of the largest local face analysis databases. According to research from Georgetown University, the database is searched about 8,000 times a month by more than 240 agencies.

The federal government has several face recognition systems, but the database most relevant for law enforcement is FBI's Next Generation Identification database which contains more than 30-million face recognition records. FBI allows state and local agencies "lights out" access to this database, which means no human at the federal level checks up on the individual searches. In turn, states allow FBI access to their own criminal face recognition databases.

FBI also has a team of employees dedicated just to face recognition searches called Facial Analysis, Comparison and Evaluation ("FACE") Services. The FBI can access over 400-million non-criminal photos from state DMVs and the State Department, and 16 U.S. states allow FACE access to driver's license and ID photos. Given the large number of DMV databases using face recognition and the number of Americans whose photos are in the State Department's database of passport and U.S. visa holders, Georgetown University has estimated close to half of all American adults have been entered into at least one if not more face recognition databases.

- **Who Sells Face Recognition**

MorphoTrust, a subsidiary of Idemia (formerly known as OT-Morpho or Safran), is one of the largest vendors of face recognition and other biometric identification technology in the United States. It has designed systems for state DMVs, federal and state law enforcement agencies, border control and airports (including TSA PreCheck), and the state department. Other common vendors include 3M, Cognitec, DataWorks Plus, Dynamic Imaging Systems, FaceFirst, and NEC Global.

- **Threats Posed by Face Recognition**

Face recognition data is easy for law enforcement to collect and hard for members of the public to avoid. Faces are in public all of the time, but unlike passwords, people can't easily change their faces. We are seeing increased information-sharing among agencies. Cameras are getting more powerful and technology is rapidly improving. Face recognition data is often derived from mugshot images, which are taken upon arrest, before a judge ever has a chance to determine guilt or innocence. Mugshot photos are often never removed from the database, even if the arrestee has never had charges brought against them. In spite of face recognition's ubiquity and the improvement in technology, face recognition data is prone to error. In fact, the FBI admitted in its privacy impact assessment that its system "may not be sufficiently reliable to accurately locate other photos of the same identity, resulting in an increased percentage of misidentifications." Although the FBI purports its system can find the true candidate in the top 50 profiles 85% of the time, that's only the case when the true candidate exists in the gallery.

If the candidate is not in the gallery, it is quite possible the system will still produce one or more potential matches, creating false positive results. These people—who aren't the candidate—could then become suspects for crimes they didn't commit. An inaccurate system like this shifts the traditional burden of proof away from the government and forces people to try to prove their innocence.

Face recognition gets worse as the number of people in the database increases. This is because so many people in the world look alike. As the likelihood of similar faces increases, matching accuracy decreases. Face recognition software is especially bad at recognizing African Americans. A 2012 study [.pdf] co-authored by the FBI showed that accuracy rates for African Americans were lower than for other demographics. Face recognition software also misidentifies other ethnic minorities, young people, and women at higher rates. Criminal databases include a disproportionate number of African Americans, Latinos, and immigrants, due in part to racially biased police practices. Therefore the use of face recognition technology has a disparate impact on people of color.

Some argue that human backup identification (a person who verifies the computer's identification) can counteract false positives. However, research shows that, if people lack specialized training, they make the wrong decisions about whether a candidate photo is a match about half the time. Unfortunately, few systems have specialized personnel review and narrow down potential matches. Face recognition can be used to target people engaging in protected speech. For example, during protests surrounding the death of Freddie Gray, the Baltimore Police Department ran social media photos through face recognition to identify protesters and arrest them. Of the 52 agencies analyzed in a report by Georgetown Center on Privacy and Technology, only one agency, the Ohio Bureau of Criminal Investigation, has a face recognition policy expressly prohibiting the use of the technology to track individuals engaged in protected free speech. Few face recognition systems are audited for misuse. Of 52 agencies surveyed by Georgetown that acknowledged using face recognition, less than 10% had a publicly available use policy. Only two agencies (the San Francisco Police Department and the Seattle region's South Sound 911) restrict the purchase of technology to those that meet certain accuracy thresholds. Only one—Michigan State Police—provides documentation of its audit process. There are few measures in place to protect everyday Americans from the misuse of face recognition technology. In general, agencies do not require warrants, and many do not even require law enforcement to suspect someone of committing a crime before using face recognition to identify them. The Illinois Biometric Information Privacy Act requires notice and consent before the private use of face recognition tech. However, this only applies to companies and not to law enforcement agencies.

