

DEEP LEARNING BASED OBJECT IDENTIFICATION AND GAZE ANALYSIS

Submitted in partial fulfillment of the requirements for
the award of
Bachelor of Engineering degree in Computer Science and Engineering

by

CHUNDURI LAKSHMI VENKATESH (17SCS0095)



**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
SCHOOL OF COMPUTING**

SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)
Accredited with Grade "A" by NAAC
JEPPIAAR NAGAR, RAJIV GANDHI SALAI,
CHENNAI – 600 119

MARCH - 2021



SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)

Accredited with "A" grade by NAAC

Jeppiaar Nagar, Rajiv Gandhi Salai, Chennai – 600 119

www.sathyabama.ac.in



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that this project report is the bonafide work of **CHUNDURI LAKSHMI VENKATESH (17SCS0095)** who carried out the project entitled "**DEEP LEARNING BASED OBJECT IDENTIFICATION AND GAZE ANALYSIS**" under my supervision from **November 2020 to March 2021**.

Internal Guide

Ms. R. Aishwarya, M.E.,

Head of the Department

Dr.L.Lakshmanan M.E., Ph.D

Dr. S. VIGNESHWARI, M.E., Ph.D.

Submitted for Viva voce Examination held on _____

Internal Examiner

External Examiner

DECLARATION

I **CHUNDURI LAKSHMI VENKATESH** hereby declare that the Project Report entitled “**DEEP LEARNING BASED OBJECT IDENTIFICATION AND GAZE ANALYSIS**” is done by me under the guidance of **Ms. R. Aishwarya, M.E.**, Department of Computer Science and Engineering at Sathyabama Institute of Science and Technology is submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering degree in Computer Science and Engineering.

DATE:

PLACE: CHENNAI

SIGNATURE OF THE CANDIDATE

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to **Board of Management** of **SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. T. Sasikala, M.E., Ph.D., Dean**, School of Computing, **Dr. S. Vigneswari, M.E., Ph.D., and Dr. L. Lakshmanan, M.E., Ph.D., Heads of the Department** of Computer Science and Engineering for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide **Ms. R. Aishwarya, M.E., Assistant Professor**, for her valuable guidance, suggestions and constant encouragement paved way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **Department of Computer Science and Engineering** who were helpful in many ways for the completion of the proj

ABSTRACT

The Internet of Things (IoT) makes an attempt to assist individuals access internet-connected devices, applications, and services anytime and anyplace. However, providing associate degree economical associate degree intuitive methodology of interaction between individuals and IoT devices continues to be an challenge. during this work, we have a tendency to propose a unique interaction system referred to as Watch & Do, wherever users will management associate degree IoT device by gazing at it and doing straightforward gestures. The projected system principally consists of 4 classes as object detection module, gaze estimation module, hand gesture recognition module and IoT controller module. The device is known by numerous deep learning-based gaze estimation and object detection techniques. Afterwards, hand gesture recognition is applied to get associate degree IoT device management command that is transmitted to the IoT platform. The experimental results and case studies demonstrate the practicableness of the projected system and the long run analysis direction

TABLE OF CONTENTS

CHAPTER	TITLE	PAGE NO.
	ABSTRACT	I
	LIST OF FIGURES	V
	LIST OF TABLES	Vi
	LIST OF ABBREVIATIONS	Vii
1.	INTRODUCTION	2
	1.1. OBJECTIVE	2
	1.2. SCOPE OF THE PROJECT	2
	1.3. LITERATURE SURVEY	3
2.	SYSTEM ANALYSIS	6
	2.1. EXISTING SYSTEM	7
	2.1.1. DISADVANTAGES	7
	2.2. PROPOSED SYSTEM	7
	2.2.1. ADVANTAGES	8
	2.3. FEASIBILITY STUDY	8
	2.3.1. ECONOMIC FEASIBILITY	8
	2.3.2. TECHNICAL FEASIBILITY	8
	2.3.3. SOCIAL FEASIBILITY	9
3.	SYSTEM SPECIFICATIONS	11
	3.1. HARDWARE SPECIFICATIONS	11
	3.2. SOFTWARE SPECIFICATIONS	11
4.	SOFTWARE DESCRIPTION	13
	4.1. FRONT END	13
	4.2. BACK END	13

5.	MODULE DESCRIPTION	17
	5.1. PROJECT DEFINITION	17
	5.2. OVERVIEW OF THE PROJECT	17
	5.3. MODULES	18
	5.3.1. OBJECT DETECTION	18
	5.3.2. GAZE ESTIMATION	18
	5.3.3. HAND GESTURE RECOGNITION	18
	5.4. ARCHITECTURE DIAGRAM	19
	5.5. UML DIAGRAMS	20
	5.5.1. Sequence Diagram	20
	5.5.2. Activity Diagram	21
	5.5.3. Collaboration Diagram	22
	5.5.4. Use Case Diagram	23
	5.6. DATABASE DESIGN	24
	5.7. INPUT DESIGN	24
	5.8. OUTPUT DESIGN	24
6.	SYSTEM TESTING	26
	6.1. TESTING OBJECTIVES	26
	6.2. TESTING LEVELS	26
	6.2.1. UNIT TESTING	26
	6.2.2. FUNCTIONAL TESTING	26
	6.2.3. PERFORMANCE TESTING	26
	6.2.4. STRESS TESTING	27
	6.2.5. STRUCTURED TESTING	27
	6.2.6. INTEGRATION TESTING	27
	6.2.7. EXPERIMENTAL OUTCOME	28

7.	SYSTEM IMPLEMENTATION	30
	7.1. IMPLEMENTATION PROCEDURE	30
	7.2. ALGORITHM	30
	7.2.1. R-CNN ALGORITHM	30
	7.2.2. CNN ALGORITHM	30
8.	CONCLUSION AND ENHANCEMENTS	33
	8.1. CONCLUSION	33
	8.2. FUTURE ENHANCEMENTS	33
	APPENDICES	33
	APPENDIX-1 SCREENSHOTS	35
	APPENDIX-2 IMPLEMENTATION CODES	37
	REFERENCES	48

LIST OF FIGURES

FIGURE NO.	NAME	PAGE NO.
5.1.	Architecture Diagram	17
5.2.	Sequence Diagram	17
5.3.	Activity Diagram	18
5.4.	Collaboration Diagram	19
5.5.	Use Case Diagram	20

LIST OF TABLES

<i>S. No</i>	<i>Table Title</i>	<i>Page No.</i>
6.1.	Experimental Outcome	27

LIST OF ABBREVIATIONS

<i>Abbreviation</i>	<i>Full form</i>
CNN	Convolutional Neural Network
R-CNN	Regional Convolutional Neural Network
SVM	Supporting Vector Machine

CHAPTER 1

INTRODUCTION

INTRODUCTION

Nowadays a huge amount of multimedia is generated in desperate manners with different devices and uploaded. During the upload, they are shared with users and ultimately, viewed or downloaded.

The technique is based on an assumption that each social network applies a peculiarly unknown manipulation method leaving some distinctive traces on the image; to be extracted to feature every platform, without revealing the identity of the user who downloads the image.

1.1. OBJECTIVE

To detect target device is identified by various deep learning-based gaze estimation and object detection techniques and user can control the target IOT device by hand gesture.

1.2. SCOPE OF THE PROJECT

The Web of Things attempts to help individuals with finding decent pace gadgets, applications, and associations at whatever point and any place. Notwithstanding, how giving a gainful and instinctual methodology for the relationship among individuals and IoT gadgets is so far an open test. Right now, propose a novel composed exertion framework called Watch and Do, where clients can control an IoT contraption by looking and doing fundamental signs. The proposed adventure contains four modules

I.e. Thing detection (CNN(R-CNN)) algorithm, look estimation hand movements and voice affirmation. The customer can perceive the article using voice affirmation and can control the thing using hand movements. Sometime later, hand signal assertion is applied to convey an IoT contraption control bearing which is transmitted to the IoT organize. The test results and pertinent examinations show the reachability of the proposed framework and prescribe future research heading.

1.3. LITERATURE SURVEY

1.3.1. Charith Perera, Arkady Zaslavsky, “Context-aware computing, learning, and big data in Internet of Things: A survey,” 2013.

Understanding sensor data is one of the main challenges that the IoT would face. This vision has been supported and heavily invested by governments, interest groups, companies, and research institutes. The IoT has gained significant attention over the last few years. With the advances in sensor hardware technology and cheap materials, sensors are expected to be attached to all the objects. Analyze, compare and consolidate past research work but also to appreciate their findings and discuss their applicability towards the IoT. The data we collect may not have any value unless we analyze, interpret, and understand it.

