

GESTURE BASED SECURED AUTO APPLIANCE CONTROL SYSTEM FOR DISABLED PEOPLE WITHOUT INTERNET

Submitted in partial fulfillment of the requirements for the award of

Bachelor of Technology degree in

Biomedical Engineering.

By

AJIN R (37240003)

SIVA SANKAR K (37240068)



**DEPARTMENT OF BIOMEDICAL ENGINEERING
SCHOOL OF BIO AND CHEMICAL ENGINEERING**

SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY

(DEEMED TO BE UNIVERSITY)

Accredited with Grade “A” by NAAC | 12B Status by UGC | Approved by AICTE

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 | 12B Status by UGC | Approved by AICTE

Jeppiaar Nagar, Rajiv Gandhi Salai, Chennai – 600 119

www.sathyabama.ac.in

DEPARTMENT OF BIOMEDICAL ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of **Ajin R (37240003)**, **Siva Sankar K (37240068)** who have done the project work as a team who carried out the project entitled “**Gesture Based Secured Auto Appliance Control System for Disabled People Without Internet**” under my supervision from **October 2020** to **March 2021**

Dr. T. SUDHAKAR, M.Sc., Ph.D.,

Internal Guide

Dr. T. SUDHAKAR, M.Sc., Ph.D.,

Head of the Department

Submitted for Viva voce Examination held on _____

Internal Examiner

External Examiner

DECLARATION

We, **Ajin R (37240003)** and **Siva Sankar K (37240068)** hereby declare that the Project Report entitled “**Gesture Based Secured Auto Appliance Control System for Disabled People Without Internet**” done by us under the guidance of **Dr.T.Sudhakar**, Department of Biomedical Engineering SIST is submitted in partial fulfillment of the requirements for the award of Bachelor of Technology degree in Biomedical Engineering.

DATE:

PLACE:

1.

2.

SIGNATURE OF THE CANDIDATES

ACKNOWLEDGEMENT

We are pleased to acknowledge our sincere thanks to the **Board of Management of SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. We are grateful to them.

We convey our thanks to **Dr.J.Prem Kumar** M.Sc., Ph.D., **T. Sudhakar** M.Sc., Ph.D., **Head of the Department, Dept. of Biomedical Engineering** for providing me necessary support and details at the right time during the progressive reviews.

We would like to express our sincere and deep sense of gratitude to our Project Guide **Dr. T. Sudhakar**, M.Sc., Ph.D., **Department of Bio Medical Engineering** for his valuable guidance, suggestions and constant encouragement paving the way for the successful completion of our project work.

We wish to express our thanks to all **Teaching and Non-teaching** staff members of the **Department of Biomedical Engineering** who were helpful in many ways for the completion of the project.

ABSTRACT

Automation is that the essential need for this world. There are various kinds of automations like building automation, industrial automation, Home automation, AI, etc. will be a possible application which provides support to any or all the persons. Home automation is that the use and control of home appliances remotely or automatically. Day by day the gap between humans and machines is being reduced. In present days hand gesture-based home automation is getting more importance. Gesture recognition refers to recognizing the motion of the human parts like hand, face, etc. Most of the electronic components manufacturers focus on the hand gesture basis. during this proposed work we'll be using AI which recognizes human faces and if the registered human is successfully detected then it recognizes the human gestures without the employment of any reasonably sensors with the assistance of a camera and acts accordingly to regulate any reasonably appliance. Thus, our project aims at discovering an automation system without the utilization of sensors or a mobile app with the assistance of AI.

TABLE OF CONTENTS

CHAPTER NO	TITLE	PAGE NO.
	ABSTRACT	v
	LIST OF SYMBOLS AND ABBREVIATION	ix
	LIST OF FIGURES	x
1	INTRODUCTION	1
	1.1 GENERAL	1
	1.2 TECHNOLOGIES USED	4
	1.2.1 Deep Learning	4
	1.2.1.1 Important of Deep Learning	6
	1.2.1.2 Deep Learning Process	6
	1.2.3 Classification Of Neural Networks	7
	1.2.4 Feed-Forward Neural Networks	9
	1.2.5 Recurrent Neural Networks (Rnns)	9
	1.2.6 Common uses of RNN	9
	1.2.7 Convolutional Neural Networks (Cnn)	10
2	REVIEW OF LITERATURE	12
	2.1 INTRODUCTION	12
	2.2 LITERATURE REVIEW	12
3	AIM AND SCOPE	22

	3.1 Aim	22
	3.2 EXISTING SYSTEM	22
	3.3 DISADVANTAGE OF EXISTING SYSTEM	22
	3.4 PROPOSED SYSTEM	23
	3.5 ADVANTAGE OF PROPOSED SYSTEM	23
	3.6 APPLICATIONS	23
4	MATERIALS AND METHODS	24
	4.1 ARCHITECTURAL DIAGRAM	24
	4.2 WORKING	25
	4.2.3 MODULE DESCRIPTION	26
	4.2.3.1 Data Collection Module	26
	4.2.3.2 Face detection module	27
	4.2.3.3 Face recognition module	29
	4.2.3.4 Hand segmentation	31
	4.2.3.5 Training gesture net	32
	4.2.3.6 Compare the gestures and switch the devices ON and OFF	33
	4.7 SOFTWARE USED	33
	4.7.1 VIRTUAL MACHINE	33
	4.7.2 PYCHARM CODE EDITOR	34
5	RESULTS AND DISCUSSION	36
	5.1 Final output obtained	36

6	CONCLUSION AND FUTURE WORK	42
	6.1 CONCLUSION	42
	6.2 SUMMARY	42
	REFERENCES	43
	APPENDIX	46

LIST OF SYMBOLS AND ABBREVIATIONS

CNN	Convolutional Neural Network
ILSVRC	ImageNet Large Scale Visual Recognition Challenge
RNN	Recurrent Neural Networks
HTS	Hand tracking and segmentation

LIST OF FIGURES

FIGURE NO	NAME OF THE FIGURE	PAGE NO.
1.1	Deep Learning Layers	5
1.2	Deep Learning Process	6
1.3	Types of Deep Learning Networks	8
1.4	CNN	10
1.5	Types of CNN	11
4.1	Face detection architecture	24
4.2	Gesture recognition	25
4.3	Ultralight face detector flow diagram	29
4.4	MobileFaceNet architercture	31
4.5	Hand Gesturenets diagram	33
5.1	Hand Gesture three	36
5.2	Hand Gesture one	36
5.3	Hand Gesture four	37
5.4	Running command in the terminal	38
5.5	Training process	38
5.6	Accuracy Generation	39
5.7	Pickle file generation	39
5.8	Automation of lights	40

CHAPTER 1

INTRODUCTION

1.1GENERAL

Smart homes utilize top most innovations such as advanced lighting elements and security systems to bring comfort and convenience to your living experience. As smart houses become more popular and also the technological advances are more widely used, owning a wise home has its advantages. Nevertheless, many factors still keep homeowners hesitant about purchasing a wise home.

Installing state-of-the-art features inside a home leads to the next tag for the property. Abhijith Punnappurath and his colleagues (2015) Existing methods for performing facial recognition in the presence of blur are built on the convolution model, which can't accommodate non-uniform blurring caused by hand-held camera tilts and rotations. The price of an intelligent home that creates our lives convenient is high because a number of the technology is comparatively new. In new smart homes, lightweight sensors and wireless cameras, a central touchscreen and automatic systems are just some advances that are standard during a new home. the price of living expenses like utilities, maintenance and repair of the technology are often expensive still. Qiang Nie and colleagues (2019) This paper introduces a view-invariant approach for human behaviour recognition that recovers corrupted skeletons based on a 3D bio-constrained skeleton model and visualises certain body-level motion features acquired during the recovery process with images to deal with noisy skeleton details and variations in viewpoints.

Owning a sensible home means having to find out a way to use your home. Unlike traditional homes, smart home technology requires you to adapt the innovations within your living area like security systems, air units and a foreign that controls your entire house. Ping Wei and friends (2019) The suggested approach is evaluated on three

complicated 3D action datasets: the MSR 3D Action Dataset, the Multiview 3D Case Dataset, and the UTKinect-Action 3D Dataset. Extensive preliminary findings on these datasets suggest that the proposed approach is efficient and beneficial. For the technology-savvy family, the smart home will help achieve convenience faster, except for others, it'll take reading manuals and learning how-to before the advantages of convenience pay off. Andrea Bandini et al (2020) In this survey, we review the literature that focuses on the hands using egocentric vision, categorising the existing approaches into: localization; interpretation; and application. Moreover, a list of the most prominent datasets with hand-based annotations is provided.

Video surveillance will be an exquisite tool in heightening security and deterring crime, but when the technology falls into the incorrect hands, problems with privacy can occur. Renliang Weng and colleagues (2015) We suggest a new partial face recognition method in which the probe partial face is aligned to gallery faces using the geometrical and textural details derived from the extracted local features. Security sensors within the doors and walls of a sensible home use wireless technology to transfer signals to a central control unit that notifies emergency officials of any foreign activity. Video surveillance feeds also are transferred wirelessly to a component of the house that may be monitored. If the security feeds from the video and sensor falls into the wrong hands, your smart home could be monitored by those who have managed to hack it.

Energy Savings

Home automation systems have definitely proven themselves within the arena of energy efficiency. Automated thermostats allow you to pre-program temperatures supported the time of day and also the day of the week. And a few even comply with your behaviors, learning and adapting to your temperature preferences without your ever inputting a pre-selected schedule. Traditional or behavior-based automation also can be applied to virtually every gadget which will be remotely controlled – from sprinkler systems to coffee makers. Actual energy savings ultimately rely on the kind of device you choose and its automation capabilities. But on the average, product manufacturers estimate the systems can help consumers save anywhere from 10 to fifteen percent off of heating and cooling bills.

Convenience

In today's fast-paced society, the less you have got to stress about, the better. Right? Convenience is another primary point of home automation devices, which virtually eliminate small hassles like turning the lights off before you head to bed or adjusting the thermostat after you awaken within the morning. Many systems include remote dashboard capabilities, so forgetting to show off that coffee pot before you allow now not requires a visit back to the house. Simply pull up the dashboard on a sensible device or computer, and switch the coffee pot off in an exceedingly matter of seconds.

Security

Remote monitoring can put your mind comfy while you're removed from the house. With remote dashboards, lights and lamps is turned on and off, and automatic blinds are often raised and lowered. These capabilities – combined with automated security systems – can facilitate your mitigate the risks of intrusions: you'll be alerted immediately if something uncharacteristic happens.

Installation

Depending on the complexity of the system, installing a home automation device will be a big burden on the homeowner. It can either cost you money if you hire an outdoor contractor or cost you time if you venture to try to to it yourself.

Complex Technology

Automating everything in life may sound extremely appealing, but sometimes an honest old-fashioned flip of the switch may be a lot easier than reaching for your smart phone to show lights on and off. Before you choose which system is correct for you, give some thought to how far you actually want to require home automation in your household.

System Compatibility

Controlling all aspects of home automation from one centralized platform is very important, but not all systems are compatible with each other. Your security system, as

an example, may require you to log in to 1 location to manage settings, while your smart thermostat may require you to log in to a different platform to show the cooling on and off. to really leverage the convenience of home automation, you'll must invest in centralized platform technology to regulate all systems and devices from one location.