CHAPTER 8

Advanced Use Cases of Power BI

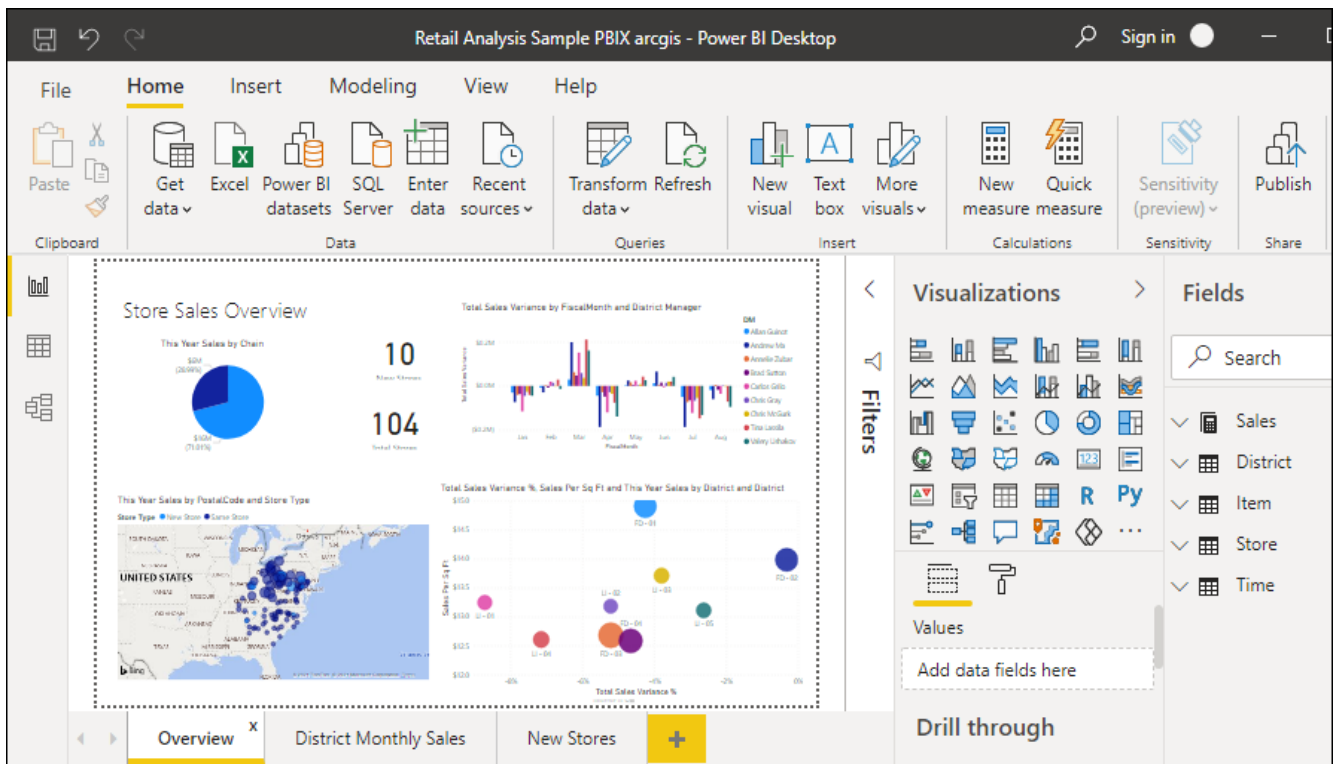


Power BI is a Data Visualization and Business Intelligence tool that converts data from different data sources to interactive dashboards and BI reports. Power BI suite provides multiple software, connector, and services - Power BI desktop, Power BI service based on S.a.a.s, and mobile Power BI apps available for different platforms. These set of services are used by business users to consume data and build BI reports.

Power BI desktop app is used to create reports, while Power BI Services (Software as a Service - SaaS) is used to publish the reports, and Power BI mobile app is used to view the reports and dashboards.

Power BI is a Data Visualization and Business Intelligence tool that converts data from different data sources to interactive dashboards and BI reports. Power BI suite provides multiple software, connector, and services - Power BI desktop, Power BI service based on SaaS, and mobile Power BI apps available for different platforms. These set of services are used by business users to consume data and build BI reports. This tutorial covers all the important concepts in Power BI and provides a foundational understanding on how to use Power BI.

8.1 Dash boards



Data and information visualization (data viz or info viz) is an interdisciplinary field that deals with the graphic representation of data and information. It is a particularly efficient way of communicating when the data or information is numerous as for example a time series.

It is also the study of visual representations of abstract data to reinforce human cognition. The abstract data include both numerical and non-numerical data, such as text and geographic information. It is related to infographics and scientific visualization. One distinction is that it's information visualization when the spatial representation (e.g., the page layout of a graphic design) is chosen, whereas it's scientific visualization when the spatial representation is given.

From an academic point of view, this representation can be considered as a mapping between the original data (usually numerical) and graphic elements (for example, lines or points in a chart). The mapping determines how the attributes of these elements vary according to the data. In this light, a bar chart is a mapping of the length of a bar to a magnitude of a variable. Since the graphic design of the mapping can adversely affect the readability of a chart, mapping is a core competency of Data visualization.

Data and information visualization has its roots in the field of statistics and is therefore generally considered a branch of descriptive Statistics. However, because both design skills and statistical and computing skills are required to visualize effectively, it is argued by authors such as Gershon and Page that it is both an art and a science.

- **What Are Interactive Dashboards?**

An interactive dashboard is a data visualization tool that allows business teams to track, analyze, and display metrics of various sorts. Dashboards feature charts, tables, maps, and other visualizations to help viewers understand the story the data tells.

But not all dashboards are created equal. Static dashboards offer only surface-level detail, leaving users adrift when they want to ask follow-up questions or see alternative outcomes based on various criteria. Business teams are starting to demand more from their dashboards.

As organizations work to become more data-driven, people must be able to access the insights they need for daily decision-making. Their dashboards must allow them to find the answers to follow-up questions or conduct what-if analyses. A good interactive dashboard will empower data exploration, enabling both technical and non-technical users to leverage business intelligence to make better decisions. (More on this later.)

- **How to Efficiently Create An Interactive Dashboard in a Day**

Dashboards shouldn't take extensive amounts of time to build. With a well-designed analytics tool, creating an interactive dashboard in a day is simple. In fact, the process should only take a few minutes. And with the right tool, once you've built the dashboard, it will continuously update, automatically — so you can set it and forget it, so to speak.

But making sure your dashboard will deliver the insights it needs requires a bit of strategy. Here's a three-step process for creating interactive dashboards more efficiently.

1. Ask four foundational questions

Before you start the process of building a reporting dashboard, you'll want to answer four key questions. Getting clear on what you need from the dashboard before you begin will ensure you don't get sidetracked with irrelevant information.

- Which questions are you trying to answer?
- What are you trying to measure?
- Which metrics matter?
- What will you do with the interactive analytics dashboard, and who needs access?

2. Determine how you'll visualize the data

Once you've answered these four questions, you'll need to determine how you'll visualize the data. Your visualizations should be interactive and make it easy to see trends and digest insights. Using the most appropriate visualization will help you communicate the data's story most efficiently.

- **Charts and Graphs**

Charts and graphs have been around so long for a reason: they're effective. These visualizations are best used for univariate data and descriptive analytics. Common types of charts and graphs include bar graphs, line graphs, area charts, and pie charts.

- **Diagrams**

When your data is hierarchical or multidimensional, a diagram helps you demonstrate complex data relationships. For example, to get a holistic picture of your organization's interactions with a customer, you might use a network diagram, block diagram, or tree diagram.

- **Matrices and Heatmaps**

Matrices and heatmaps are also good options for multidimensional data visualization. With a heatmap, gradients show the strength of a correlation. With a pairwise scatterplot matrix, you can observe potential relationships or patterns. Black and white scatterplots can reveal two dimensions, while adding color will allow you to visualize three dimensions.

- **Geographic Maps**

If you want to display geographic data by location, a map is an excellent visualization technique. It's important to note that you'll want to limit your data points when working with geographic maps, or the visual will become cluttered.

3. Decide how you want to share your dashboard

You'll also want to think about how you'll share the dashboard. Is simple sharing sufficient, or will you embed it into a webpage or third-party application? With Sigma, you can share or embed interactive dashboards in seconds.

Going Beyond the Dashboard

Dashboards are an important starting point in the data exploration process. But they're just that — a starting point. As mentioned earlier, business teams must be able to explore the data and derive sufficient insights as soon as possible, if not on the same day.