1.3.2. Chuvu Wang , Jian Liu “Multi-Touch in the Air :Device-free Finger Tracking and Gesture Recognition via COTS RFID”,2018.

These solutions are either easily affected by the environmental noise or incapable of sensing fine-grained gestures at the finger level. We experimentally investigate the impact of tag array deployment on the signal quality. Gesture recognition has gained considerable attention in emerging applications to provide a better user experience for human-computer interaction. Experiments validate that RF-finger can achieve as high as 88% and 92% accuracy for finger tracking and multi-touch gesture recognition.

1.3.3. C. Perera , A. Zaslavsky , M. Compton, P. Christen, and D. Georgakopoulos ,“Semantic-driven configuration of Internet of Things middleware”, 2013.

we have introduced the CASCoM approach that allows non-IT experts to configure IoT middleware efficiently and effectively. Our model supports single-click configuration by eliminating sequences of manual activities needed to be carried out by users otherwise. The problem we addressed in this paper is comprehensively analyzed and presented with use-case scenarios in detail Our objective is to automate and simplify the configuration of an IoT middleware and improve its usability so non-IT

experts can use it efficiently and effectively.

1.3.4. Gururaj Kulkarni, S. H Manoor ,“ Enabling Technologies, Protocols, and Applications: A Detailed Survey on IOT”,2017.

There are various issues which need thorough research to improve the quality of life. An overview of IoT technologies, protocols and applications and related issues with a comparison of other papers. IoT has potential to broaden its horizon by enabling communication between smart objects. IoT will change everything drastically if implemented successfully. We presented some handsome application of IoT and its comfort in life. Some important issues that needed to be resolved have been discussed before wide acceptance of this technology.

1.3.5. Luigi Atzori , Antonio Iera,“The Internet of Things: A survey”,2010.

Each IoT element so as to make it possible to reach them from any other node of the network, looks more suitable for the traditional Internet paradigm. The IoT has the potential to add a new dimension to this process by enabling communications with and among smart objects, thus leading to the vision of “anytime, anywhere, any media, anything” communications. Technologies make the IoT concept feasible but do not fit well with the scalability and efficiency requirements. It is emphasized that “to the extent that everyday objects become information security risks, the IoT could distribute those risks far more widely than the Internet has to date”.

1.3.6. Nataniel Ruiz Eunji Chong,“Fine-Grained Head Pose Estimation Without Keypoints”,2017.

We show that a multi-loss deep network can directly, accurately and robustly predict head rotation from image intensities. Landmark-to-pose methods are studied in this work to show their dependence on extraneous factors such as head model and landmark detection accuracy.

Head pose recovery inherently is a two step process with numerous opportunities for error. Generic head models can introduce errors for any given participant, and the process of deforming the head model to adapt to each participant requires significant amounts of data and can be computationally expensive.

1.3.7. R.Vignesh and A.Samydurai ,“Security on Internet of Things (IOT) with Challenges and Countermeasures”,2017.

As against to the fixed Internet, in addition to humans, an IoT fastens a large number of machines, Many other areas that require things to interact over the Internet to executes business assignments cleverly without human involvement The IoT framework is vulnerable to attacks at each layer Each layer of IoT has intrinsic security issues related with it. basic three layer architectural structure of IoT with respect to the devices and technologies that surround each layer.

1.3.8. Xucong Zhang, Yusuke Sugano,“Appearance-Based Gaze Estimation in the Wild”, 2015.

Our CNN-based estimation model significantly outperforms state-of-the-art methods in the most challenging person- and pose-independent training. Our dataset provide a critical insight on addressing grand challenges in daily-life gaze interaction. Existing datasets have been collected under controlled laboratory conditions and methods have been not evaluated across multiple datasets. The generalization capability of learning based methods has not been examined yet.

1.3.9. Zhiang Hu,“Real-Time Head Pose Estimation with Convolutional Neural Networks”,2015.

It is important to find the balancing point of the complexity of CNN so that the network can predict the head poses. CNN can over fit our tried dataset easily and lose position information because of the pooling layers. The input range image and thus cannot generalize to applications where hardware constraints limit computational resources. The head pose space is often discredited, and separate detectors are learned for subsets of poses.

CHAPTER 2
SYSTEM ANALYSIS

SYSTEM ANALYSIS

2.1. EXISTING SYSTEM

In The target device is identified by various Algorithms are used for gaze estimation and object detection techniques. Afterwards, hand gesture recognition is applied to generate an IoT device control command which is transmitted to the IoT platform. This algorithms results are not accuracy. The experimental results and case studies demonstrate the feasibility of the proposed system and imply the future research directions.

2.1.1. Disadvantages

- CNN not accepting the bulk image to estimate object .
- CNN is very low accuracy.

2.2. PROPOSED SYSTEM

Proposed system works as follows. First, the Watch module records the opposite side of the user to detect and recognize the types of IoT devices installed in the room. Second, the Watch module detects the user's head region and then computes a fine- grained head pose information (i.e., pitch, yaw, and roll) to estimate the user's gaze position. With this information, the proposed system can identify the target device. Then, the Do module captures the user's hand gestures. A combination of hand gesture information and the type of selected IoT device is then translated into an IoT command and transmitted to IoT platforms. The target device is identified by deep learning Algorithms are used for gaze estimation and object detection techniques.

2.2.1. Advantages

- In proposed system used R-CNN algorithm .It has high accuracy .
- R-CNN algorithm has capable of accepting bulk images as input and estimate the object.
- R-CNN algorithm is faster than the CNN algorithm.
- the final output of this system is to delivered as voice message. It consists information like type of object and name of the object in front of the user.

2.3. FEASIBILITY STUDY

The feasibility of the project is analyzed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.

2.3.1. Economic Feasibility

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

2.3.2. Technical Feasibility

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

2.3.3. Social Feasibility

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system.

CHAPTER 3
SYSTEM SPECIFICATION

SYSTEM SPECIFICATIONS

3.1. HARDWARE SPECIFICATIONS

Processor	:	Intel I3 and Above
Motherboard	:	Genuine Intel AMD/AMRR
RAM	:	Minimum 4 GB and Above
Hard Disk	:	500 GB and Above
Webcam-2		
Arduino UNO		
Gesture Sensors		
Relay circuit board		

3.2. SOFTWARE SPECIFICATIONS

Operating System	:	Windows 7,8,10(64bit)
Technologies	:	R-CNN and CNN, Python and anaconda.
IDE	:	Jupyter Note Book and Spyder IDE
Front end		Python
Back end		R-CNN, CNN

CHAPTER 4

SOFTWARE DESCRIPTION

SOFTWARE DESCRIPTION

4.1. FRONT END

Python

Python is a widely used general-purpose, high level programming language. It was initially designed by Guido van Rossum in 1991 and developed by Python Software Foundation. It was mainly developed for emphasis on code readability, and its syntax allows programmers to express concepts in fewer lines of code. Python is a programming language that lets you work quickly and integrate systems more efficiently.

4.2. BACK END

Regional CNN

In this chapter we will talk about the Region based methods for object detection. In particular, R-CNN (Regional CNN), the original application of CNNs to this problem, along with its descendants Fast R-CNN, and Faster R-CNN. In classification, there's generally an image with a single object as the focus and the task is to say what that image is. But when we look at the world around us, we carry out far more complex tasks. We see complicated sights with multiple overlapping objects, and different backgrounds and we not only classify these different objects but also identify their boundaries, differences, and relations to one another. The goal of a Region based CNN is to take an image input to detected and localize the object in the image. All three above mentioned architectures were used for object detection and localization for a driving scenario to detect cars, pedestrians and traffic signs etc. and performed a time study to conclude which of the three architectures can be used as a detector for a real- world driving scenario for detection of objects present on road.

We will also discuss their architecture and describe how they perform the task of object detection and localization given the input image.

R-CNN

R-CNN was first introduced by Ross Girshick, Jeff Donahue, Trevor Darrell and Jitendra Malik in 2014. Region based Convolutional Network(R-CNN) combines two ideas: (1) one can apply high-capacity Convolutional Networks (CNN'S) to bottom-up region proposals in order to localize and segment objects and (2) When labelling data is scarce, supervised pre-training for an auxiliary task, followed by domain-specific- fine-tuning, boosts performance significantly. They combined region proposals with CNNs and called the resulting model an R-CNN or Region-based Convolutional Network.