Cost

Even though the price of home automation systems has become much more affordable in recent years, the cost to purchase and install a device can still add up. Consumer Reports offers a wide range of information and insights – including costs – on the best home automation systems on the market and In each and every time the WiFi is not connected so, this is main disadvantage of home automation

1.2TECHNOLOGIES USED:

1.2.1 Deep learning

Deep learning could be a computer software that mimics the network of neurons in a very brain. it's a subset of machine learning and is named deep learning because it makes use of deep neural networks.

Deep learning algorithms are constructed with connected layers.

- The first layer is named the Input Layer
- The last layer is named the Output Layer
- All layers in between are called as Hidden Layers. The word deep means the network join neurons in additional than two layers.

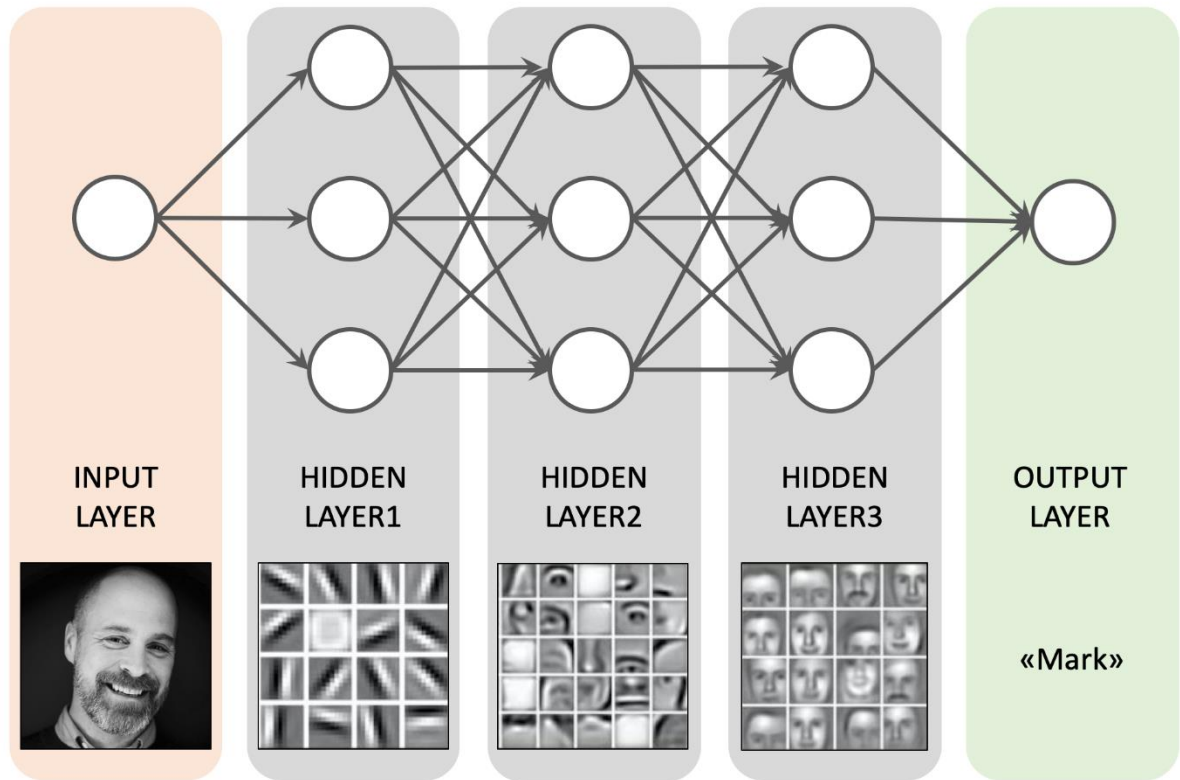


Figure 1.1 Deep Learning Layers

Each Hidden layer consists of neurons. The neurons are connected to every other. The neuron will process and so propagate the sign it receives the layer above it. The strength of the signal given the neuron within the next layer depends on the load, bias and activation function. The network consumes large amounts of input file and operates them through multiple layers; the network can learn increasingly complex features of the info at each layer.

1.2.1.1 Importance of Deep Learning:

Deep learning may be a powerful tool to create prediction and actionable result. Deep learning excels in pattern discovery (unsupervised learning) and knowledge-based prediction. Big data is that the fuel for deep learning. When both are combined, a corporation can reap unprecedented leads in terms of productivity, sales, management, and innovation. Deep learning can outperform traditional methods. As an example, deep learning algorithms are 41% more accurate than machine learning algorithms in image classification, 27% yet one more accurate in biometric authentication and 25% in voice recognition.

1.2.1.2 Deep Learning Process:

A deep neural network provides the state-of-the-art accuracy in many tasks, from object detection to speech recognition. They will learn automatically, without predefined knowledge explicitly coded by the programmers.

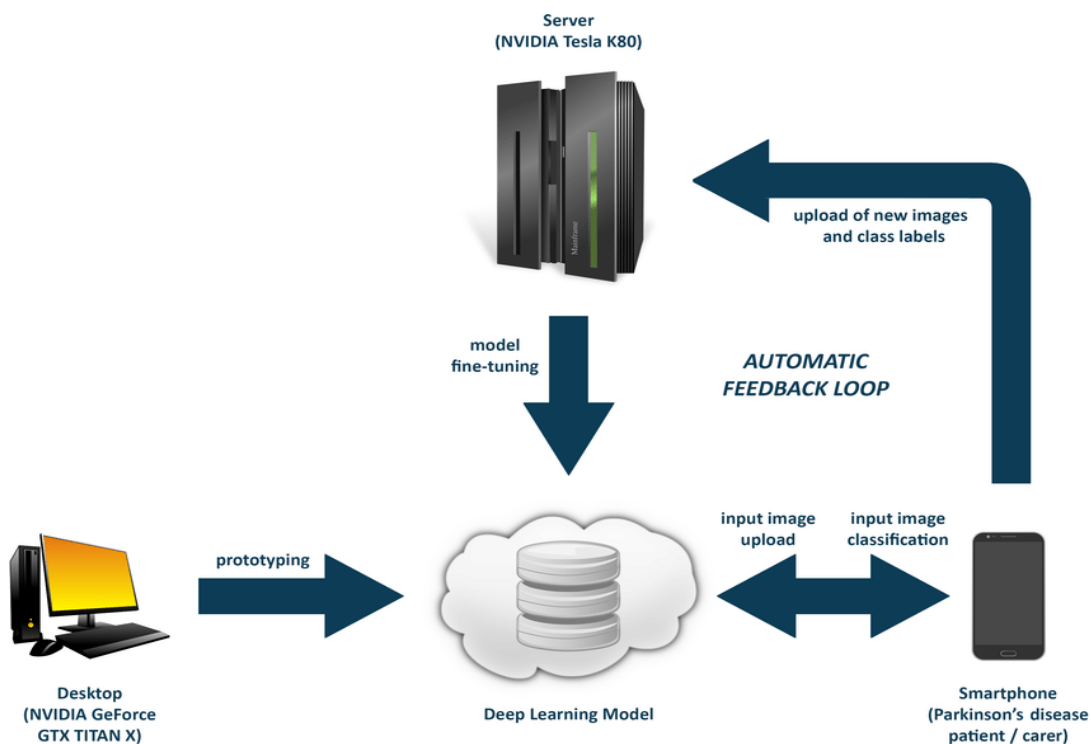


Figure 1.2 Deep Learning Process

To grasp the concept of deep learning, imagine a family, with an infant and oldsters. The toddler points objects along with his finger and always says the word 'cat.' As its parents are concerned about his education, they keep telling him 'Yes, that's a cat' or 'No, that's not a cat.' The infant persists in pointing objects but becomes more accurate with 'cats.' the insufficient kid, deep down, doesn't know why he can say it's a cat or not. He has just learned a way to hierarchies complex features springing up with a cat by watching the pet overall and still specialize in details like the tails or the nose before to form up his mind. A neural network works quite the identical. Each layer represents a deeper level of information, i.e., the hierarchy of information. A neural network with four layers will learn more complex feature than thereupon with two layers.

The learning occurs in two phases.

- The first phase consists of applying a nonlinear transformation of the input and make a statistical model as output.
- The second phase aims at improving the model with a mathematical method referred to as derivative.

Neural network repeats these two phases hundreds to thousands of your time until it's reached a tolerable level of accuracy. The repeat of this two-phase is termed an iteration.

1.2.1.3 Classification Of Neural Networks:

Shallow neural network: The Shallow neural network has just one hidden layer between the output and input. Deep neural network: Deep neural networks have over one layer. For example, Google LeNet model for image recognition counts twenty two layers. Nowadays, deep learning is employed in many ways sort of a driverless car, portable, Google program, Fraud detection, TV, and so on.

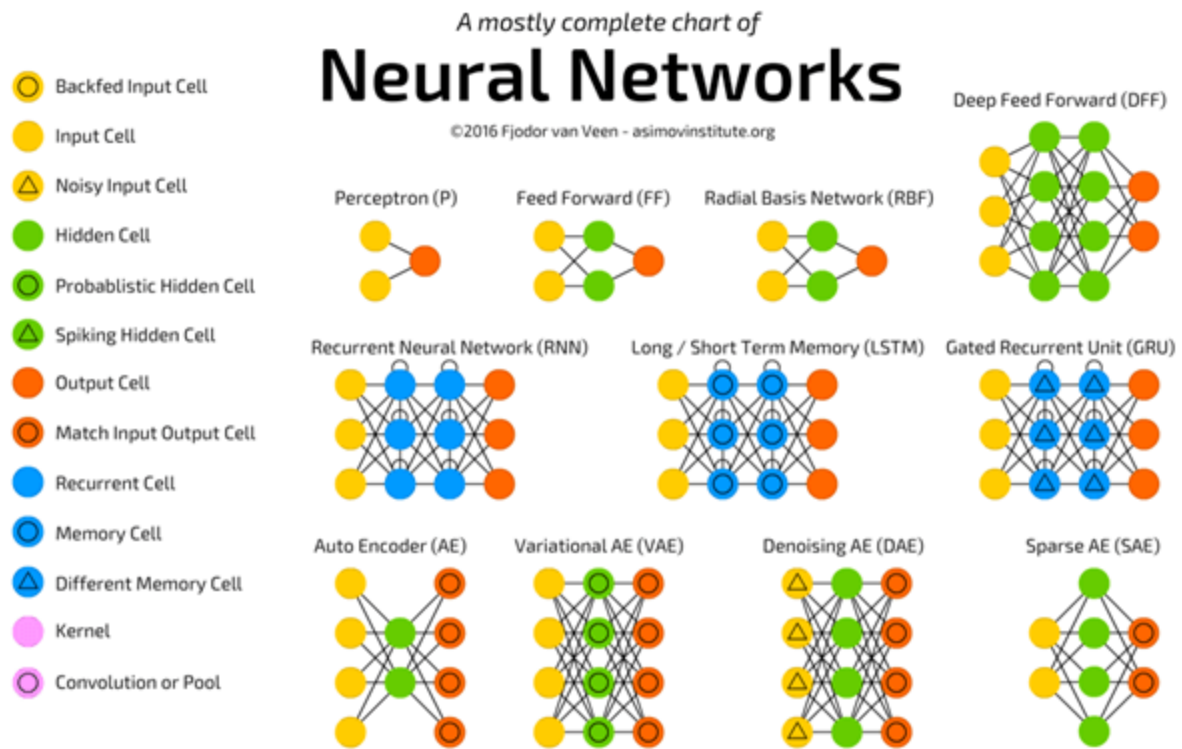


Figure 1.3 Types of Deep Learning Networks

1.2.1.3 Feed-Forward Neural Networks:

The simplest kind of artificial neural network. With this kind of architecture, information flows in barely one direction, forward. It means, the information's flows starts at the input layer and goes to the "hidden" layers, and end at the output layer. The network doesn't have a loop. Information stops at the output layers. Nour El Din Elmadany et (2018) For fusing information from multisets, we suggest two novel information fusion techniques.