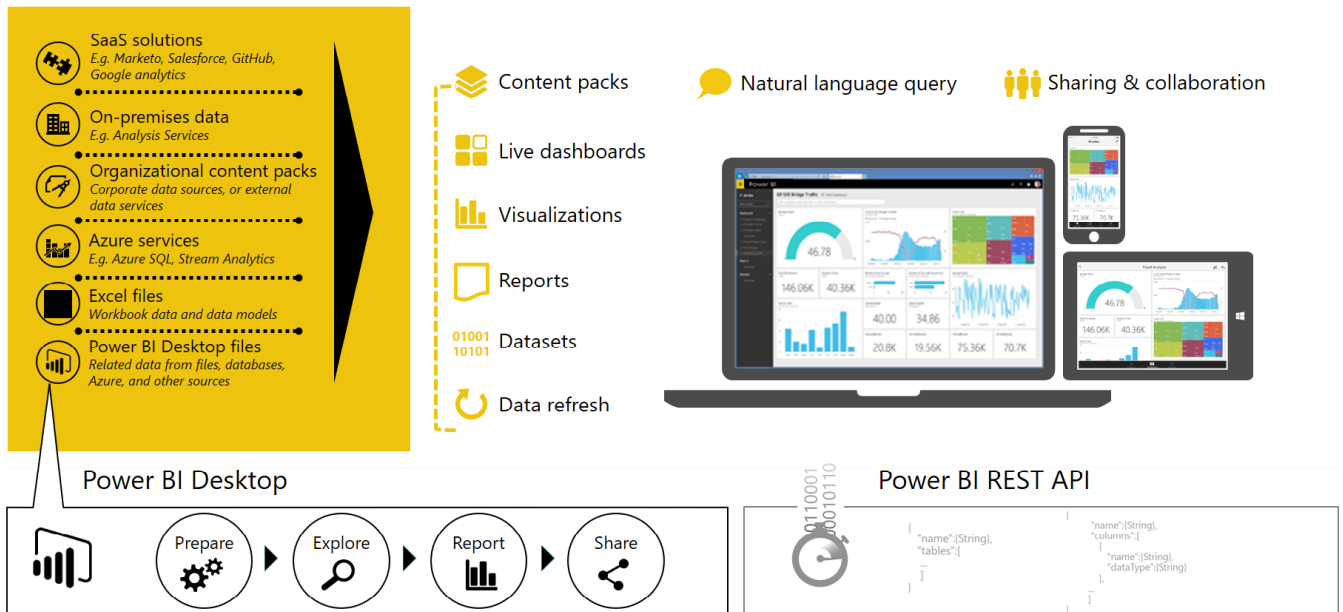
Point-and-click dashboards limit users to a predefined set of business metrics. The result is surface-level reporting on historical data. Without involving the already overwhelmed BI team, there's no ability to ask follow-up questions or see likely outcomes based on various scenarios. It's easy to see why between 60% and 73% of all company data goes unused when traditional dashboards are the norm.

Today's business teams need the ability to dig directly into data — often at the most granular level — to make decisions on-demand. Modern self-service analytics moves beyond the static dashboard and towards interactive dashboards that reveal real-time, highly relevant insights on the same day a question, request, or hypothesis is submitted. The dashboard is being redefined as a BI and cloud data exploration tool that empowers even non-technical users to independently find answers to their pressing questions — without waiting on data teams.

Using Sigma Deep Dive to Power Data Exploration:

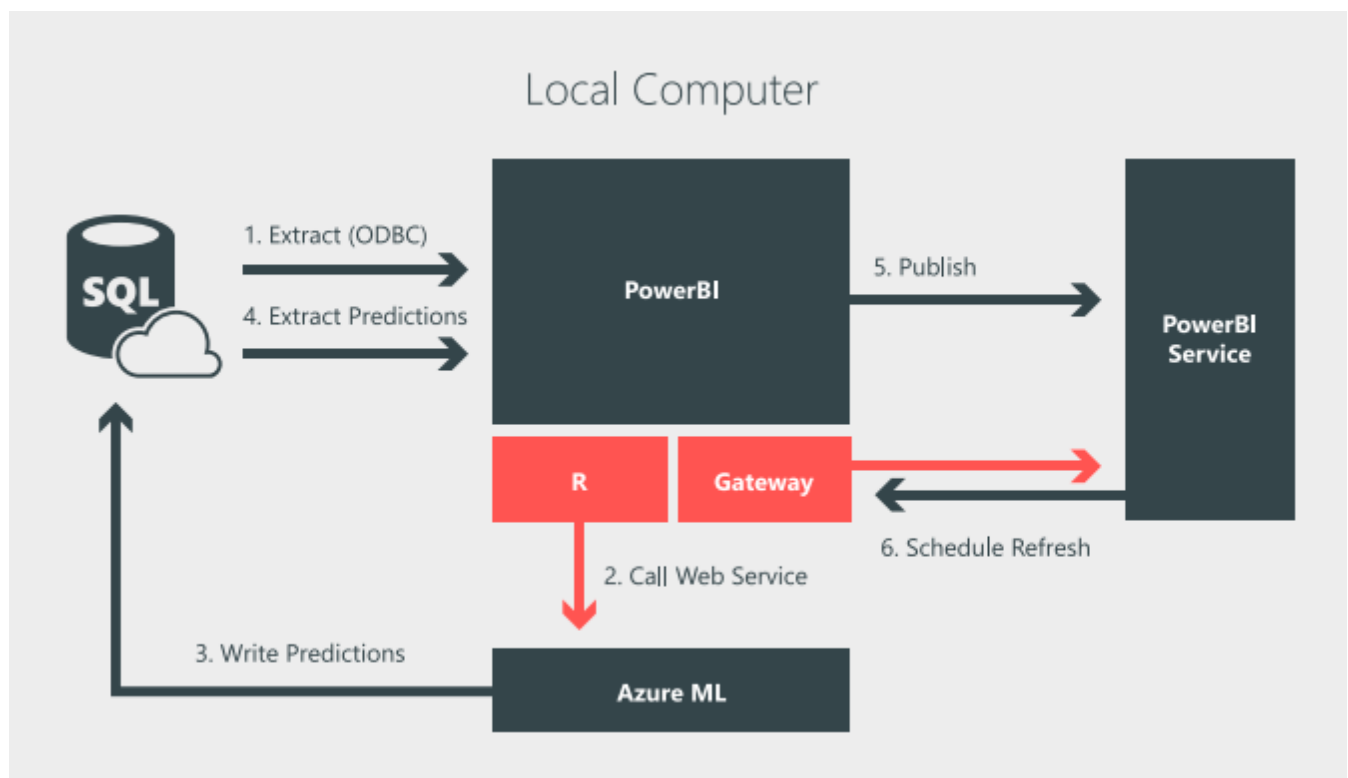
When a team discovers an interesting trend in a dashboard, they usually want to know more. What's driving the trend? What will happen to the trend if a variable changes? This is where chart drill-downs are essential. Drill-downs are interactive and allow you to explore data slices easily. Team members can click on a visualization and see hierarchies of data based on what they selected.

In Sigma, each visualization is tied to a worksheet. If you have follow-up questions, you can query the data directly to help you better understand what's going on. Business teams can truly do their own analyses, exploring the underlying data in just a few clicks. With Sigma's spreadsheet-like UI, team members can slice, dice, filter, and calculate data using the same format, functions, and formulas as traditional spreadsheets — empowering users to drill down to record-level detail without writing SQL or proprietary code.



DESCRIPTION: Using Sigma’s intuitive dashboard builder allows anyone to build a powerful, flexible, and interactive dashboard in minutes, if not a day. And any of the underlying data in the dashboard worksheet can be accessed in a single click, making it easy to iterate and update your dashboard for whatever needs arise.