R-CNN Architecture:

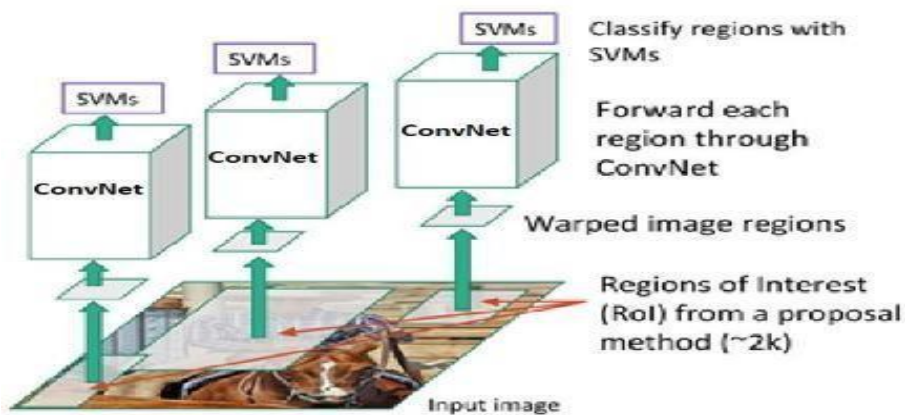


Fig:4.2(i) R-CNN Architecture

In R-CNN we input an image and then using the basic feature detection techniques such as edge detection etc. we get Region Proposals Also referred as Regions of interest. This process is also known as selective search.

In the next step after these region proposals are extracted these wrapped image regions go through a trained CNN in our implementation Alex Net and then on the final layer the classification is done Using a Support Vector Machine (SVM) classifier which classifies whether this is an object and if so what. Pictorial representation of above process is shown in Figure.

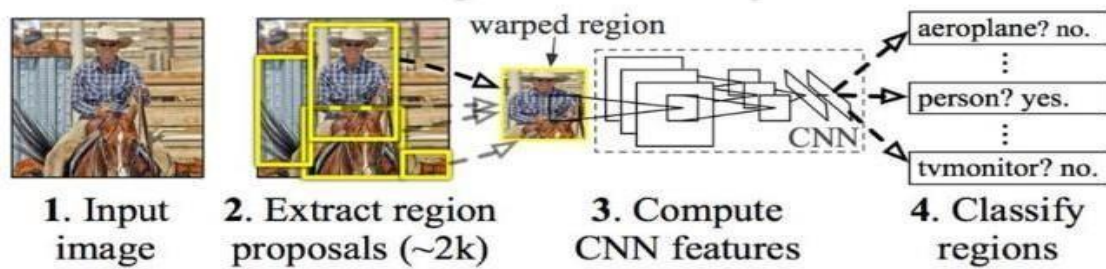


Fig:4.2(ii)R-CNN

Convolutional Neural Network(CNN):

Introduction:

Convolutional neural networks (CNN) sounds like a weird combination of biology and math with a little CS sprinkled in, but these networks have been some of the most influential innovations in the field of computer vision. 2012 was the first year that neural nets grew to prominence as Alex Krizhevsky used them to win that year's ImageNet competition (basically, the annual Olympics of computer vision), dropping the classification error record from 26% to 15%, an astounding improvement at the time. Ever since then, a host of companies have been using deep learning at the core of their services. Facebook uses neural nets for their automatic tagging algorithms, Google for their photo search, Amazon for their product recommendations, Pinterest for their home feed personalization, and Instagram for their search infrastructure. However, the classic, and arguably most popular, use case of these networks is for image processing. Within image processing, let's take a look at how to use these CNNs for image classification.

CHAPTER 5
MODULE DESCRIPTION

MODULE DESCRIPTION

5.1. PROBLEM DEFINITION

Now the user can store the files with effective security mechanisms and retrieve files from the cloud server in a combine key process with updated key and private key.

To access this project concepts while the large number of cloud service provider in a better manner. File security mechanisms are used to save the files in a effective notations. The cloud user allow to retrieve data from the different cloud service provider.

5.2. OVERVIEW OF THE PROJECT

This system fully focuses on the encryption with outsourced revocation. User can get the service from the service provider after that can upload the files to the corresponding cloud server. PKG (Private Key Generator) is the process to generate private key to the user and cloud server. Cloud server having KU-CSP (Key Update Cloud Service Provider). PKG to send the outsource key to the CSP.

CSP can provide the updated key to the user. When the file revocation process the private key and updated key to be combined and verify to the user after that the file can be downloaded from the cloud server. File revocation is the process to outsource the data from one server to another server.

When the revocation process the revocation request can be send to the server and after that the private key and updated key combine with matching and revocation the file, using machine learning.

After the completion of file revocation it can be downloaded from the corresponding server. KU-CSP can update the key for the outsource the date to the user to cloud service provider. Before store the files into the server it can be verify, encrypt and re-encrypt the file after that stored into cloud server.

5.3. MODULES

5.3.1. Object Detection

The Watch module also records the opposite side of the user to detect the IoT devices installed in the room and recognizes their types. For object detection and recognition, recent deep learning approaches have shown excellent performance in terms of accuracy. The Deep Learning framework divides each input image into $S \times S$ grids and each grid predicts N bounding boxes and their confidence scores which indicate whether the bounding box contains an object or not.

5.3.2. Gaze Estimation

We use the orientation of the face (i.e., pitch, yaw, and roll) and the distance between the Watch module and the user as head pose information for training a gaze estimator. A deep learning-based approach called Deep Gaze was employed to compute the value of face orientation. The Deep Gaze uses a Region based- convolutional neural network (R-CNN) to output the values of face orientation from the given face image.

5.3.3. Hand gesture recognition

Gesture recognition is performed by a commercial gesture sensor module. This sensor supports the following nine hand gestures: “Up”, “Down”, “Left”, “Right”, “Forward”, “Backward”, “Clockwise”, “Counter clockwise”, and “Wave”. User’s hand gestures are associated to different actions according to the type of the selected IoT device.