1.2.1.4 Recurrent Neural Networks (Rnns):

RNN could be a multi-layered neural network that may store information in context nodes, allowing it to be told data sequences and output variety or another sequence. In simple words it a man-made neural networks whose connections between neurons include loops. RNNs are similar temperament for processing sequences of inputs.

- The RNN neurons will receive an indication that time to the beginning of the sentence.
- The network receives the word "Do" as an input and produces a vector of the quantity. This vector is fed back to the neuron to produce a memory to the network. This stage helps the network to recollect it received "Do" and it received it within the first position.
- The network will similarly proceed to the following words. The network takes the word "you" and "want." The state of the neurons is updated upon receiving each word.
- The conclusion occurs after receiving the word "a." The neural network will provide a probability for every English word which will be wont to complete the sentence. A well-trained RNN probably assigns a high probability to "café,""drink," etc.

1.2.1.5 Common uses of RNN:

- Help securities traders to generate analytic reports
- Detect abnormalities in the contract of financial statement
- Detect fraudulent credit-card transaction

- Provide a caption for images
- Power chatbots
- The standard uses of RNN occur when the practitioners are working with time-series data or sequences (e.g., audio recordings or text).

1.2.1.6 Convolutional Neural Networks (Cnn):

CNN could be a multi-layered neural network with a singular architecture designed to extract increasingly complex features of the info at each layer to work out the output. CNN's are compatible for perceptual tasks.

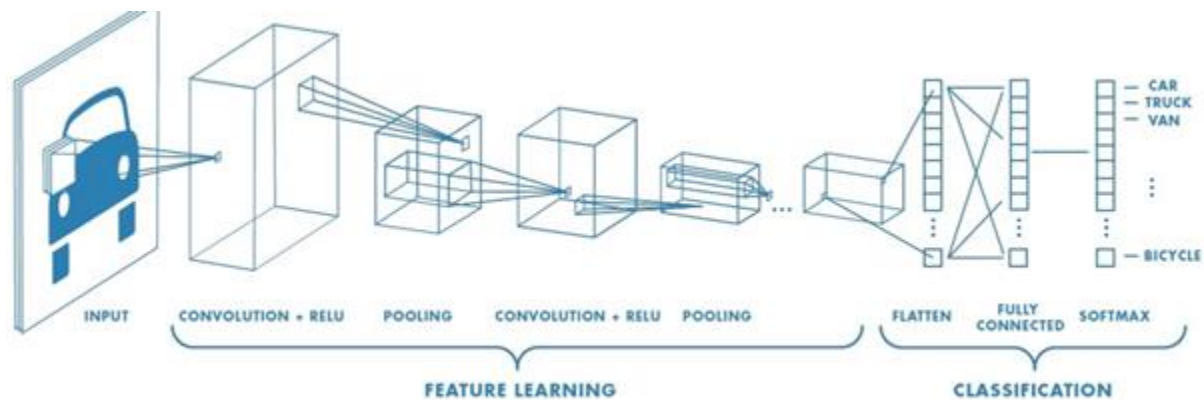


Figure 1.4 CNN

CNN is generally used when there's an unstructured data set (e.g., images) and therefore the practitioners must extract information from it

For instance, if the task is to predict a picture caption

- The CNN receives a picture of as an example a cat, this image, in computer term, may be a collection of the pixel. In general, a greyscale image has one layer and a colour image has three layers.
- During feature learning (i.e., hidden layers), the network can recognise specific

features such as the cat's tail, ear, and so on.

- When the network thoroughly learned a way to recognize an image, it can provide a probability for every image it knows. The label with the very best probability will become the prediction of the network.

Convolutional Neural Networks are multi-layer neural networks that recognise visual patterns directly from pixel images with little or no preprocessing. The ImageNet project may be a massive visual database intended for use in software research on visual perception. The ImageNet project runs an annual software contest, the ImageNet Large Scale Visual Recognition Challenge where software programs compete to properly classify and detect objects and scenes. Here i'll speak about CNN architectures of ILSVRC top competitors.

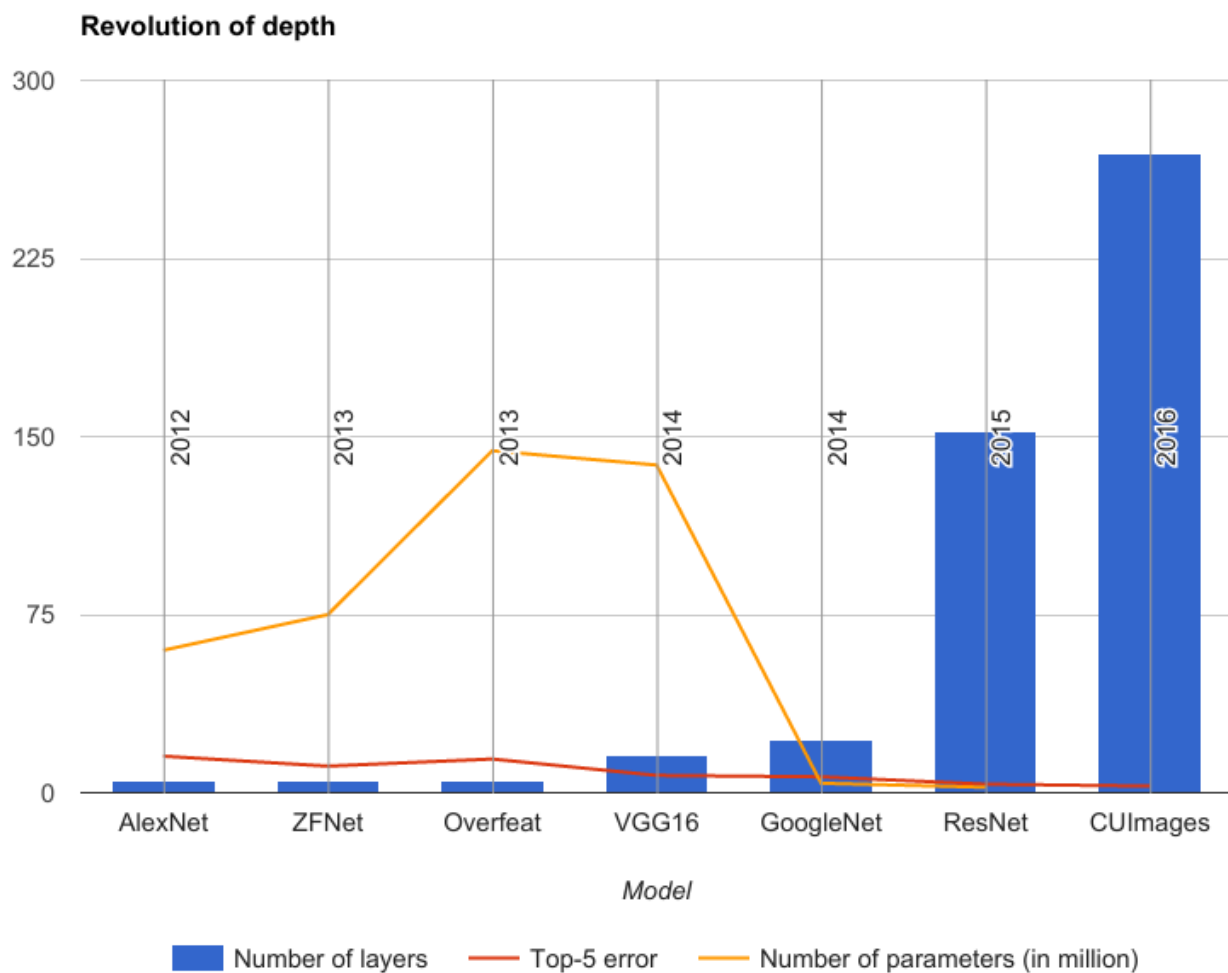


Figure 1.5 Types of CNN

CHAPTER - 2

LITERATURE SURVEY

2.1 INTRODUCTION:

The following shows survey did for Hand Gesture based home automation. The most popular of the existing techniques is been discussed as follows.

2.2 LITERATURE SURVEY:

Andrea Bandini and Jos´e Zariﬀa (2020).Due to the availability of inexpensive wearable cameras and large annotated datasets, egocentric vision (also known as first-person vision – FPV) applications have exploded in popularity in recent years. The position of the wearable camera (usually mounted on the head) allows recording exactly what the camera wearers have ahead of them, particularly hands and manipulated objects. This inherent benefit allows researchers to observe the hands from a variety of angles, including locating hands and their parts within photographs, determining what behaviour and activities the hands are engaged in, and creating human-computer interfaces that rely on hand gestures. We study the literature on hands using egocentric vision in this survey, categorising the current methods into three categories: localization (where are the hands or pieces of them?), interpretation (what are the hands doing?), and application (e.g., systems that used egocentric hand cues for solving a selected problem). Moreover, a listing of the foremost prominent datasets with hand-based annotations is provided.A list of the foremost prominent datasets with hand-based annotations is provided.The position of the wearable camera (usually mounted on the head) allows recording exactly what the camera wearers have before of them.

Qiang Nie, Jiangliu Wang et al, (2019) Skeleton-based human activity recognition has been a hot topic in recent years. Most existing researches are supported the skeleton data obtained from Kinect which is noisy and unstable, specifically within the case of occlusions.This paper introduces a view-invariant approach for human activity recognition

that recovers corrupted skeletons assisted by a 3D bio-constrained skeleton model and visualises certain body-level motion features obtained during the recovery process with images to resolve the noisy skeleton data and variance of viewpoints. The bio-constrained skeleton model is defined with two sorts of constraints: 1) constant bone lengths and 2) motion limits of joints. supported the bio-constrained model, a good method is proposed for skeleton recovery. Two kinds of new motion features, the Euclidean Distance Matrix between joints (JEDM) which contains the worldwide structure information of the body, and therefore the local dynamic variation of the joint Euler angles (JEAs), are employed in describing human activity. These two styles of features are encoded into different motion images which are fed into a two-stream convolutional neural network for learning different action patterns. a way for view-invariant skeleton-based action recognition supported a proposed 3D bio-constrained skeleton model. within the 3D bio-constrained skeleton model, lengths of bones are constant and joint motion limits recommended by the American Association of Orthopedic Surgeons (AAOS) are utilized. Skeleton recovery with the defined bio-constrained skeleton model is introduced to cater to corrupted skeletons or error joints. Skeleton recovery with the defined bio-constrained skeleton model is introduced to deal with corrupted skeletons or error joints. The project does not focus on the hand gestures.