8.2 Advanced Analytics with Power BI



Imagine if you could review the latest output of your organization's fraud model on demand, or analyze the sentiment of social media users who tweet or post about your products. Power BI brings the predictive power of advanced analytics to allow users to create predictive models from their data, enabling organizations to make data-based decisions across all aspects of their business. Through machine learning, computers are able to act without being explicitly programmed. Instead, they can teach themselves to grow and change when exposed to new data. Once the work of science fiction, machine learning is rapidly becoming part of our daily lives— through practical speech recognition programs, more effective web searches, and even self-driving cars. Using Azure Machine Learning Studio, users can quickly create predictive models by dragging, dropping, and connecting data modules. Power BI then allows users to visualize the results of their machine learning algorithm. To accomplish this in Power BI, first use R to extract data from Azure SQL that has not yet been scored by the machine learning model. Next, use R to call the Azure Machine Learning web service and send it the unscored data. Write the output of the Azure Machine Learning model back into SQL and use R to read scored data into Power BI. Then, publish the Power BI file to the Power BI service. Finally, use the Personal Gateway to schedule a refresh of the data, which triggers a scheduled rerun of the R script and brings in the new predictions.

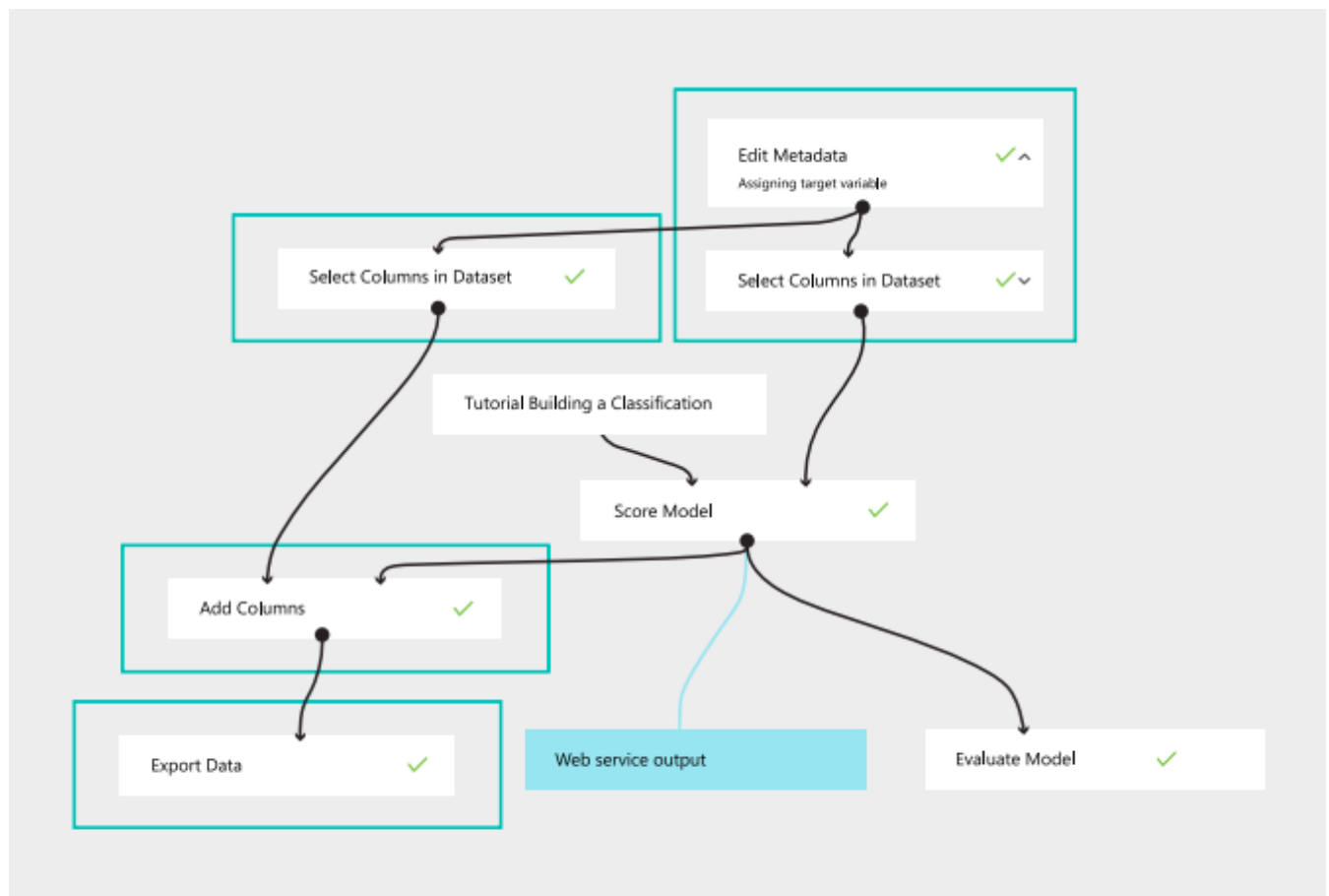


Fig: process of prediction

Data streaming in Power BI:

- Real-time dashboards

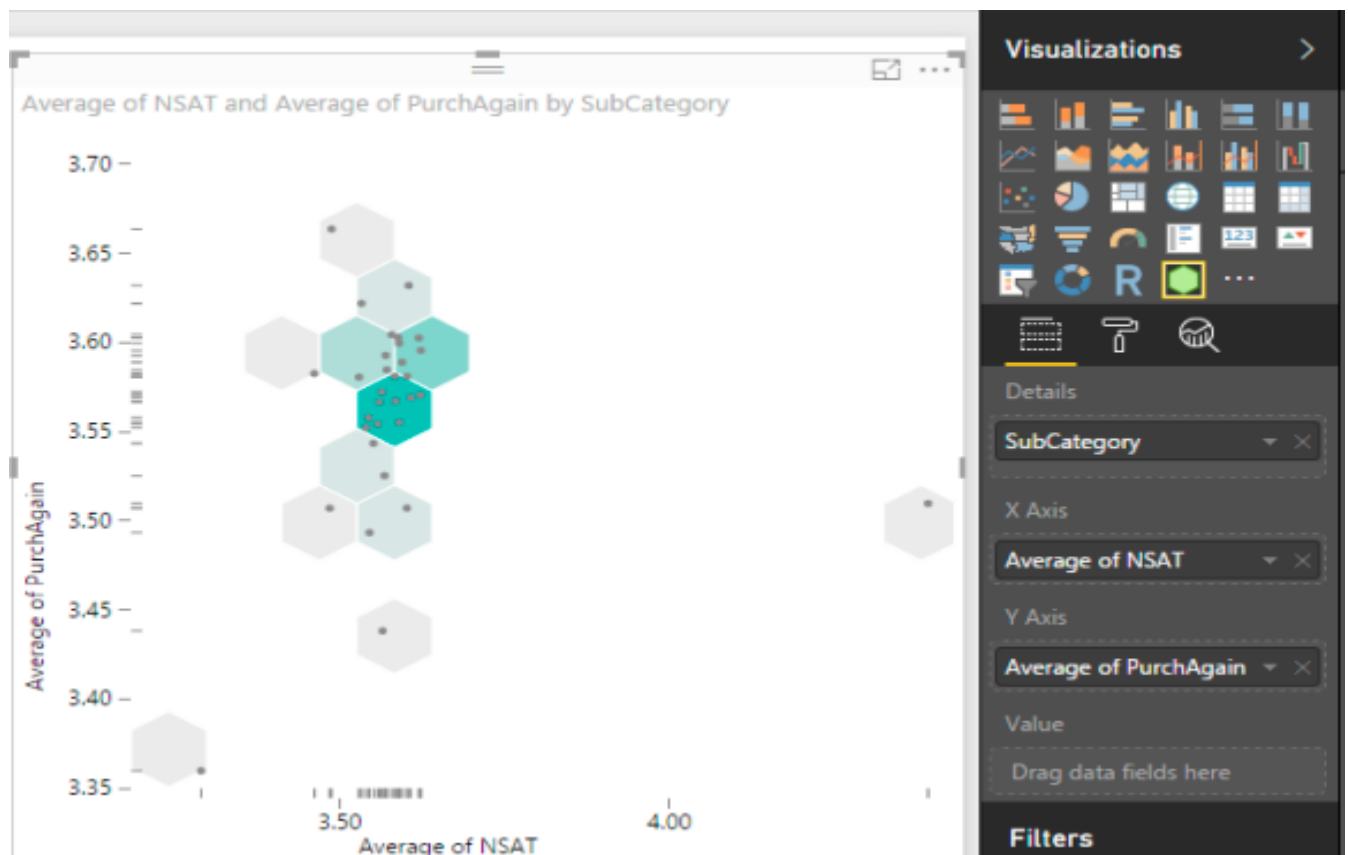
Power BI lets you easily display and analyze your real time data, empowering your organization to gain instant insights from time-sensitive information. Monitor social media campaigns as they go viral. Display streaming video on your dashboards. Bring your dashboards to life with IoT sensor readouts. The rich functionality of Power BI, combined with the velocity of real-time data, will transform the way you do business. Real-time data is all around us, and Power BI features are designed to help you get to that data with minimum setup cost. Integration with streaming data providers such as Azure Event Hub, PubNub, and Temboo allows Power BI to get to your real-time data—no matter where it lives. In addition, Power BI allows you to dive deep into your real-time data through integration with Microsoft Azure Stream Analytics and Azure Machine Learning. Predictive intelligence can help you take proactive action to stay on the right course, and stream analytics can shape and aggregate your data. Data streams are stored in the cloud for historical analysis and visualization.