5.4 ARCHITECTURE DIAGRAM

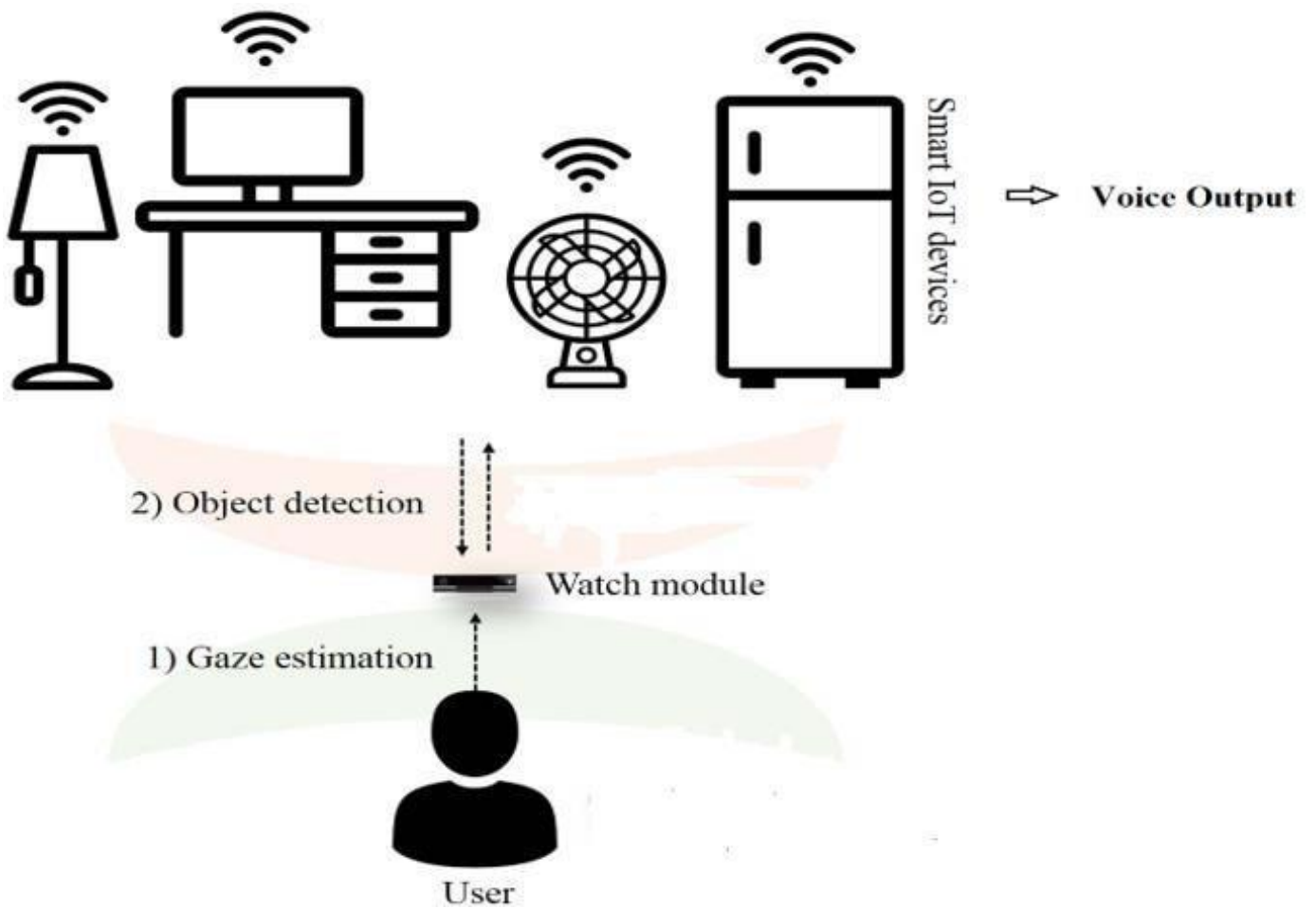


Fig:5.4 Architecture diagram

5.5 UML DIAGRAM

5.5.1 SEQUENCE DIAGRAM

A Sequence diagram is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of Message Sequence diagrams are sometimes called event diagrams, event sceneries and timing diagram.

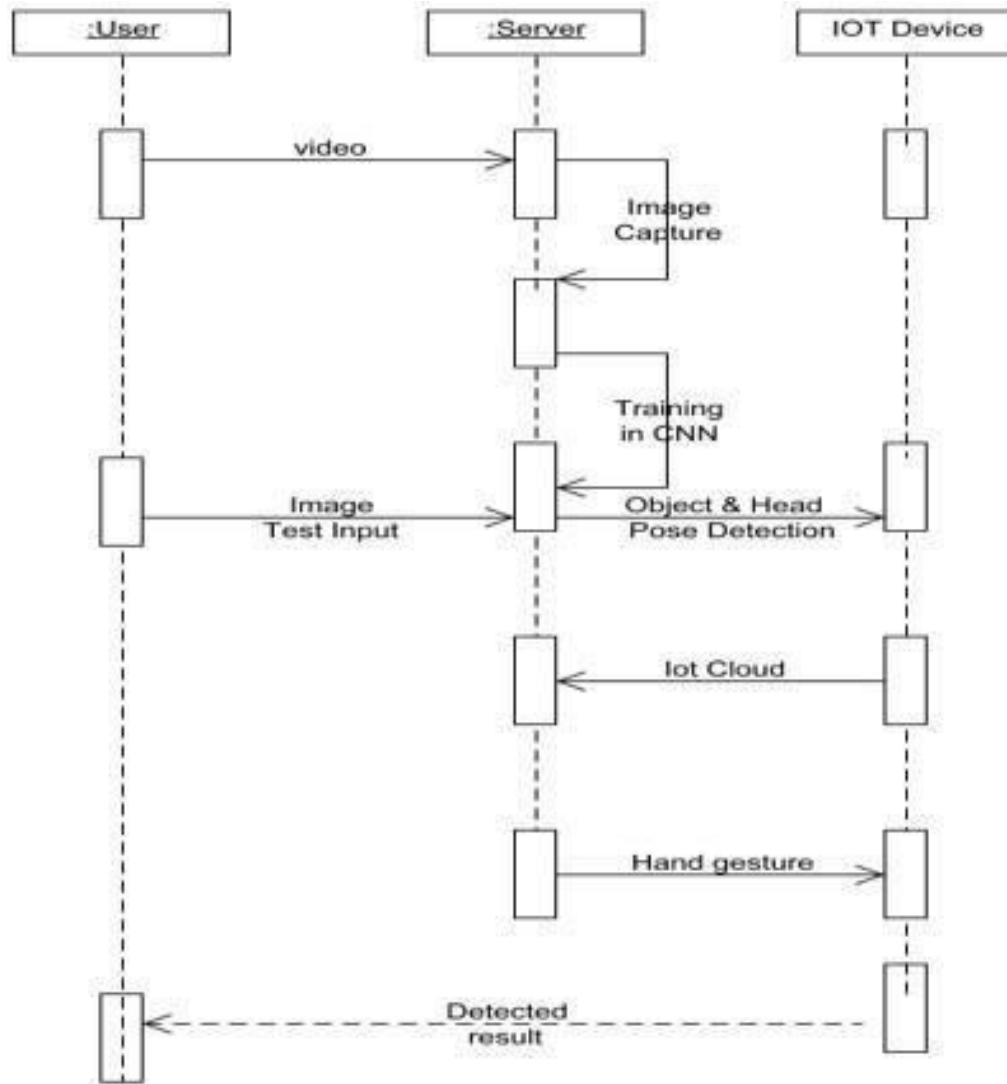


Fig:5.5 Sequence diagram

5.5.2 ACTIVITY DIAGRAM

Activity diagram is a graphical representation of workflows of stepwise activities and actions with support for choice, iteration and concurrency. An activity diagram shows the overall flow of control.

The most important shape types:

- Rounded rectangles represent activities.
- Diamonds represent decisions.

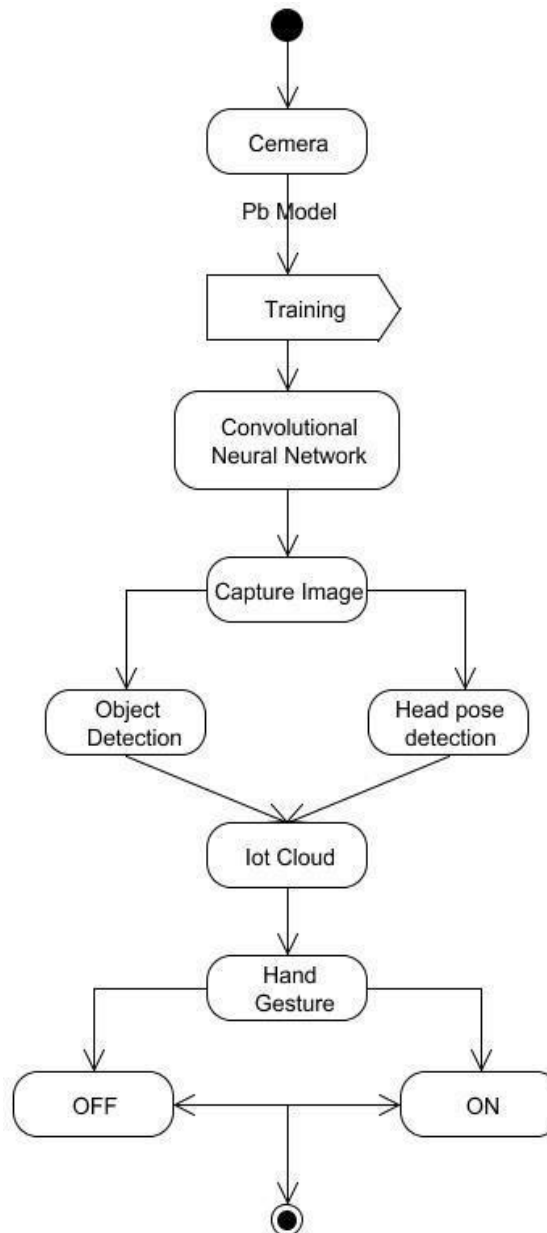


Fig:5.6 Activity diagram

5.5.3 COLLABORATION DIAGRAM

UML Collaboration Diagrams illustrate the relationship and interaction between software objects. They require use cases, system operation contracts and domain model to already exist. The collaboration diagram illustrates messages being sent between classes and objects.