An-An Liu* *et al*, (2018) in this study Multi-Domain & Multi Task Learning for Human Action Recognition on Transactions of Image Processing. Invariant across domains (view-invariant & modality- invariant) The ability to recognise human behaviour needs feature representation. Moreover, given a discriminative visual representation, it's critical to find the latent correlations among multiple actions so as to facilitate action modeling. To address these issues, we propose a multi-domain and multi-task learning (MDMTL) method for extracting domain-invariant information for multi-view and multi-modal action representation and exploring the relatedness among multiple action categories. For discriminative feature learning, we present a sparse transfer learning-based approach for co-embedding multi-domain (multi-view & multi-modality) data into one common space. In addition, for joint task modelling and task relatedness-induced feature learning, visual feature learning is integrated into the multi-task learning system,

with the Frobenius-norm regularisation term and thus the sparse constraint term. MDMTL is the first supervised system to realise domain-invariant feature learning and task modelling for multi-domain action recognition, to the best of our knowledge. We aim to resolve the multi-domain action recognition problem by exploring and fusing view-invariant and modality-invariant information. We proposed the multi-domain & multi-task learning (MDMTL) framework, which jointly addresses the issues of domain-invariant feature learning and multi-task modeling. Jointly addresses the problems of domain-invariant feature learning and multi-task modeling. The results are not reliable compared to any other projects.

Ping Wei *et al*, (2018) 3D human activity representation and recognition are important issues in many multimedia applications. Though latent state approaches are commonly used for action modelling, previous research has assumed that action latent states are single-attribute. This assertion is incorrect when describing complex action structures. We propose that latent states have composite attributes in this paper, and we present a special composite latent structure (CLS) model to describe and identify 3D human behaviour with skeleton sequences. An individual's action is modeled with a hierarchical graph, which represents the action sequence as sequential atomic actions. An atomic action is represented as a composite latent state which consists of a latent semantic attribution and a latent geometric attribution. A discriminative EM-like algorithm is proposed to be told the model parameters and also the composite latent structures of human actions. A composite attribute iterative programming algorithm is proposed to acknowledge the action and infer the action's latent temporal structure given a 3D skeleton sequence. The proposed approach is tested on the MSR 3D Action Dataset, Multiview 3D Event Dataset, and UTKinect-Action3D Dataset, which are all difficult 3D action datasets. Evaluate the proposed method on three challenging 3D action datasets - MSR 3D Action Dataset, Multiview 3D Event Dataset, and UTKinect-Action3D Dataset. The results of the project are not very accurate, hence is not applicable for real time usage.

Zhigang Tu *et al*, (2018) Despite outstanding performance in image recognition, convolutional neural networks (CNNs) don't yet achieve the identical impressive results

on action recognition in videos. this is often partially thanks to the lack of CNN for modeling long-range temporal structures especially those involving individual action stages that are critical to human activity recognition. We propose a unique action-stage emphasized spatio- temporal Vector of Locally Aggregated Descriptors method to aggregate informative deep features across the whole video in step with adaptive video features segmentation and adaptive segment feature sampling (AVFS-ASFS). The main frame features are selected and the related deep features are immediately broken into segments in our ActionS- ST-VLAD encoding approach, which uses AVFS-ASFS. The features in each segment belong to a temporally consistent ActionS. Then, using the extracted main frame attribute in each section as a reference, a flow-guided warping technique is used to detect and discard redundant feature charts, while the informative ones are aggregated using the similarity weight we exploited. In addition, we use an RGBF modality to catch motion salient regions within RGB images that resemble action movement. Exploit an RGBF modality to capture motion salient regions in the RGB images corresponding to action activity. The results are not up to the mark, hence is not applicable for long term purposes.

Nour El Din Elmadany *et al*, (2018) The problem of human activity recognition, during which each action is captured by multiple sensors and represented by multisets. We propose two novel information fusion techniques for fusing the knowledge from multisets. the primary technique is biset globality locality preserving canonical correlation analysis (bglpcca), which aims to be told the common feature subspace between two sets. The second technique is multiset globality locality preserving canonical correlation analysis (mglpcca), which aims to cater to three or more sets. The proposed bglpcca and mglpcca are ready to learn a low-dimensional common subspace that retains information samples' local and global structures. Furthermore, two new descriptors for both depth and skeleton are introduced. We then propose a replacement act recognition framework employing the proposed bglpcca or mglpcca to be told the shared subspace from multiple sets of features including skeleton, depth, and optical flow. Biset globality locality preserving canonical correlation analysis (bglpcca) for learn- ing the common subspace from two sets, and multiset globality locality preserving canonical correlation analysis

(mglpcca) for learning the common subspace from three or more sets. In feature extractions, two new descriptors were introduced. Boa was proposed for skeleton and hpdmm-cnn for depth. The project gives solutions to two major challenges for robotic applications. The project is fundamentally not satisfying the problem of the hand gesture.

Sid Ahmed Walid Talha¹ *et al*, (2018) Human Action recognition (HAR) is basically utilized in the sphere of Ambient Assisted Living (AAL) to make an interaction between humans and computers. In these applications, it can't be asked to people to act non-naturally. The algorithm should adapt and therefore the interaction needs to be as quick as possible to create this interaction fluent. To enhance the present algorithms with regards thereto points, we propose a completely unique method supported skeleton information provided by RGB-D cameras. This approach is ready to hold out early action recognition and is more robust to viewpoint variability. To achieve this goal, a brand new descriptor called Body Directional Velocity is proposed and a real-time classification is performed. A novel approach to perform human activity recognition using RGB-D sensors. We've got focused our work on two major challenges for robotic applications, the robustness to viewpoint changes and therefore the early recognition property. The project focuses primarily on two major challenges for robotic applications, the robustness to viewpoint changes and the early recognition property. The project uses the RGB-D cameras that are very poor in quality.

Aouaidjia Kamel *et al*, (2018) A method (action-fusion) for human activity recognition from depth maps and posture data using convolutional neural networks (cnns). Two input descriptors are used for action representation. The primary input may be a depth motion image that accumulates consecutive depth maps of somebody's action, whilst the second input may be a proposed moving joints descriptor which represents the motion of body joints over time. So as to maximise feature extraction for accurate action classification, three cnn channels are trained with different inputs. The primary channel is trained with depth motion images (dmis), the second channel is trained with both dmis and moving joint descriptors together, and also the third channel is trained with moving joint descriptors only. The action predictions generated from the three cnn channels are fused together for the ultimate action classification. We propose several fusion score

operations to maximise the score of the proper action. The effects by fusing the outputs of three channels was greater than using one channel or fusing two channels alone, according to the experiments. Our proposed approach was tested on three publicly available datasets: 1) the microsoft action 3-d dataset (msraction3d); 2) the university of Texas at Dallas-multimodal human behaviour dataset; and 3) the multimodal action dataset (mad). The results of the fusing output of three channels are better than using one channel or fusing two channels only. The project cannot be trusted, since this project lacks human face detection.

Hongsong Wang *et al*, (2018) Skeleton-based action recognition becomes popular due to the event of cost-effective depth sensors and fast pose estimation algorithms. Traditional methods are supported pose descriptors often fails on large scale datasets because of the limited representation of engineered feature. Recent recurrent neural networks (RNN) based approaches mostly specialize in the temporal evolution of body joints and neglect the geometric relations. during this paper, we aim to leverage the geometric relations among joints for act recognition. We introduce three primitive geometries such as joints, edges and surfaces. Accordingly, a generic end-to-end Recurrent Neural Networks based network is meant to accommodate the three inputs. For action recognition, a unique viewpoint transformation layer and temporal dropout layers are utilized within the RNN based network to be told robust representations. And for action detection, we first perform frame-wise action classification, then exploit a completely unique multi-scale window algorithm. we aim to be told representations from primitive geometries of human skeletons, i.e., joints, edges and surfaces. For action recognition, we propose a unique RNN based architecture to accommodate the three inputs. For action detection, we first perform frame-wise action classification, then design a multi-scale window search algorithm to get detection results. First performs frame-wise action classification, then design a multi-scale sliding window search algorithm to generate detection results. There are some accuracy issues because it focuses primarily on the recognition part and not the security part.

Hossein Rahmani *et al*, (2016) Recognizing the human actions from unknown and unseen views may be a challenging problem. We propose a robust Non-

Linear Knowledge Transfer Model (R-NKTM) for recognising human interaction from novel perspectives. The proposed R-NKTM is a deep fully-connected neural network that finds a series of non-linear transformations that links the views and passes information of human behaviour from some unknown view to a shared high-level virtual view. The R-NKTM is based on 2D simulations of dense trajectories of virtual 3D human models fitted to actual motion capture data, and it generalises to real-world videos of human behaviour. The beauty of our method is that we only need to master one R-NKTM for all acts and all views in order to move information from any actual human activity video without having to retrain or fine-tune the model. Thus, R-NKTM can efficiently scale to include new action classes. R-NKTM is learned with dummy labels and doesn't require knowledge of the camera viewpoint at any stage. an algorithm for unsupervised learning of a strong Non-linear Knowledge Transfer Model (R-NKTM) for cross-view action recognition. Robust Non-linear Knowledge Transfer Model (R-NKTM) for cross-view action recognition is best when comes to recognition. R-NKTM is learned with dummy labels and does not require knowledge of the camera, making it not very user-friendly.

Jian Cheng *et al*, (2016) supervised temporal t-stochastic neighbour embedding (st-tsne) and increased learning were enabled by a human behaviour recognition system for human silhouette sequences. Inspired by the some and its variants, is proposed to be told the underlying relationship between action frames in a very manifold, where the category label information and temporal information are introduced to well represent those frames from the identical action classes. We present three methods for performing low-dimensional embedding of recent data, which is an important step in action recognition. Two of them, locally linear embedding and locality preserving projection, are inspired by local approaches. These two methods are suggested for determining concrete linear representations based on the local neighbour relationship, and their efficacy in maintaining the intrinsic action structure is studied. the remainder one is predicated on manifold-oriented stochastic neighbor projection to search out a linear projection from high-dimensional to low-dimensional space capturing the underlying pattern manifold. Learns explicit linear representations following the local neighbor relationship, and their effectiveness is investigated for preserving the intrinsic action

structure. The proposed project is based on the neighboring embeddings making it very inefficient.

Sungpill Choi *et al.*, (2017) Hand gesture recognition is considered new Human Computer Interaction (HCI) technologies for the following generation of mobile devices. Previous hand gesture implementation requires an oversized memory and computation power for hand segmentation, which fails to relinquish real-time interaction with mobile devices to users. As a result, we introduce a low-latency and memory-efficient hand segmentation architecture for natural hand gesture recognition in this paper. We suggest a streaming hand contour tracing unit and a fast contour filling unit to achieve both high memory performance and low latency. In comparison to the best-in-class, it reaches 7.14 ms latency with just 34.8 KB on-chip memory, which is 1.65 times less latency and 1.68 times less on-chip memory. It achieves 7.14 ms latency with only 34.8 KB on-chip memory, which are 1.65 times less latency and 1.68 times less on-chip memory, respectively, compare to the best-in-class. It is very poor in terms of optimization, hence, making the whole system very inefficient.