- **Setup of real-time streaming data sets**

With Power BI real-time streaming, you can stream data and update dashboards in real time. Any visual or dashboard that can be created in Power BI can also be created to display and update real-time data and visuals. The devices and sources of streaming data can be factory sensors, social media, service usage metrics, and anything else from which time-sensitive data can be collected or transmitted. Streaming data can be consumed two ways in Power BI: as tiles with visuals from streaming data, or as data sets created from streaming data that persist in Power BI.

- **Custom visualizations**

Custom visualization in Power BI allows you to create full custom visuals to add to reports or submit to the Power BI community for others to use. You can create a custom visual using Power BI developer tools, which let you design and test a custom visual by writing custom visual TypeScript code and creating CSS. Learn more: [Power BI Custom Visuals Gallery](#) Once you've tested your custom visual, you can export it to your Power BI dashboard, or submit it to the Power BI visuals gallery.



- **Data Analysis Expressions:**

Data Analysis Expressions (DAX) is a collection of functions, operators, and constants that can be used in a formula or expression to calculate and return one or more values. DAX helps you create new information from data that is already in your model. Learning how to create effective DAX formulas will help you get the most out of your data. It's quite easy to create a new Power BI Desktop file and import data into it. You can even create reports that show valuable insights without using any DAX formulas at all.

Here is an example of DAX in action.

Previous Quarter Sales = CALCULATE(SUM(Sales[SalesAmount]),
PREVIOUSQUARTER(Calendar[DateKey]))

This formula will calculate the total sales for the previous quarter, depending on the filters applied in a report. For example, if we put SalesAmount and our new Previous Quarter Sales measure in a chart, and then added Year and QuarterOfYear as slicers, we'd get something like this: Data Analysis Expressions

But, what if you need to analyze growth percentages across product categories and for different date ranges? Or, you need calculate year-over-year growth compared to market trends? DAX formulas provide this capability and many other important capabilities as well. Learning how to create effective DAX formulas will help you get the most out of your data. When you get the information you need, you can begin to solve real business problems that affect your bottom line. This is the power in Power BI, and DAX will help you get there.

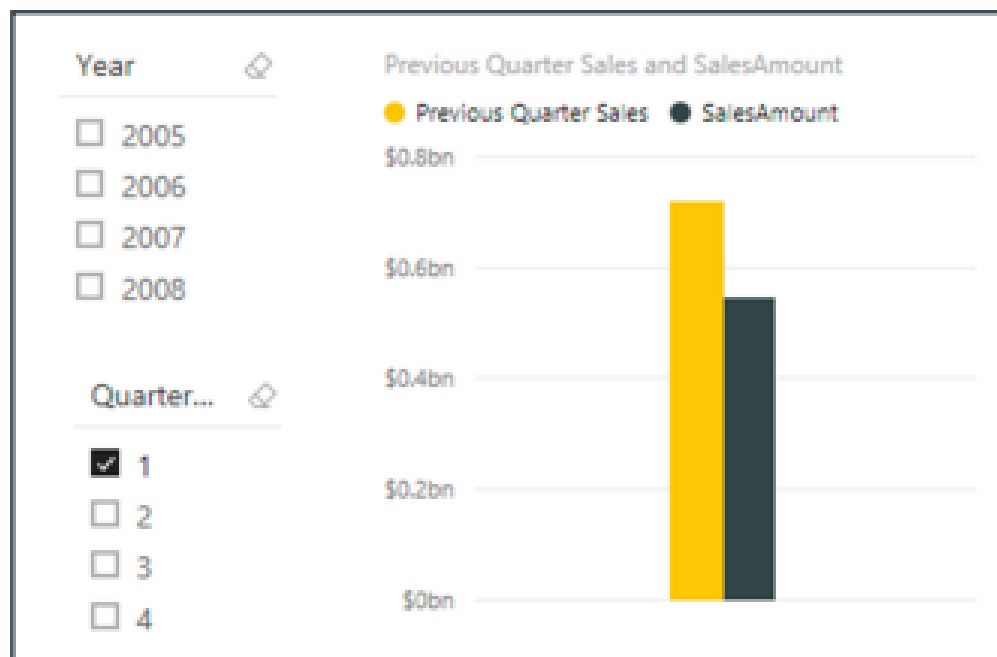


Fig: Output of above DAX code

Power BI business analytics tools enable users at all levels of an organization to analyze data and share insights. Through dashboards, Power BI provides a 360-degree view of your most important metrics—in one place, updated in real time, and available on all of your devices. With one click, explore data using intuitive tools to quickly find answers and Additional resources: Explore Power BI Download for free Guided learning uncover new insights. Power BI allows you to dig deep into your data, while being productive and creative with what you build. With more than 20 built-in visuals and a gallery of vibrant custom visualizations, Power BI makes it easy to use advanced analytics to effectively communicate your message and address business challenges.