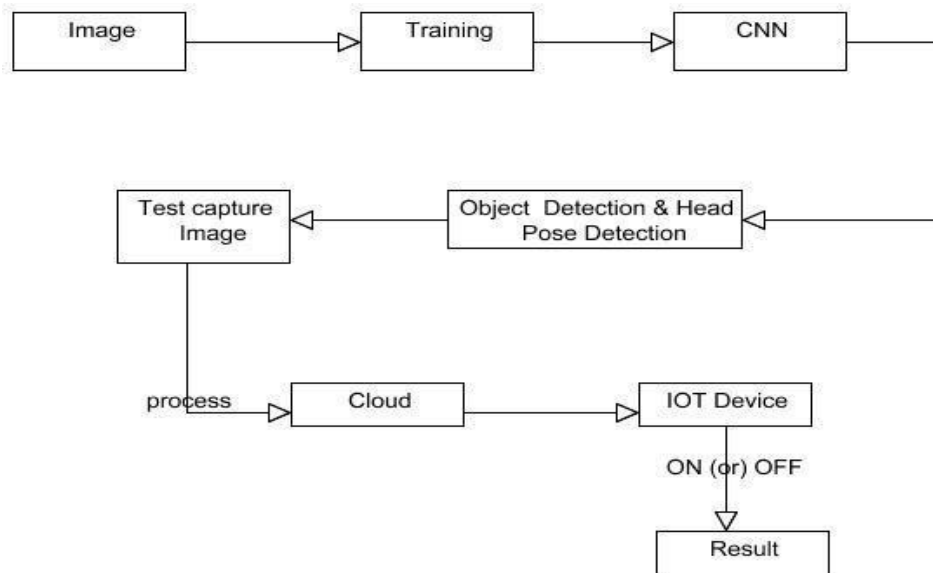


Fig:5.7 Collaboration diagram

5.5.4 USE CASE DIAGRAM

A Use case Diagram is used to present a graphical overview of the functionality provided by a system in terms of actors, their goals and any dependencies between those use cases.

Use case diagram consists of two parts:

Use case: A use case describes a sequence of actions that provided something of measurable value to an actor and is drawn as a horizontal ellipse.

Actor: An actor is a person, organization or external system that plays a role in one or more interaction with the system.

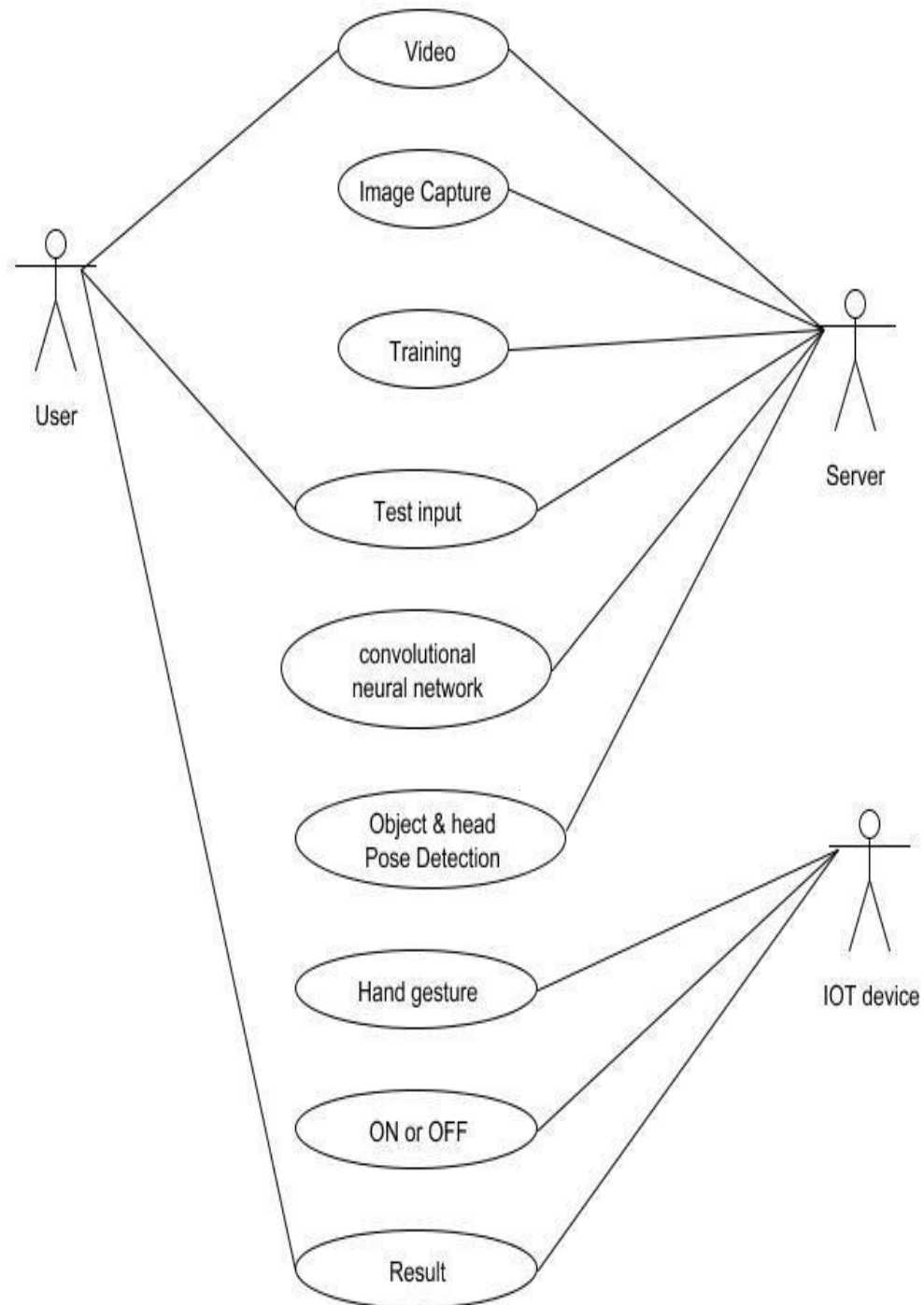


Fig:5.8 Use case diagram

5.6. DATABASE DESIGN

A database is a collection of interrelated data stored with minimum redundancy to serve many users quickly and efficiently. The general Objectives of the database design are to make the data access easy, inexpensive and flexible to the user. The data in the system has to be stored and retrieved from database.

Designing the database is the part of system design. Data elements and data structures to be stored have been identified at analysis stage and are structured and put together to design the data storage and retrieval system.

5.7. INPUT DESIGN

The input of a system can be defined as the information that is provided to the system. This is used for future processing by the system to obtain meaningful information, which helps in decision-making. Input design is the process of converting user-oriented inputs to a computer-based format.

Input is a part of overall system design, which requires special attention. Inaccurate input data are the most common cause of errors in error processing. Input design can control errors entered by users. Entered data have to be checked for their accuracy and direction of errors. Appropriate error message have to be displayed. When an invalid data is entered, the user should not be allowed to type that data.

5.8. OUTPUT DESIGN

The computer output is the most important and direct source of information to the user. Efficient and intelligible output design improves the system's relationship with the user and helps in decision making.

Output design was studied going actively during the study phase. The objective of the output design is defined the contents and format of all documents and reports in an attractive and useful format.

CHAPTER 6
SYSTEM TESTING

SYSTEM TESTING

6.1. TESTING OBJECTIVES

Testing is the process of detecting errors. Testing plays a critical role in assuring quality and ensuring the reliability of software. The results of testing are used later on during maintenance also.

- 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 error, if it exists
- The software more or less confirms to the quality and reliable.

6.2. TESTING LEVELS

System testing is stage of implementation which is aimed at ensuring that the system works accurately and efficient before live operation commences. Testing is vital the success of the system. System testing makes a logical assumption that if all the parts of the system are correct, the goal will be successfully achieved.

6.2.1. Unit Testing

Unit testing is conducted to verify the functional performance of each modular component of the software. Unit testing focuses on the smallest unit of the software design (i.e.), the module. The white-box testing techniques were heavily employed for unit testing.

6.2.2. Functional Testing

Functional test cases involved exercising the code with nominal input values for which the expected results are known, as well as boundary values and special values, such as logically related inputs, files of identical elements, and empty files.

6.2.3. Performance Testing

It determines the amount of execution time spent in various parts of the unit, program throughput, and response time and device utilization by the program unit.

6.2.4. Stress Testing

Stress Test is those test designed to intentionally break the unit. A Great deal can be learned about the strength and limitations of a program by examining the manner in which a programmer in which a program unit breaks.

6.2.5 Structured Testing

Structure Tests are concerned with exercising the internal logic of a program and traversing particular execution paths. The way in which White-Box test strategy was employed to ensure that the test cases could Guarantee that all independent paths within a module have been have been exercised at least once.

- Exercise all logical decisions on their true or false sides.
- Execute all loops at their boundaries and within their operational bounds.
- Exercise internal data structures to assure their validity.
- Checking attributes for their correctness.

6.2.6 Integration Testing

Integration testing is a systematic technique for construction the program structure while at the same time conducting tests to uncover errors associated with interfacing. i.e., integration testing is the complete testing of the set of modules which makes up the product. The objective is to take untested modules and build a program structure tester should identify critical modules. Critical modules should be tested as early as possible. One approach is to wait until all the units have passed testing, and then combine them and then tested. This approach is evolved from unstructured testing of small programs. Another strategy is to construct the product in increments of tested units. A small set of modules are integrated together and tested, to which another module is added and tested in combination. And so on. The advantages of this approach are that, interface dispenses can be easily found and corrected.