Renliang Weng *et al.*, (2015) Over the past three decades, variety of face recognition methods are proposed in computer vision, and most of them use holistic face images for person identification. Human faces may be occluded by other artefacts in several real-world contexts, particularly in certain unconstrained settings, making it challenging to obtain completely holistic face images for identification. To handle this, we propose a replacement partial face recognition approach to acknowledge persons of interest from their partial faces. We first detect main points and remove their local textural characteristics from a gallery image and a probe face patch. Then, to discriminatively compare these two extracted local feature sets, we propose a strong point set matching (RPSM) process, in which both the textural and geometrical knowledge of local features are directly used for matching at the same time. Lastly, the similarity of two faces is converted because the distance between these two aligned feature sets. The suggested method's efficacy is shown by experimental findings on four public face datasets. Experimental results on four public face datasets shows the effectiveness of the

proposed approach. It is very poor in terms of optimization, hence, making the whole system very inefficient.

Abhijith Punnapurath *et al*, (2015) Existing methods for performing facial recognition in the presence of blur depend on the convolution paradigm, which can't accommodate non-uniform blurring that occurs often in hand-held cameras due to tilts and rotations. We suggest a technique for face recognition using arbitrarily-shaped kernels in the presence of space-varying motion blur in this article. We prove that the set of all images obtained by non-uniformly blurring a given image forms a convex set by modelling the blurred face as a convex mixture of geometrically transformed instances of the centred gallery face. We first suggest a nonuniform blur-robust algorithm based on the concept of a sparse camera trajectory within the camera motion space, which we use to construct an energy function with a l1-norm restriction on the camera motion. The framework is then extended to handle illumination variations by exploiting the very fact that the set of all images obtained from a face image by non-uniform blurring and changing the illumination forms a bi-convex set. Finally, we propose a sublime extension to also account for variations in pose. The project is an elegant extension to also account for variations in pose. The project is very insecure since anyone can perform the gesture to unlock the actions.

Jiwen Lu *et al*, (2015) Because of their excellent robustness and strong discriminative strength, binary feature descriptors like local binary patterns (LBP) and its variants are commonly used in many face recognition systems. However, the majority of current binary face descriptors are hand-crafted, which necessitates extensive advanced information. We suggest a compact binary face descriptor (CBFD) function learning approach for face recognition and representation in this article. We first compute the difference between each pixel and its neighbouring pixels to obtain pixel difference vectors (PDVs) in local patches from each face image. After that, we learn an unsupervised attribute mapping to project these pixel difference vectors (PDVs) into low-dimensional binary vectors, where 1) the variance of all binary codes within the training set is maximised, 2) the loss between the first real-valued codes and even the learned binary codes is minimised, and 3) the redundancy information in PDVs is eliminated and

comprehensible. Finally, as the final representation for each face graphic, we cluster and pool these binary codes into a histogram function. Furthermore, we suggest a coupled CBFD (C-CBFD) system for heterogeneous face recognition by reducing the modality distance between heterogeneous faces at the feature stage. Our methods outperform state-of-the-art face descriptors, according to extensive experimental findings on five commonly used face datasets. Results on five widely used face datasets shows that our methods outperform state-of-the-art face descriptors. This system is hand crafted making it a very tiresome procedure to follow, hence is not recommended.

2.3 MAJOR DISADVANTAGES IN EXISTING SYSTEM

- The existing system for automation has only Hand gesture recognition.
- Not reliable if no internet connection is available.
- Normal people cannot afford.
- Failure of sensor modules will lead to inefficient automation.

CHAPTER - 3

AIM AND SCOPE

3.1 AIM

During this proposed work we'll be using AI which recognizes human faces and if the registered human is successfully detected then it recognizes the human gestures without the employment of any reasonably sensors with the assistance of a camera and acts accordingly to regulate any reasonably appliance. Thus, our project aims at discovering an automation system without the utilization of sensors or a mobile app with the assistance of AI.

3.2 EXISTING SYSTEM:

Owing to the availability of low-cost wearable cameras and large annotated datasets, egocentric vision (also known as first-person vision – FPV) implementations have exploded in popularity in recent years. The position of the wearable camera (usually mounted on the head) allows recording exactly what the camera wearers have before of them, specifically hands and manipulated objects. This inherent benefit allows researchers to observe the hands from a variety of angles, including locating hands and their parts within photographs, determining what behaviour and behaviours the hands are engaged in, and creating human-computer interfaces that rely on hand movements. We study the literature on hands using egocentric vision in this report, categorising the most common methods into three categories: localization (where are the hands or pieces of them?), understanding (what are the hands doing?), and implementation (e.g., systems that used egocentric hand cues for solving a selected problem). Moreover, an inventory of the foremost prominent datasets with hand-based annotations is provided.

3.3 DISADVANTAGES OF EXISTING SYSTEM:

- The existing system for automation has only Hand gesture recognition.
- Not reliable if no internet connection is available.
- Normal people cannot afford.

- Failure of sensor modules will lead to inefficient automation.

3.4 PROPOSED SYSTEM:

Automation is that the essential need for this world. There are various sorts of automations like building automation, industrial automation, Home automation, computing, etc. may be a possible application which provides support to any or all the persons. Home automation is that the use and control of home appliances remotely or automatically. The distance between robots and humans is closing every day. Hand gesture-based home automation is gaining popularity these days. Recognizing the expression of human parts such as the hand, ears, and so on is known as gesture recognition. The majority of electronic component manufacturers base their products on hand gestures. during this proposed work we are going to be using computing which recognizes human faces and if the registered human is successfully detected then it recognizes the human gestures without the utilization of any reasonably sensors with the assistance of a camera and acts accordingly to manage any reasonably household appliance. Thus, our project aims at discovering an automation system without the utilization of sensors or a mobile app with the assistance of computer science.

3.5 ADVANTAGES OF PROPOSED SYSTEM

- Face recognition based Hand gesture auto appliance control
- More reliable
- No sensors required
- No internet connectivity needed
- Suitable for physically challenged people

3.5 APPLICATIONS:

- Automate the control of devices using gesture recognition.
- Applicable in all work places, homes, etc.
- Helpful for physically challenged people.

CHAPTER 4

MATERIALS AND METHODS

DETAILED DESIGN OF THE PROJECT:

This chapter describes the overall detailed architectural design. It also describes each module that is to be implemented along with Data Flow diagram.

4.1 ARCHITECTURAL DIAGRAM:

The architectural diagram is shown in

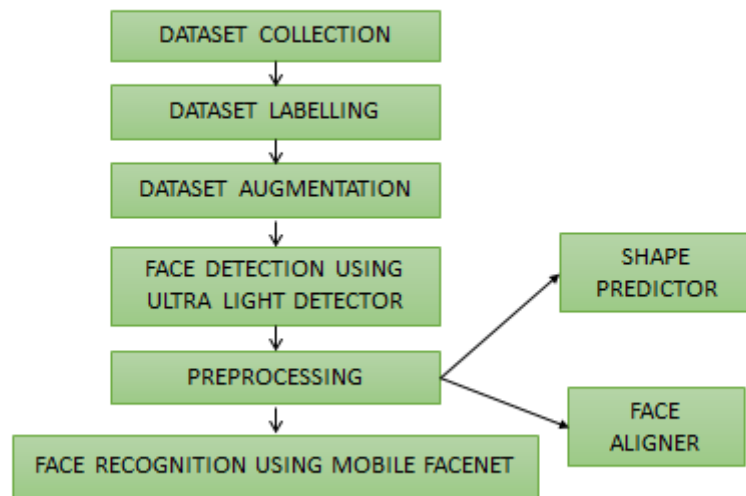


Figure 4.1: Face detection architecture

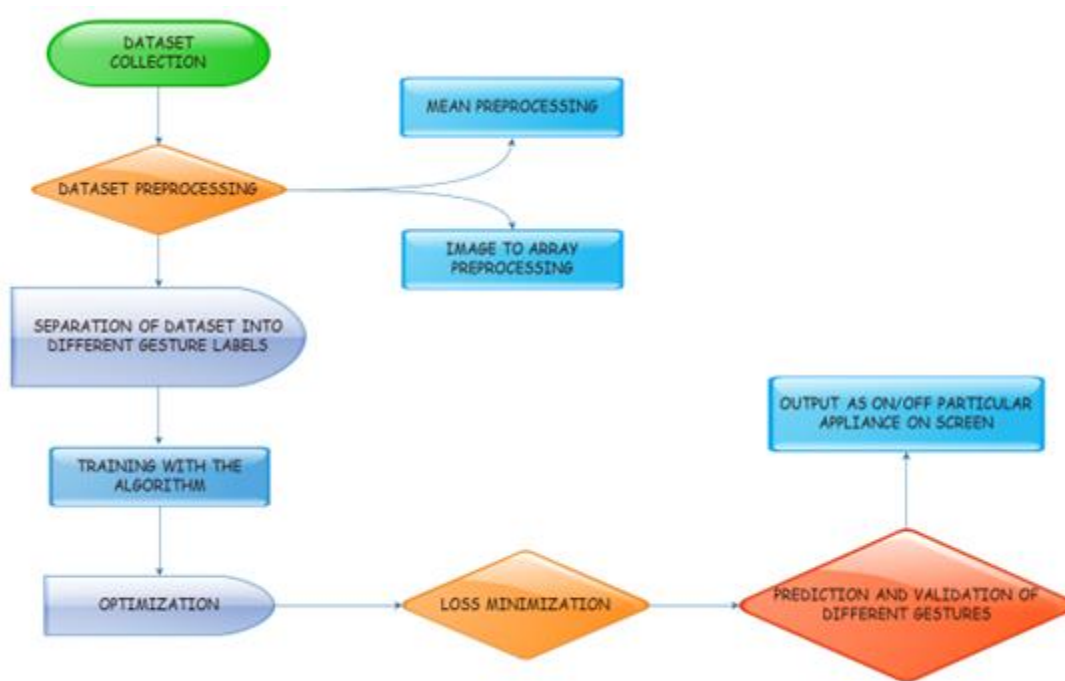


Figure.4.2 Gesture recognition

4.2 WORKING:

In this project, we will be designing a hand gesture recognition based home automation system. The initial step would be to collect the datasets of the faces of all registered people living in the house. Where only when the face of the person is recognized the hand gestures of the person are the accepted. will. The next step is to collect the dataset of different gestures by taking pictures of hands and collecting from various resources from internets.so, both these datasets are them trained by raspberry pi or the pc to identify the faces and particular gestures in order to switch on and off the appliances. so, initially raspberry pi will be configured with raspbian OS, it will be given with opencv libraries which will be used to trained the model in order to recognize the gestures. A camera module will be interfaced with raspberry pi/ PC system which will lively stream the gestures shown in front of it. These gestures are captured and given to the raspberry pi which will analyse it to switch on and off the appliances. From the dataset collected initially we will train the model and preprocess the gestures in order to get the

approximate output. A driver circuit with a relay is also connected to the raspberry pi which will switch on and off high voltage appliances. Whenever the camera gives a face input, and then accepts the hand gesture input only if the face is recognized and then compares as to which appliance should be switched on based on that particular gesture. From this model we will be able to control appliances without any internet connection using artificial intelligence.