Fig : Sample Dash Board

Users can calculate their usage by selecting the appliances they use and how much time they use per day. Then they can choose their state or location to estimate the cost of their electricity bill. The dashboard showcases the following

- Total Annual Energy (kWh)
- Total Annual Cost (\$)
- Annual Costs by Category or Appliance
- Annual Energy by Appliance

CHAPTER 9

Smart Civilization Planning for Future Estimation of Waste Production

9.1 Our Dash Board

Our plan and motive for developing this dash board is for estimating the production of waste in the future from our current 2022 to 2034. We designed this dash board from excel analysis based on math which is used to calculate future population analyzed the future waste production for each year from 2022 to 2034. which will help in municipal corporations to plan ahead for growing production of waste which is in increment or decrement according to future requirements.

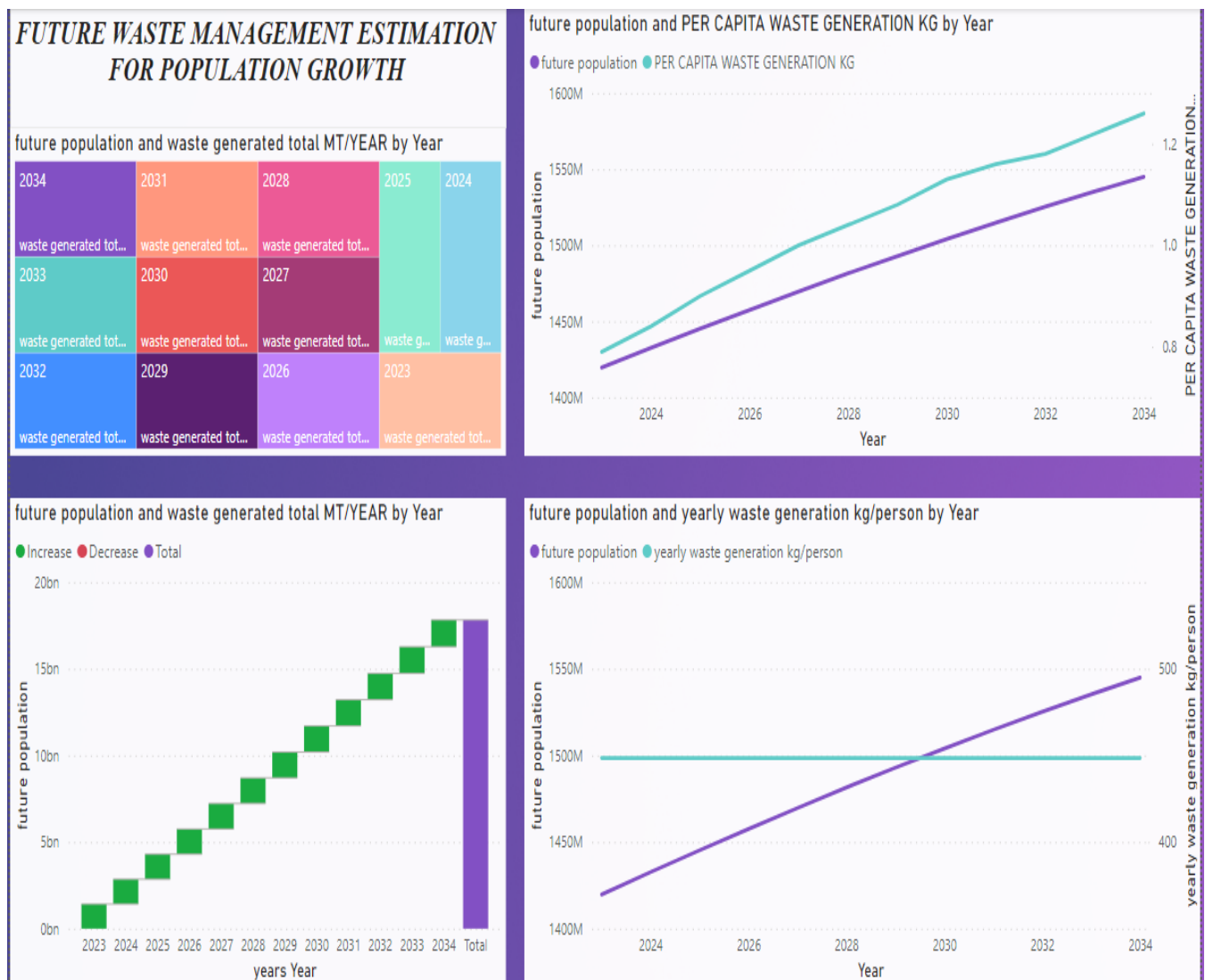
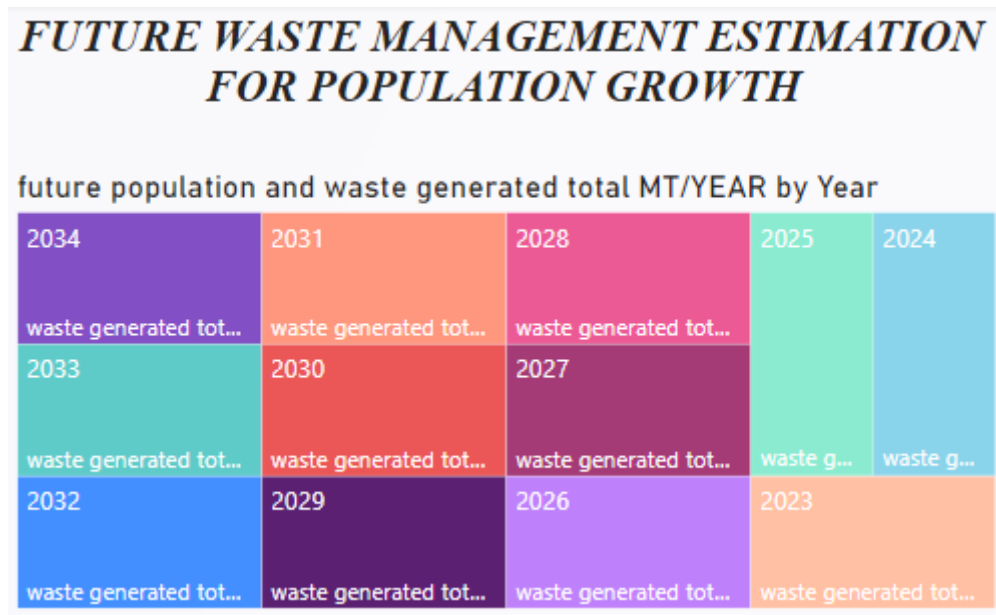