The major error that was faced during the project is linking error. When all the modules are combined the link is not set properly with all support files. Then we checked out for interconnection and the links. Errors are localized to the new module

and its intercommunications. The product development can be staged, and modules integrated in as they complete unit testing. Testing is completed when the last module is integrated and tested.

6.2.5. Experimented Outcome

EXISTING SYSTEM	PROPOSED SYSTEM
The target device is identified by the YOLO Framework with CNN algorithm.	In proposed system used R-CNN algorithm .It has high accuracy .
YOLO framework divides each input image into $S \times S$ grids and each grid predicts N bounding boxes and their confidence scores which indicate whether the bounding box contains an object or not.	R-CNN algorithm has capable of accepting bulk images as input and estimate the object. R-CNN algorithm is faster than the CNN algorithm.
CNN not accepting the bulk image to estimate object .CNN is very low accuracy.	the final output of this system is to delivered as voice message. It consists information like type of object and name of the object in front of the user. User can control the objects using hand gesture.

Table 6.1. Experimented outcome

CHAPTER 7
SYSTEM IMPLEMENTATION

SYSTEM IMPLEMENTATION

7.1. IMPLEMENTATION PROCEDURE

Implementation is the stage of the project when the theoretical design is turned out into a working system. Each program is tested individually at the time of development using the data and has verified that this program linked together in the way specified in the programs specification, the computer system and its environment is tested to the satisfaction of the user. The final stage is to document the entire system which provides components and the operating procedures of the system.

7.2. ALGORITHM

7.2.1. Regional Convolution Neural Network Algorithm

In this chapter we will talk about the Region based methods for object detection. In particular, R-CNN (Regional CNN), the original application of CNNs to this problem, along with its descendants Fast R-CNN, and Faster R-CNN. In classification, there's generally an image with a single object as the focus and the task is to say what that image is. But when we look at the world around us, we carry out far more complex tasks. We see complicated sights with multiple overlapping objects, and different backgrounds and we not only classify these different objects but also identify their boundaries, differences, and relations to one another. The goal of a Region based CNN is to take an image input to detected and localize the object in the image. All three above mentioned architectures were used for object detection and localization for a driving scenario to detect cars, pedestrians and traffic signs etc. and performed a time study to conclude which of the three architectures can be used as a detector for a real- world driving scenario for detection of objects present on road. We will also discuss their architecture and describe how they perform the task of object detection and localization given the input image.

7.2.2 Convolution Neural Network:

Convolutional neural networks (CNN) sounds like a weird combination of biology and

math with a little CS sprinkled in, but these networks have been some of the most influential innovations in the field of computer vision. 2012 was the first year that neural nets grew to prominence as Alex Krizhevsky used them to win that year's ImageNet competition (basically, the annual Olympics of computer vision), dropping the classification error record from 26% to 15%, an astounding improvement at the time. Ever since then, a host of companies have been using deep learning at the core of their services. Facebook uses neural nets for their automatic tagging algorithms, Google for their photo search, Amazon for their product recommendations, Pinterest for their home feed personalization, and Instagram for their search infrastructure. However, the classic, and arguably most popular, use case of these networks is for image processing. Within image processing, let's take a look at how to use these CNNs for image classification.

CHAPTER 8
CONCLUSION AND ENHANCEMENTS

CONCLUSION AND FUTURE ENHANCEMENTS

8.1. CONCLUSION

Finally, we have proposed a novel methodology to distinguish images for a specified social network, allowing a distinction among the processing suffered by the images when uploaded on a specific social network (Facebook, Instagram).

The definition of this technique based on such features which by resorting at trained classifiers, which actually is able to identify the social platform of provenance and also to detect the quality factor before uploading.

A personalized recommendation approach was proposed by combining social network factors: personal interest, interpersonal interest similarity, and interpersonal influence. We conducted extensive experiments on two large real-world social rating datasets, and showed significant improvements over existing approaches that use mixed social network information.

8.2. FUTURE ENHANCEMENTS

The definition of a technique based on such features which by resorting at trained classifiers is able to identify the social platform of provenance and also to detect the quality factor before uploading.

This module is expected to be patented and legal licensing would be done to protect the author and coder copyrights and will be deployed on a common platform via a legal quisel for Facebook and Instagram.

Furthermore, the research will again resume in uploading and saving of videos in the web stream and protecting videos against duplication from the social network.

APPENDICES

APPENDICS-I

OBJECT DETECTION:

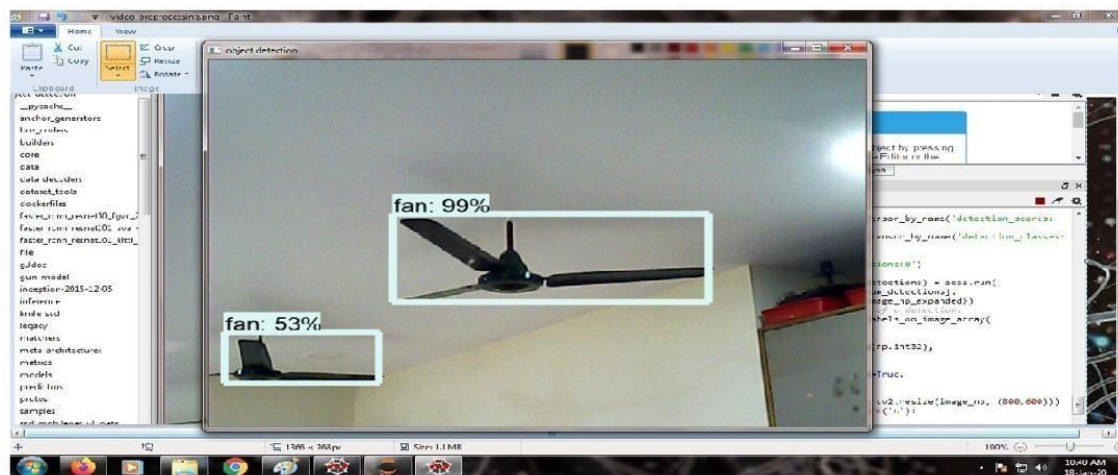


Fig a(i):Object detection

Fig a(ii): Object detection

GAZE ESTIMATION:

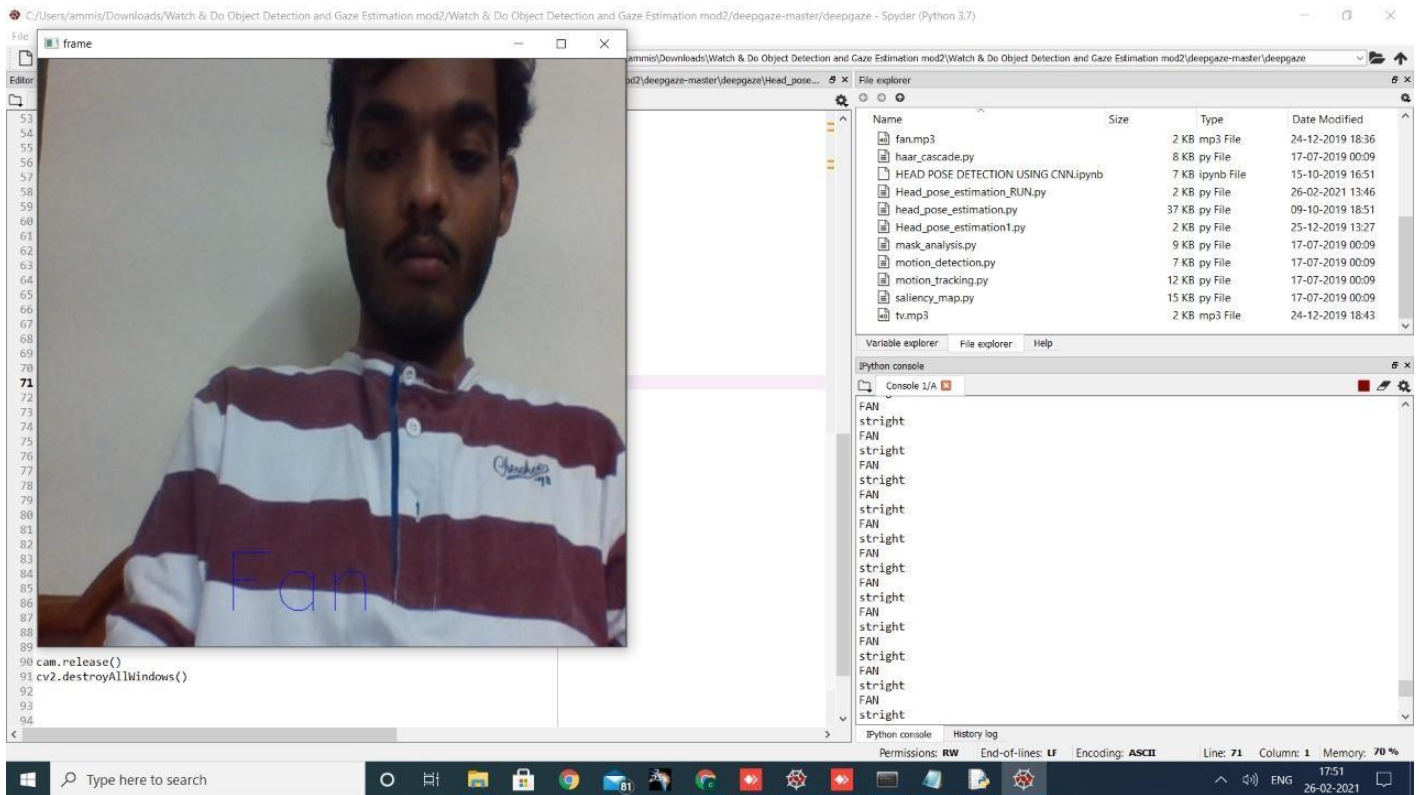


Fig: b(i) Gaze estimation

IMPLEMENTATION CODE

Head_pose_run.py

```
import os
import datetime
import time
import tensorflow as tf
import cv2
#from gtts import gTTS
from playsound import playsound
from deepgaze.head_pose_estimation import CnnHeadPoseEstimator
#from firebase import firebase
#fixfixed_interval = 3
#firebase = firebase.FirebaseApplication('https://watch-and-do.firebaseio.com',
None)
sess = tf.Session()
my_head_pose_estimator = CnnHeadPoseEstimator(sess)
cam = cv2.VideoCapture(0)
try:
    if not os.path.exists('data'):
        os.makedirs('data')
except OSError:
    print ('Error: Creating directory of data')
my_head_pose_estimator.load_yaw_variables(os.path.realpath("E:/Watch & Do
Object Detection and Gaze Estimation mod2/deepgaze-
master/etc/tensorflow/head_pose/yaw/cnn_cccdd_30k.tf"))
last_command=""
```

```

cur_cmd=""
currentframe = 0
while (True):
    ret,frame = cam.read()
    frame = cv2.resize(frame,(640,640))

    if cv2.waitKey(1) & 0xFF == ord('q'):
        break
    ss = my_head_pose_estimator.return_yaw(frame)
    k=ss[0][0][0]
    if k>40 and k<100:
        print('right') print('TV')
        playsound("tv.mp3")
        status="TV"
        data={"Status":status}
        # firebase.put(", 'WATCH AND DO/IOT device', data)
        cur_cmd="TV"
    elif k<(-40)and k>(-100):
        print('left')
        print('Refrigerator')
        playsound("bulb.mp3")
        status="bulb"
        data={"Status":status}
        #firebase.put(", 'WATCH AND DO/IOT device', data)
        cur_cmd="bulb"
    elif k>(-40) and k<(40):
        print('stright')
        print('FAN')

```

```

status="fan"
data={"Status":status}
#firebase.put("", 'WATCH AND DO/IOT device', data)
playsound("fan.mp3")
cur_cmd="Fan"
cv2.putText(frame,cur_cmd, (200,600), cv2.FONT_HERSHEY_SIMPLEX,
1,(0,0,255),6)
cv2.imshow('frame',frame)
cam.release()
cv2.destroyAllWindows()

```

Head_pose_detection.py

```

#!/usr/bin/env python

#The MIT License (MIT)
#Copyright (c) 2016 Massimiliano Patacchiola #
#THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND,
EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF
#MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND
NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT
HOLDERS BE LIABLE FOR ANY
#CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF
CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
CONNECTION WITH THE
#SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

import numpy as np
import tensorflow as tf

```

```

import cv2
import os.path
import imp #to check for missing modules
import math

#Check if dlib is installed
try:
    imp.find_module('dlib')
    IS_DLIB_INSTALLED = True
    import dlib
    print('[DEEPGAZE] head_pose_estimation.py: the dlib library is installed.')
except ImportError:
    IS_DLIB_INSTALLED = False
    print('[DEEPGAZE] head_pose_estimation.py: the dlib library is not installed.')

#Enable if you need printing utilities
DEBUG = False
class CnnHeadPoseEstimator:
    """ Head pose estimation class which uses convolutional neural network

    It finds Roll, Pitch and Yaw of the head given an head figure as input.
    It manages input (colour) picture larger than 64x64 pixels. The CNN are
robust
    to variance in the input features and can handle occlusions and bad
    lighting conditions. The output values are in the ranges (degrees):
    ROLL=[-40, +40]
    YAW=[-100, +100]

```

```

def __init__(self, tf_session):
    """ Init the class

    @param tf_session An external tensorflow session
    """

    self._sess = tf_session

def print_allocated_variables(self):
    """ Print all the Tensorflow allocated variables

    """

    all_vars = tf.all_variables()

    print("[DEEPGAZE] Printing all the Allocated Tensorflow Variables:")
    for k in all_vars:
        print(k.name)

def _allocate_yaw_variables(self):
    """ Allocate variables in memory (for internal use)

    The variables must be allocated in memory before loading
    the pretrained weights. In this phase empty placeholders are
    defined and later fill with the real values.
    """

    self._num_labels = 1

    # Input data [batch_size, image_size, image_size, channels]
    self.tf_yaw_input_vector = tf.placeholder(tf.float32, shape=(64, 64, 3)) #
    Variables.

```

```

#Conv layer
#[patch_size, patch_size, num_channels, depth] self.hy_conv1_weights =
tf.Variable(tf.truncated_normal([3, 3, 3, 64],
stddev=0.1))
self.hy_conv1_biases = tf.Variable(tf.zeros([64]))
#Conv layer
#[patch_size, patch_size, depth, depth]
self.hy_conv2_weights = tf.Variable(tf.truncated_normal([3, 3, 64, 128],
stddev=0.1))
self.hy_conv2_biases = tf.Variable(tf.random_normal(shape=[128]))
#Conv layer
#[patch_size, patch_size, depth, depth]
self.hy_conv3_weights = tf.Variable(tf.truncated_normal([3, 3, 128, 256],
stddev=0.1)) #was[3, 3, 128, 256]
self.hy_conv3_biases = tf.Variable(tf.random_normal(shape=[256]))

#Dense layer
#[ 5*5 * previous_layer_out , num_hidden] wd1
#here 5*5 is the size of the image after pool reduction (divide by half 3 times)
self.hy_dense1_weights = tf.Variable(tf.truncated_normal([8 * 8 * 256, 256],
stddev=0.1)) #was [5*5*256, 1024]
self.hy_dense1_biases = tf.Variable(tf.random_normal(shape=[256]))
#Dense layer
#[ , num_hidden] wd2
#self.hy_dense2_weights = tf.Variable(tf.truncated_normal([256, 256],
stddev=0.01))
#self.hy_dense2_biases = tf.Variable(tf.random_normal(shape=[256]))
#Output layer
self.hy_out_weights = tf.Variable(tf.truncated_normal([256,

```

```

self._num_labels], stddev=0.1))
    self.hy_out_biases =
tf.Variable(tf.random_normal(shape=[self._num_labels]))

    # dropout (keep probability)
    #self.keep_prob = tf.placeholder(tf.float32, name="keep_prob")

    # Model.
    def model(data):

        X = tf.reshape(data, shape=[-1, 64, 64, 3])
        if(DEBUG == True): print("SHAPE X: " + str(X.get_shape()))

        # Convolution Layer 1
        conv1 = tf.tanh(tf.nn.bias_add(tf.nn.conv2d(X, self.hy_conv1_weights,
strides=[1, 1, 1, 1], padding='SAME'),self.hy_conv1_biases))
        if(DEBUG == True): print("SHAPE conv1: " + str(conv1.get_shape())) #
        Max Pooling (down-sampling)
        pool1 = tf.nn.max_pool(conv1, ksize=[1, 2, 2, 1], strides=[1, 2, 2, 1],
padding='SAME')
        if(DEBUG == True): print("SHAPE pool1: " + str(pool1.get_shape())) #
        Apply Normalization
        norm1 = tf.nn.lrn(pool1, 4, bias=1.0, alpha=0.001 / 9.0, beta=0.75) #
        Apply Dropout
        #norm1 = tf.nn.dropout(norm1, _dropout)

        # Convolution Layer 2
        conv2 = tf.tanh(tf.nn.bias_add(tf.nn.conv2d(norm1, self.hy_conv2_weights,
strides=[1, 1, 1, 1], padding='SAME'),self.hy_conv2_biases))