4.3 MODULES DESCRIPTION:

- Dataset Collection Module
- Face detection module
- Face recognition module
- Hand Segmentation
- Training with Gesturennet
- Compare the gestures and Switching the devices on off

4.3.1 Data Collection Module:

A data set may be a collection of knowledge. Deep Learning has emerged as the preferred tool for tackling a wide range of difficult real-world problems. It's definitely far and away the most effective performing method for computer vision tasks. The image above showcases the ability of deep learning for computer vision. With enough training, a deep network can segment and identify the "key points" of each person within the image. These deep learning machines that are working so well need fuel many fuel; that fuel is data. The more labelled data we've, the higher our model performs. Experiments performed by An-An Liu et al. (2019) on the INRIA Xmas Motion Acquisition Sequences data collection, the MSR Everyday Operation 3D (DailyActivity3D) data set, and the Multi-modal & Multi-view & Interactive data set, which is the most recent and largest multi-view and multi-model behaviour recognition data set, indicate that MDMTL outperforms state-of-the-art approaches. The thought of more data resulting in better performance has even been explored at a large-scale by Google with a dataset of 300 Million images! When deploying a Deep Learning model in an exceedingly real-world application, data must be constantly

fed to continue improving its performance. And, within the deep learning era, data is extremely well arguably your most useful resource. There are three steps of collecting data

Scraping From the Web

Manually finding and downloading images takes a protracted time simply thanks to the number of human work involved. The task probably has some quite common objects are to be detected. then that becomes the keyword for web-scraping. It also becomes the category name for that object. From the sounds of it this is often after all very easy for a task like image classification where the photographs annotations are quite coarse. But to try and do something like instance segmentation? we'd like labels for each single pixel within the image. to urge those, it's best to use some really great image annotation tools that are already out there. The paper shows the way to create a model that, given a rough set of polygon points around an object, can generate the pixel labels for segmentation. Deep extreme cut is additionally quite similar except they use only the four extreme points round the object. this may then give some nice bounding box and segmentation labels. an alternative choice is to use an existing image annotation GUIs. Label someone very fashionable where one can draw both bounding boxes and set polygon points for segmentation maps. Amazon Mechanical Turk is additionally an inexpensive option.

Third-party:

Since data has become such a valuable commodity within the deep learning era, many start-ups have began to offer their own image annotation services they'll gather and label the info. Given an outline of what quite data and annotations needed. Mighty, a company that specialises in self-driving car picture annotation and has grown to be a major player in the field, was also present at CVPR 2018. Payment AI is less advanced than Mighty AI, and it can annotate images from any domain. They also offer a pair more tools like video and landmark annotations.

4.3.2 Face Detection Module

The "Ultra-Light-Fast-Generic-Face-Detector" is designed for general-purpose face detection applications in low-power computing devices, including Android and iOS tablets, and PCs. The model could be a real-time ultra-lightweight universal face detection model designed for edge computing devices or low-power devices. It are often utilized in

low-power computing devices like ARM for real-time common scene faces. Hongsong Wang and colleagues (2018) The aim of this paper is to use geometric relationships between joints to recognise behaviour. Three simple geometries are introduced: joints, corners, and textures. Facial recognition technology is widely used in security monitoring, surveillance, human-computer interaction, entertainment, etc. Detecting human faces in digital images is that the opening in biometric authentication, and a perfect face detection model is evaluated by how quickly and accurately it performs. Aouaidjia Kamel and her colleagues (2019) We introduce a framework (Action-Fusion) for recognising human actions from depth maps and posture data using convolutional neural networks in this article (CNNs). The Face-Detector-1MB stands out at terms of speed — the model's default FP32 precision (.pth) file size is 1.1MB, and also the inference frame int8 is quantized to a size of 300KB. In terms of model calculation, the input resolution of 320×240 is merely about 90 to 109 MFlops. The Face-Detector-1MB training process used a VOC dataset generated by the broader FACE dataset, a face detection benchmark. WIDER FACE has been released in 2015 and consists of 32,203 images and 393,703 face bounding boxes with a high degree of variability in scale, pose, expression, occlusion and illumination.

The 1MB lightweight model comes during a version-slim with slightly faster simplification, and a version-RFB with a modified RFB module for higher precision. The model has been tested on Ubuntu16.04, Windows 10, Python3.6, Pytorch1.2, CUDA10.0, etc.

Features

The default FP32 precision (.pth) file size is 1.041.1MB, so the inference frame int8 is roughly 300KB.

- The input size of 320x240 is around 90109 M-Flops in the model's estimation.
- The model is available in two versions: version-slim (slightly quicker simplification) and version-RFB (with the modified RFB module, higher precision).
- Provides pre-training templates with larger face training at 320x240 and 640x480 input resolutions to increase the number of possible implementation scenarios.
- Onnx export is supported, and it's simple to instal.

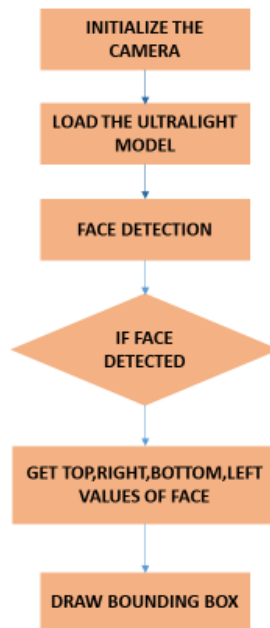


Figure 4.3 Ultralight face detector flow diagram

4.3.3 Face Recognition Module

For recognizing the face we use mobile facenet which could be a more accurate in classifying face. MobileFaceNet may be a neural network and obtains accuracy upto 99.28 percent on labelled faces within the wild (LFW) dataset, and a 93.05 percent accuracy on recognising faces within the AgeDB dataset. The network used around 1,000,000 parameters taking only 24 milliseconds to run and produce results on a Qualcomm Snapdragon processor. we are able to compare this performance to accuracies of 98.70 percent and 89.27 percent for ShuffleNet, which has more parameters and takes a touch longer to execute on the CPU.

The researchers have made it easy to switch the world average pooling layer within the CNN with a depth wise convolution layer, which improves performance on identity verification. This development is basically important because the AI world searches for efficient models that run on small compute powers which are available on today's mobile phones.

Compressing pretrained networks via information distillation is another method for obtaining lightweight facial verification models. Such methods obtained 97.32 percent facial authentication accuracy on LFW with a model size of 4.0 MB. MobileFaceNets achieves equal precision on a low cost budget, which is a phenomenal accomplishment.

MobileFaceNet architecture is partially inspired by the MobileNetV2 architecture. The residual bottlenecks that proposed in MobileNetV2 are used as our main building blocks. The researchers use PReLU because the non-linearity, which is most suited to facial verification than using ReLU. The researchers also use a quick downsampling strategy at the start of the network, and a linear 1×1 convolution layer following a linear global depthwise convolution layer because the feature output layer. The detailed architecture is mentioned in the table below:

Input	Operator	t	c	n	s
$112^2 \times 3$	conv3x3	-	64	1	2
$56^2 \times 64$	depthwise conv3x3	-	64	1	1
$56^2 \times 64$	bottleneck	2	64	5	2
$28^2 \times 64$	bottleneck	4	128	1	2
$14^2 \times 128$	bottleneck	2	128	6	1
$14^2 \times 128$	bottleneck	4	128	1	2
$7^2 \times 128$	bottleneck	2	128	2	1
$7^2 \times 128$	conv1x1	-	512	1	1
$7^2 \times 512$	linear GDConv7x7	-	512	1	1
$1^2 \times 512$	linear conv1x1	-	128	1	1

Table 4.1: MobileFaceNet architecture

The primary MobileFaceNet network uses 0.99 million parameters. to cut back computational cost, the researchers decided to alter input resolution from 112×112 to 112×96 or 96×96. The the linear 1×1 convolution layer after the linear GDConv layer was also far from MobileFaceNet. this provides a resulting network called MobileFaceNet M.

The researchers have used MobileNetV1, ShuffleNet, and MobileNetV2 because the baseline models. For a fair comparison of results, all MobileFaceNet models and baseline models are trained from scratch on the CASIA-Webface dataset by ArcFace failure. The training is finished at 60K iterations.

MobileFaceNet, MobileFaceNet (11296), and MobileFaceNet (9696) are trained on the MSCeleb-1M database's cleaned training collection, which contains 3.8 million images from 85,000 subjects, in order to achieve even better results. On LFW and AgeDB-30, the precision of our primary MobileFaceNet is increased to 99.55 percent and 96.07 percent, respectively.

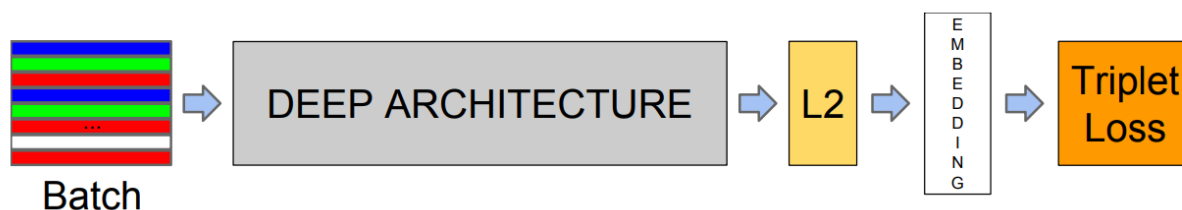


Figure 4.4 MobileFaceNet architecture

4.3.4 Hand Segmentation:

Because of the complexities of vision-based approaches, such as changing lighting conditions, complicated backgrounds, and colour detection; difference in human colouring complexion demanded the rigorous creation of algorithm for natural interface. Color is extremely powerful descriptor for object detection. As a result, colour information was used for segmentation, which is invariant to hand rotation and geometric variation. Human perceives characteristics of color component like brightness, saturation and hue component than the share of primary color red, green, and blue. Color models are useful for to specify a selected modify standard way. it's space-coordinated system within which any specified color represented by single point. Three strategies for robust hand identification and segmentation were presented here, each using a different colour space. For the preprocessing of the HGR method, a hand tracking and segmentation (HTS) technique using the HSV colour space has been established.

4.3.5 Training with gesturennet:

The original images used for hand gesture recognition within the work are demonstrated. These images are captured with a standard camera. These hand images are taken under the identical condition. The background of those images is identical. So, it's easy and effective to detect the hand region from the first image using the background subtraction method. However, in some cases, there are other moving objects included within the results of background subtraction. The complexion is accustomed discriminate the hand region from the opposite moving objects. Hossein Rahmani and his colleagues (2016) For human behaviour recognition from novel perspectives, we suggest the R-NKTM (Robust Non-Linear Information Transfer Model). The proposed R-NKTM is a deep fully-connected neural network that seeks a non-linear virtual path that links the views to transfer information of human behaviour from some unknown view to a shared high-level virtual view. the colour of the skin is measured with the HSV model. The skin color's HSV (hue, saturation, and value) values are 315, 94, and 37, respectively. The gesture recognition invariant to image scale is formed by resizing the image of the detected hand. The hand detection output may be a binary image with white pixels representing representatives of the hand area and black pixels representing the background. An example of the hand detection. Then, the subsequent procedure is implemented on the binary hand image to segment the fingers and palm the palm point is defined because the center point of the palm. it's found by the strategy of distance transform. Distance transform also called distance map may be a representation of a picture. within the distance transform image, each pixel records the space of it and therefore the nearest boundary pixel. An example of distance transform is demonstrated may be a binary image and is that the distance transform image. The block city distance is employed to live the distances between the pixels and also the nearest boundary pixels. As is shown within the figure, the middle point of the binary image is with the most important distance. Thus, within the distance transform image of the binary hand image, the pixel with largest distance is chosen because the palm point. The found palm point is marked with the purpose of the green color. In this, we introduce a brand new algorithm.

It's named as gesturenent. using this algorithm with different gestures are trained with high accuracy whenever, the input is given videos streaming that may be recognize the model automatically.