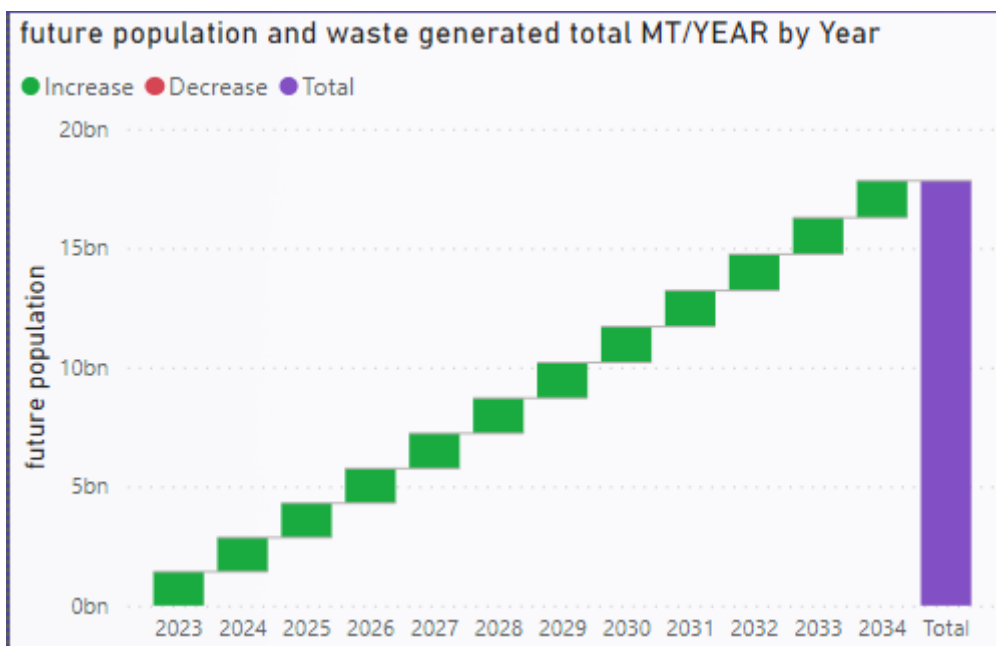
Fig: Dash board

9.2 Selection of year



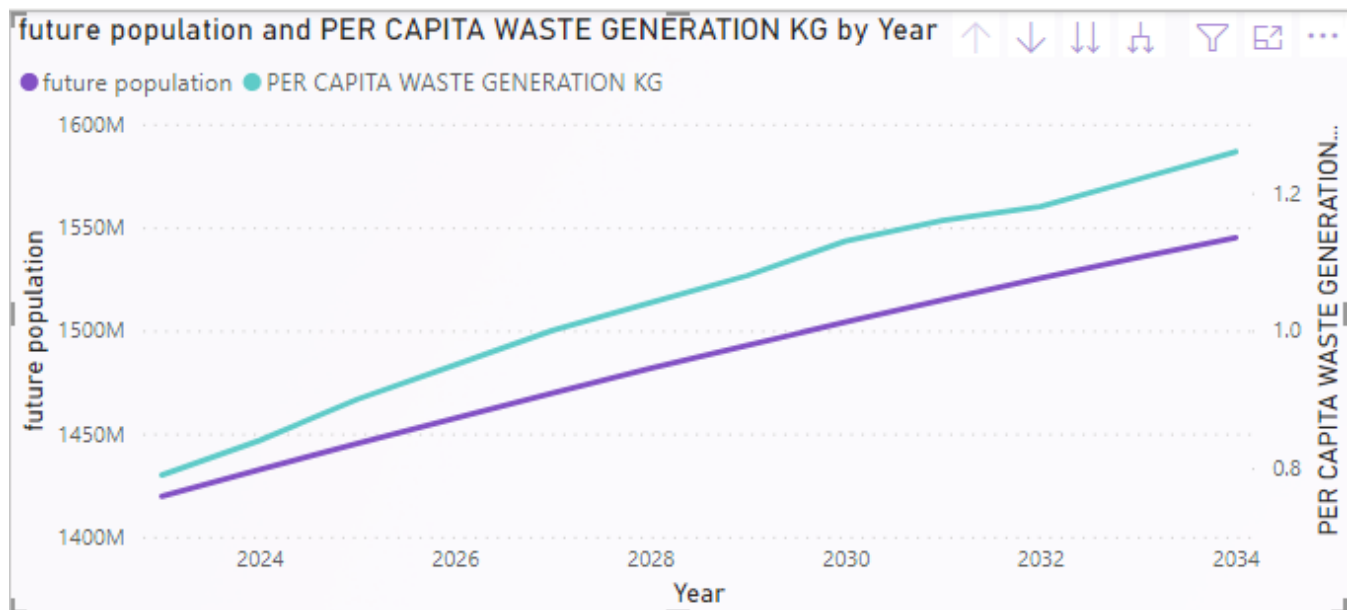
This visual in our dash board contains buttons when user clicks on the specific year the dash board gives estimated insight of that year.

9.3 Visual For Total production in step by step:



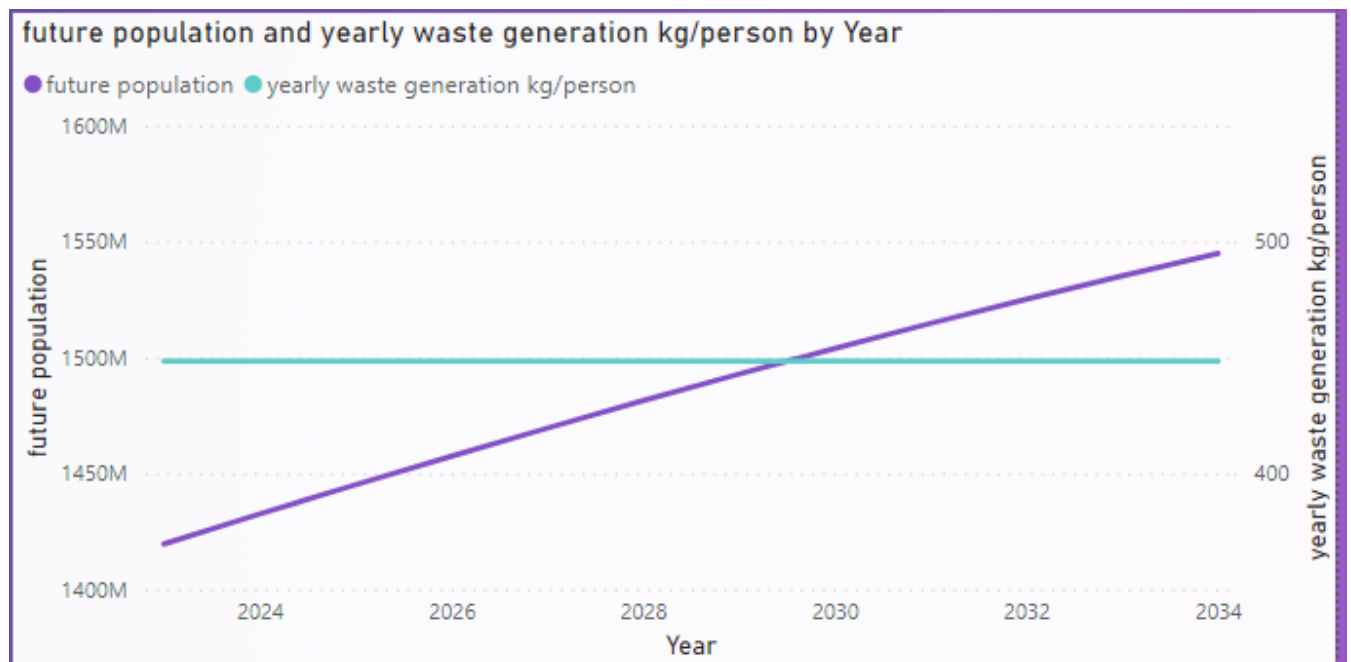
This visual in dash board will show the total production year wise and every year based total production of waste from year 2023 to 2034