```

```

if(DEBUG == True): print("SHAPE conv2: " + str(conv2.get_shape())) #
Max Pooling (down-sampling)
pool2 = tf.nn.max_pool(conv2, ksize=[1, 2, 2, 1], strides=[1, 2, 2, 1],
padding='SAME')
if(DEBUG == True): print("SHAPE pool2: " + str(pool2.get_shape())) #
Apply Normalization
norm2 = tf.nn.lrn(pool2, 4, bias=1.0, alpha=0.001 / 9.0, beta=0.75) #
Apply Dropout
#norm2 = tf.nn.dropout(norm2, _dropout)

# Convolution Layer 3
conv3 = tf.tanh(tf.nn.bias_add(tf.nn.conv2d(norm2, self.hy_conv3_weights,
strides=[1, 1, 1, 1], padding='SAME'),self.hy_conv3_biases))
if(DEBUG == True): print("SHAPE conv3: " + str(conv3.get_shape())) #
Max Pooling (down-sampling)
pool3 = tf.nn.max_pool(conv3, ksize=[1, 2, 2, 1], strides=[1, 2, 2, 1],
padding='SAME')
if(DEBUG == True): print("SHAPE pool3: " + str(pool3.get_shape())) #
Apply Normalization
norm3 = tf.nn.lrn(pool3, 4, bias=1.0, alpha=0.001 / 9.0, beta=0.75)

# Fully connected layer 4 dense1
= tf.reshape(norm3, [-1,
self.hy_dense1_weights.get_shape().as_list()[0]]) # Reshape conv3
if(DEBUG == True): print("SHAPE dense1: " + str(dense1.get_shape()))
dense1 = tf.tanh(tf.matmul(dense1, self.hy_dense1_weights) +
self.hy_dense1_biases)

```

```

def load_roll_variables(self, rollFilePath): """
    Load variables from a tensorflow file

    It must be called after the variable allocation.

    This function take the variables stored in a local file
    and assign them to pre-allocated variables.

    @param rollFilePath Path to a valid checkpoint """

    #Allocate the variables in memory
    self._allocate_roll_variables()

    #It is possible to use the checkpoint file
    #y_ckpt = tf.train.get_checkpoint_state(rollFilePath)
    #.restore(self._sess, y_ckpt.model_checkpoint_path)

    #For future use, allocating a fraction of the GPU
    #gpu_options = tf.GPUOptions(per_process_gpu_memory_fraction=0.5)
    #Allocate only half of the GPU memory

    if(os.path.isfile(rollFilePath)==False): raise ValueError('[DEEPGAZE]
CnnHeadPoseEstimator(load_roll_variables): the roll file path is incorrect.')

    tf.train.Saver(((("conv1_roll_w": self.hr_conv1_weights, "conv1_roll_b":
self.hr_conv1_biases,
                    "conv2_roll_w": self.hr_conv2_weights, "conv2_roll_b":
self.hr_conv2_biases,
                    "conv3_roll_w": self.hr_conv3_weights, "conv3_roll_b":
self.hr_conv3_biases,

```

```

"dense1_roll_w": self.hr_dense1_weights, "dense1_roll_b": P3D_LEFT_EYE,
        P3D_STOMION])

#Return the 2D position of our landmarks
img_h, img_w, img_d = image.shape
landmarks_2D = self._return_landmarks(inputImg=image, roiX=0, roiY=img_w,
roiW=img_w, roiH=img_h, points_to_return=TRACKED_POINTS)

#Print som red dots on the image
#for point in landmarks_2D:
    #cv2.circle(frame,( point[0], point[1] ), 2, (0,0,255), -1)

#Applying the PnP solver to find the 3D pose
#of the head from the 2D position of the
#landmarks.

#retval - bool

#rvec - Output rotation vector that, together with tvec, brings
#points from the world coordinate system to the camera coordinate system.
#tvec - Output translation vector. It is the position of the world origin
(SELLION) in camera co-ords

retval, rvec, tvec = cv2.solvePnP(landmarks_3D,
        landmarks_2D,
        self.camera_matrix,
        self.camera_distortion)

#Get as input the rotational vector
#Return a rotational matrix
rmat, _ = cv2.Rodrigues(rvec)
#euler_angles contain (pitch, yaw, roll)
#euler_angles = cv.DecomposeProjectionMatrix(projMatrix=rmat,
cameraMatrix=camera_matrix, rotMatrix, transVect, rotMatrX=None,
rotMatrY=None, rotMatrZ=None)

```

```

head_pose = [ rmat[0,0], rmat[0,1], rmat[0,2], tvec[0],
               rmat[1,0], rmat[1,1], rmat[1,2], tvec[1],
               rmat[2,0], rmat[2,1], rmat[2,2], tvec[2],
               0.0,    0.0,    0.0,    1.0 ]
#print(head_pose) #TODO remove this line
return self.rotationMatrixToEulerAngles(rmat)
# Calculates rotation matrix to euler angles
# The result is the same as MATLAB except the order #
of the euler angles ( x and z are swapped ).
def rotationMatrixToEulerAngles(self, R) :

    #assert(isRotationMatrix(R))

    #To prevent the Gimbal Lock it is possible to use #a
    threshold of 1e-6 for discrimination
    sy = math.sqrt(R[0,0] * R[0,0] + R[1,0] * R[1,0])
    singular = sy < 1e-6
    if not singular :
        x = math.atan2(R[2,1] , R[2,2])
        y = math.atan2(-R[2,0], sy)
        z = math.atan2(R[1,0], R[0,0])
    else :
        x = math.atan2(-R[1,2], R[1,1])
        y = math.atan2(-R[2,0], sy)
        z = 0
    return np.array([x, y, z])

```

REFERENCES

- [1] C. Perera , A. Zaslavsky , M. Compton, P. Christen, and D. Georgakopoulos , “Semantic-driven configuration of Internet of Things middleware,” in *Proc. SKG*, Beijing, China, 2013.
- [2] Charith Perera, Arkady Zaslavsky, “Context-aware computing, learning, and big data in Internet of Things: A survey,” 2013.
- [3] Gururaj Kulkarni, S. H Manoor , “Enabling Technologies, Protocols, and Applications: A Detailed Survey on IOT”, 2017.
- [4] H. B. Pandya and T. A. Champaneria, “Internet of Things: Survey and case studies”, 2015.
- [5] J. B. B. Neto, T. H. Silva, R. M. Assunç ao, R. A. F. Mini, and A. A. F. Loureiro, “Sensing in the collaborative Internet of Things”, Mar. 2015.
- [6] K. Wu *et al.*, “HJam: Attachment transmission in WLANs”, 2012.
- [7] K. Wu *et al.*, “CSI-based indoor localization”, Jul. 2013.
- [8] Luigi Atzori , Antonio Iera, “The Internet of Things: A survey”, 2010.
- [9] Nataniel Ruiz Eunji Chong, “Fine-Grained Head Pose Estimation Without Keypoints”, 2017.
- [10] R. Mahmoud, T. Yousuf, F. Aloul, and I. Zualkernan, “Internet of Things(IoT) security: Current status, challenges and prospective measures,” in *Proc. ICITST*, London, U.K., 2015.
- [11] R. Vignesh and A. Samyadurai , “Security on Internet of Things (IOT) with Challenges and Countermeasures”, 2017.
- [12] W. Kluge *et al.*, “A fully integrated 2.4-GHz IEEE 802.15.4-compliant transceiver for ZigBeeTM applications”, Dec. 2006.
- [13] Xucong Zhang, Yusuke Sugano, “Appearance-Based Gaze Estimation in the Wild”, 2015.

- [14] Y. Zou *et al.*, “GRfid: A device-free RFID-based gesture recognition system,” *IEEE Trans. Mobile Comput.*, vol. 16, no. 2, pp. 381–393, Feb. 2017.
- [15] Y. Zou, W. Liu, K. Wu, and L. M. Ni, “Wi-Fi radar: Recognizing human behavior with commodity Wi-Fi,” *IEEE Commun. Mag.*, Oct. 2017.
- [16] Zhiang Hu, “Real-Time Head Pose Estimation with Convolutional Neural Networks”, 2015.