Figure 4.5 Hand Gesturenent diagram

4.3.6 Compare the gestures and switch the devices ON and OFF:

Combination of gestures to control the fan & light ON & OFF. Without internet connection the gestures can control the fan & light ON & OFF. Incoming gestures is always monitored and type of gestures which will be used to switch the appliances ON & OFF are compared from the incoming gestures. With the gestures compared, the fan/light are switched ON or OFF. The aged people and the physically challenged people can use this efficiently. Thus this project helps in automatic control of appliances using artificial intelligence.

4.7 SOFTWARES USED

4.7.1 VIRTUAL MACHINE

A virtual machine app creates a virtualized environment—called, simply enough, a virtual machine—that behaves like a separate computer system, complete with virtual hardware devices. The VM runs as a process in a window on your current operating system. You can boot an operating system installer disc (or live CD) inside the virtual machine, and the operating system will be “tricked” into thinking it’s running on a real computer. It will install and run just as it would on a real, physical machine. Whenever you want to use the operating system, you can open the virtual machine program and

use it in a window on your current desktop. In the VM world, the operating system actually running on your computer is called the host and any operating systems running inside VMs are called guests. It helps keep things from getting too confusing. In a particular VM, the guest OS is stored on a virtual hard drive—a big, multi-gigabyte file stored on your real hard drive. The VM app presents this file the guest OS as a real hard drive. This means you won't have to mess around with partitioning or doing anything else complicated with your real hard drive. Virtualization does add some overhead, so don't expect them to be as fast as if you had installed the operating system on real hardware.

4.7.2 PYCHARM CODE EDITOR

PyCharm is an integrated development environment (IDE) used in computer programming, specifically for the Python language. It is developed by the Czech company JetBrains. It provides code analysis, a graphical debugger, an integrated unit tester, integration with version control systems (VCSes), and supports web development with Django as well as data science with Anaconda. PyCharm is cross-platform, with Windows, macOS and Linux versions. The Community Edition is released under the Apache License, and there is also Professional Edition with extra features – released under a proprietary license

Features

- Coding assistance and analysis, with code completion, syntax and error highlighting, linter integration, and quick fixes
- Project and code navigation: specialized project views, file structure views and quick jumping between files, classes, methods and usages
- Python refactoring: includes rename, extract method, introduce variable, introduce constant, pull up, push down and others
- Support for web frameworks: Django, web2py and Flask
- Integrated Python debugger
- Integrated unit testing, with line-by-line code coverage

- Google App Engine Python development
- Version control integration: unified user interface for Mercurial, Git, Subversion, Perforce and CVS with change lists and merge
- Support for scientific tools like matplotlib, numpy and scipy

It competes mainly with a number of other Python-oriented IDEs, including Eclipse's PyDev, and the more broadly focused Komodo IDE.

CHAPTER - 5

RESULTS AND DISCUSSION

5.1 FINAL OUTPUTS OBTAINED

We have developed hand gesture recognition based home automation system. The first step will be dataset collection which can be collected using real time video streaming. that can be seen in the following figure.

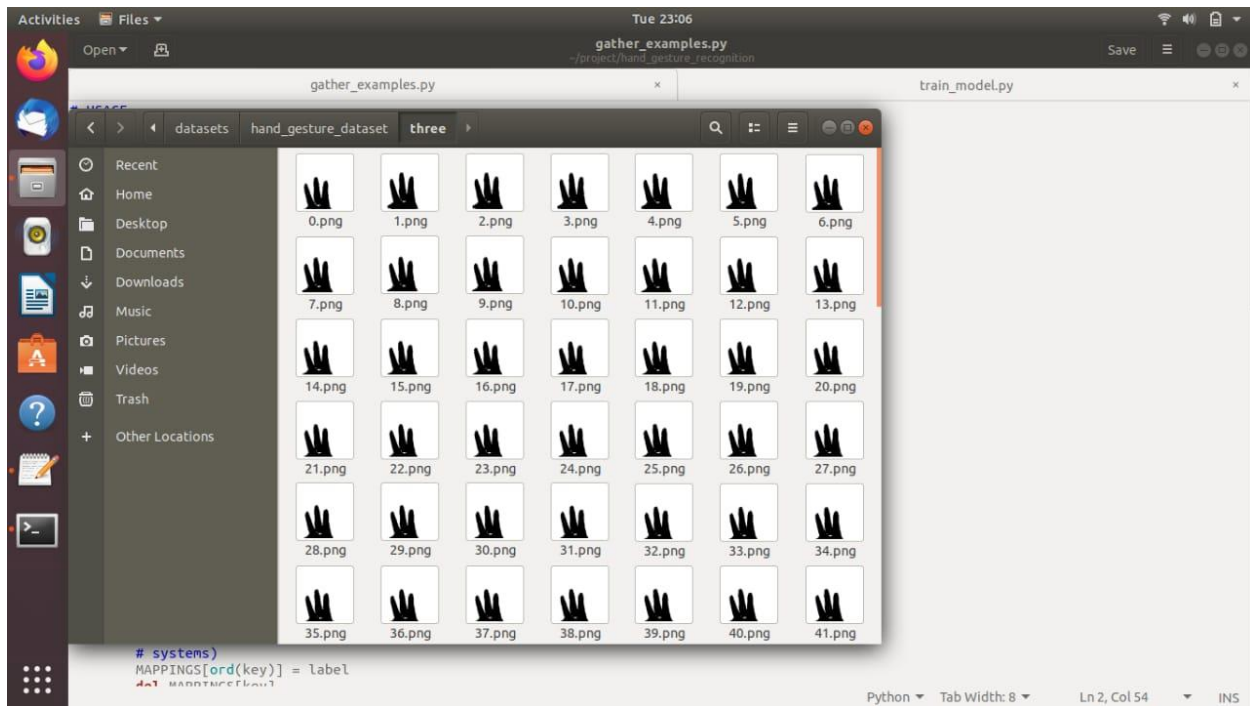


Figure 5.1: Hand gesture three

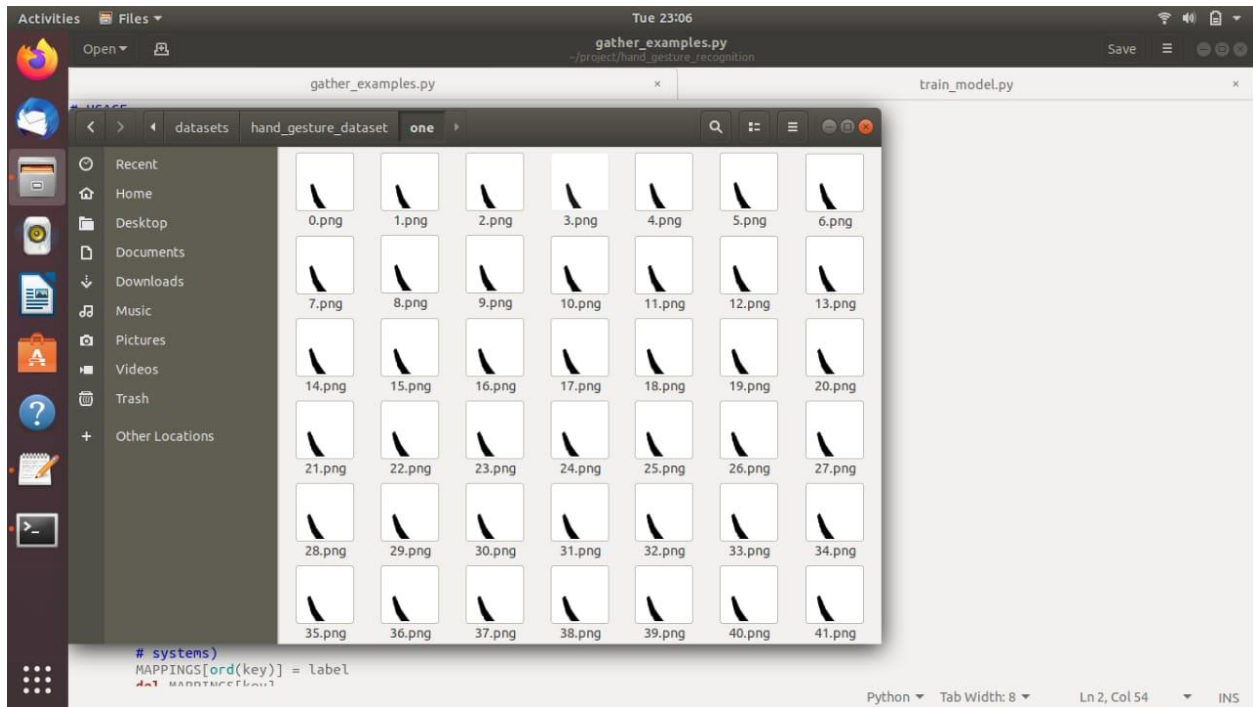
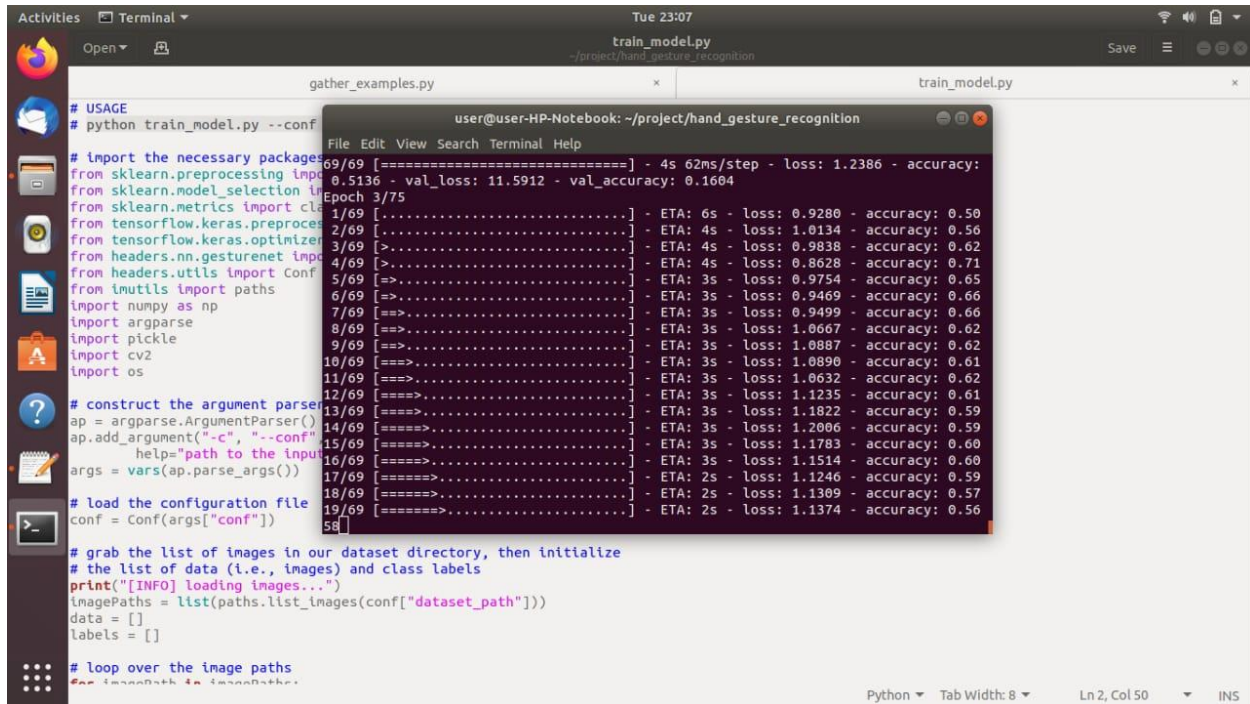