9.4 Graph of future population with per capita waste generation



DESCRIPTION: This visual in dash board will give user idea about per capita waste generation of future population year wise

9.5 Visual of future population with kg/person in upcoming years



DESCRIPTION: This visual in our dash will give idea about future population and yearly waste generation and how it can grow or decrease in upcoming years

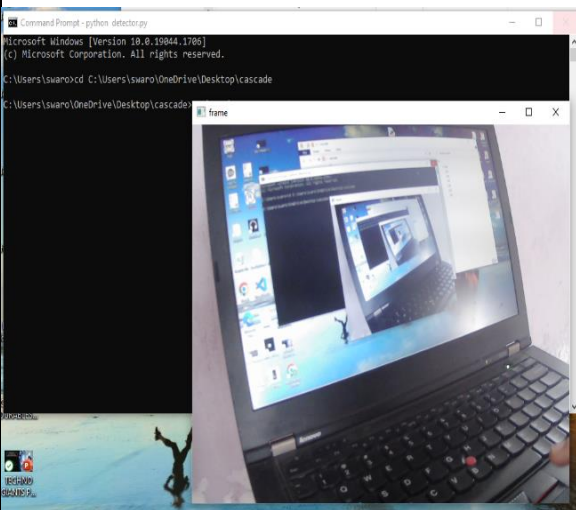
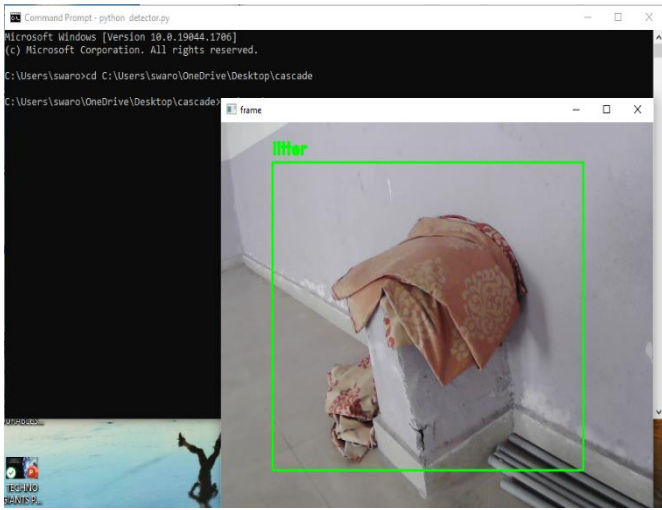
Chapter 10

Detecting garbage with computer vision

Along with analysis of future data and planning dashboard. We developed a computer vision system which detects the garbage in its location. which helps in automated cleaning and keeping our premises clean from analysis we done on waste production which is more in future we are giving a new system to detect the garbage without human intervention with the use of computer vision we are going to detect the garbage.

- Our program will detect the solid waste in its sight.

10.1 validation table

s.n	Don't Detect Useful Items	Detects Garbage
1.		
2.	Description: when cam is pointed to laptop it didn't detect anything to worry about the above is camera sight when its is fired to detect a garbage	Description: the above image is a camera detecting solid waste material which detects the garbage on road or where it is used on its sight when it is used to detect some garbage at a particular location

CHAPTER 11

CONCLUSION AND FUTURE ENHANCEMENTS

Litter is associated with many social, environmental, and economical issues. It can also be a source of safety hazards for road users. The issue of litter is a complex matter caused by various reasons which do not have a single solution. Removal methods focus on collecting the existing litter which can be very effective on a short-term basis but they are usually costly. Furthermore, roadside litter is a growing problem due to the increase in VMT and contributing factors such as the emergence of the COVID-19 pandemic. Therefore, people are littering much faster than DOTs and volunteer programs can pick it up. Preventive methods, on the other hand, following a longer-term approach that might even take up to a generation to be completely effective. A long-term effort for mitigating this problem must include all the discussed methods to gain maximum success. To change the littering behaviors, engaging communities, continuous educational programs, encouraging volunteer works, improving the design of the trash receptacles, revising legislation, and establishing more effective enforcement methods are the major effective strategies. In terms of public awareness, it is also very important to choose the proper methods to maximize the audience and send the right message to the right group. Social media, especially if it is accompanied by celebrities' endorsement is expected to be a strong tool. Although a change in littering attitude is fundamental for any litter abatement program, the role of producers should not be ignored either. The drastic surge in the amount of plastic litter in the last 50 years is a direct result of the increase of the products, and producers' choice of packaging material. Governments need to encourage, or even enforce industries to think and act with a more sustainable approach.

FUTURE ENHANCEMENTS:

As we added the dashboard for the garbage or Litter detection for getting the Litter percentage. In Future we can also give the alert for an area if the percentage in the dashboard shows more that helps in easy reducing of litter.

- Arrange a sound machine in garbage office that helps the officers to clear the litter easily from that area

12. REFERENCES

• General

1. Q. Fang, C. Xu, J. Sang, M. S. Hossain, and A. Ghonim, “Folksonomy-based visual ontology construction and its applications,” *IEEE Transactions on Multimedia*, vol. 18, no. 4, pp. 702–713, 2016.
View at: [Publisher Site](#) | [Google Scholar](#)
2. H. Rezatofighi, N. Tsoi, J. Gwak, A. Sadeghian, I. Reid, and S. Savarese, “Generalized intersection over union: a metric and a loss for bounding box regression,” in *Proceedings of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 658–666, Long Beach, CA, USA, June 2019.
View at: [Google Scholar](#)
3. Z. Zheng, P. Wang, W. Liu, and D. Ren, “Distance-IoU loss: faster and better learning for bounding box regression,” *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 7, pp. 12993–13000, 2020.
View at: [Publisher Site](#) | [Google Scholar](#)
4. S. Woo, J. Park, J. Y. Lee, and I. S. Kweon, “CBAM: convolutional block attention module,” in *Proceedings of the ECCV, Munich, Germany, September 2018*.
View at: [Google Scholar](#)
5. Q. Wang, B. Wu, P. Zhu, P. Li, W. Zuo, and Q. Hu, “ECA-net: efficient channel attention for deep convolutional neural networks,” in *Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11531–11539, Seattle, WA, USA, June 2020.
View at: [Google Scholar](#) Copyright.
6. Girshick, R., Donahue, J., Darrell, T., & Malik, J. (2014). Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 580–587).
7. He, K., Zhang, X., Ren, S., & Sun, J. (2015). Spatial pyramid pooling in deep convolutional networks for visual recognition. *IEEE transactions on pattern analysis and machine intelligence*, 37, 1904–1916.