Figure 5.2: Hand gesture one



Figure 5.3: Hand gesture four

As the dataset collection is over, these datasets are trained for gesture recognition. The command for training is run in the terminal that can be seen in the following figure.



```

# USAGE
# python train_model.py --conf conf

# import the necessary packages
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split
from sklearn.metrics import classification_report
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.optimizers import Adam
from headers.nn.gestureenet import GestureNet
from headers.utils import Conf
from imutils import paths
import numpy as np
import argparse
import pickle
import cv2
import os

# construct the argument parser
ap = argparse.ArgumentParser()
ap.add_argument("-c", "--conf",
    help="path to the input configuration file")
args = vars(ap.parse_args())

# load the configuration file
conf = Conf(args["conf"])

# grab the list of images in our dataset directory, then initialize
# the list of data (i.e., images) and class labels
print("[INFO] loading images...")
imagePaths = list(paths.list_images(conf["dataset_path"]))
data = []
labels = []

# loop over the image paths
for imagePath in imagePaths:
    image = cv2.imread(imagePath)
    image = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)
    image = image / 255.0
    data.append(image)
    label = conf["label_map"][cv2.imread(imagePath).shape[0]]
    labels.append(label)

# encode the class labels
le = LabelEncoder()
le.fit(labels)
labels = le.transform(labels)

# partition the data into training and testing sets
trainData, testData, trainLabels, testLabels = train_test_split(data, labels,
    stratify=labels, test_size=0.2, random_state=42)

# initialize the data generators
trainDataGenerator = ImageDataGenerator(
    rotation_range=10, width_shift_range=0.1, height_shift_range=0.1,
    shear_range=0.2, zoom_range=0.2, horizontal_flip=True,
    fill_mode="nearest")

testDataGenerator = ImageDataGenerator(
    rotation_range=10, width_shift_range=0.1, height_shift_range=0.1,
    shear_range=0.2, zoom_range=0.2, horizontal_flip=True,
    fill_mode="nearest")

# compile the model
model = GestureNet()
model.compile(optimizer=Adam(), loss="categorical_crossentropy",
    metrics=["accuracy"])

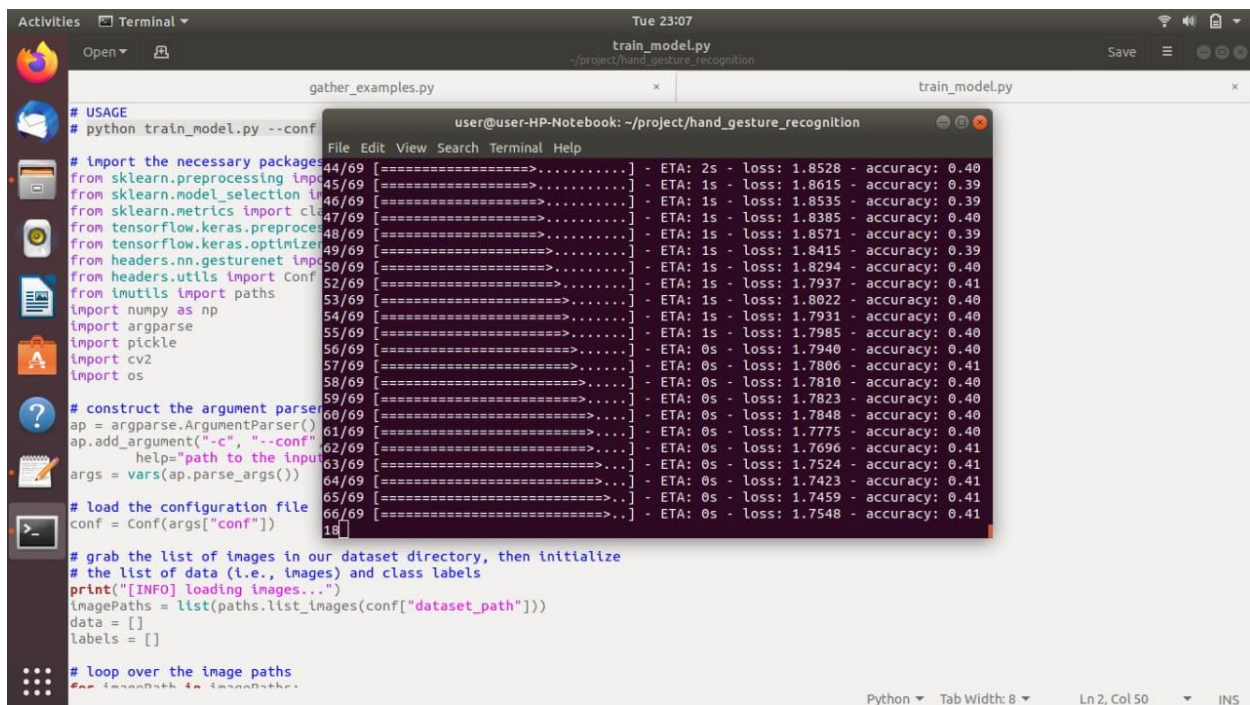
# train the model
trainDataGenerator.fit(trainData)
testDataGenerator.fit(testData)

# train the model
model.fit(trainDataGenerator.flow(trainData, trainLabels, batch_size=32),
    testDataGenerator.flow(testData, testLabels, batch_size=32),
    validation_data=(testData, testLabels), epochs=69)

# evaluate the model
model.evaluate(testDataGenerator.flow(testData, testLabels, batch_size=32))

```

Figure 5.4: Running command in terminal



```

# USAGE
# python train_model.py --conf conf

# import the necessary packages
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split
from sklearn.metrics import classification_report
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.optimizers import Adam
from headers.nn.gestureenet import GestureNet
from headers.utils import Conf
from imutils import paths
import numpy as np
import argparse
import pickle
import cv2
import os

# construct the argument parser
ap = argparse.ArgumentParser()
ap.add_argument("-c", "--conf",
    help="path to the input configuration file")
args = vars(ap.parse_args())

# load the configuration file
conf = Conf(args["conf"])

# grab the list of images in our dataset directory, then initialize
# the list of data (i.e., images) and class labels
print("[INFO] loading images...")
imagePaths = list(paths.list_images(conf["dataset_path"]))
data = []
labels = []

# loop over the image paths
for imagePath in imagePaths:
    image = cv2.imread(imagePath)
    image = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)
    image = image / 255.0
    data.append(image)
    label = conf["label_map"][cv2.imread(imagePath).shape[0]]
    labels.append(label)

# encode the class labels
le = LabelEncoder()
le.fit(labels)
labels = le.transform(labels)

# partition the data into training and testing sets
trainData, testData, trainLabels, testLabels = train_test_split(data, labels,
    stratify=labels, test_size=0.2, random_state=42)

# initialize the data generators
trainDataGenerator = ImageDataGenerator(
    rotation_range=10, width_shift_range=0.1, height_shift_range=0.1,
    shear_range=0.2, zoom_range=0.2, horizontal_flip=True,
    fill_mode="nearest")

testDataGenerator = ImageDataGenerator(
    rotation_range=10, width_shift_range=0.1, height_shift_range=0.1,
    shear_range=0.2, zoom_range=0.2, horizontal_flip=True,
    fill_mode="nearest")

# compile the model
model = GestureNet()
model.compile(optimizer=Adam(), loss="categorical_crossentropy",
    metrics=["accuracy"])

# train the model
trainDataGenerator.fit(trainData)
testDataGenerator.fit(testData)

# train the model
model.fit(trainDataGenerator.flow(trainData, trainLabels, batch_size=32),
    testDataGenerator.flow(testData, testLabels, batch_size=32),
    validation_data=(testData, testLabels), epochs=69)

# evaluate the model
model.evaluate(testDataGenerator.flow(testData, testLabels, batch_size=32))

```

Figure 5.5: Training process

As the training gets completed, an accuracy is predicted that can be seen in the following figure

```

# USAGE
# python train_model.py --conf conf

# Import the necessary packages
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split
from sklearn.metrics import classification_report
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.optimizers import Adam
from headers.nn.gestureenet import GestureNet
from headers.utils import Conf
from imutils import paths
import numpy as np
import argparse
import pickle
import cv2
import os

# construct the argument parser
ap = argparse.ArgumentParser()
ap.add_argument("-c", "--conf",
    help="path to the input configuration file")
args = vars(ap.parse_args())

# load the configuration file
conf = Conf(args["conf"])

# grab the list of images in our dataset directory, then initialize
# the list of data (i.e., images) and class labels
print("[INFO] loading images...")
imagePaths = list(paths.list_images(conf["dataset_path"]))
data = []
labels = []

# loop over the image paths
for imagePath in imagePaths:
    image = cv2.imread(imagePath)
    image = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)
    image = image / 255.0
    data.append(image)
    label = conf["label_map"][cv2.imread(imagePath).shape[0]]
    labels.append(label)

# split the data into training and testing sets
trainData, testData, trainLabels, testLabels = train_test_split(
    data, labels, test_size=0.2, random_state=42)

# create the data generators
trainDataGenerator = ImageDataGenerator(
    rotation_range=10,
    width_shift_range=0.1,
    height_shift_range=0.1,
    shear_range=0.2,
    zoom_range=0.2,
    horizontal_flip=True,
    vertical_flip=True)

testDataGenerator = ImageDataGenerator(
    rotation_range=10,
    width_shift_range=0.1,
    height_shift_range=0.1,
    shear_range=0.2,
    zoom_range=0.2,
    horizontal_flip=True,
    vertical_flip=True)

# compile the model
model = GestureNet()
model.compile(optimizer=Adam(), loss='categorical_crossentropy', metrics=['accuracy'])

# train the model
trainDataGenerator.fit(trainData)
testDataGenerator.fit(testData)

# train the model
model.fit(trainDataGenerator.flow(trainData, trainLabels, batch_size=32),
    testDataGenerator.flow(testData, testLabels, batch_size=32),
    validation_data=(testData, testLabels),
    epochs=100,
    verbose=1)

# evaluate the model
print("[INFO] evaluating network...")
_, accuracy, loss = model.evaluate(testData, testLabels, batch_size=32)
print("0.9183 - val_loss: 1.9580 - val_accuracy: 0.6738")

# save the model
print("[INFO] saving model...")
model.save("gestureenet.h5")

# serialize the label encoder
le = LabelEncoder()
le.fit(labels)
le.save("lb.pickle")

# print the confusion matrix
print(classification_report(testLabels, model.predict(testData)))

```

	precision	recall	f1-score	support
five	0.47	1.00	0.64	33
four	0.08	0.07	0.07	30
ignore	1.00	1.00	1.00	30
one	1.00	0.20	0.33	30
three	1.00	1.00	1.00	34
two	1.00	0.70	0.82	30
accuracy			0.67	187
macro avg	0.76	0.66	0.64	187
weighted avg	0.76	0.67	0.65	187

Figure 5.6: Accuracy generation

After this all dataset collection, dataset training,etc are over, a model file is been generated in the form of pickle file and that can be seen in the figure below

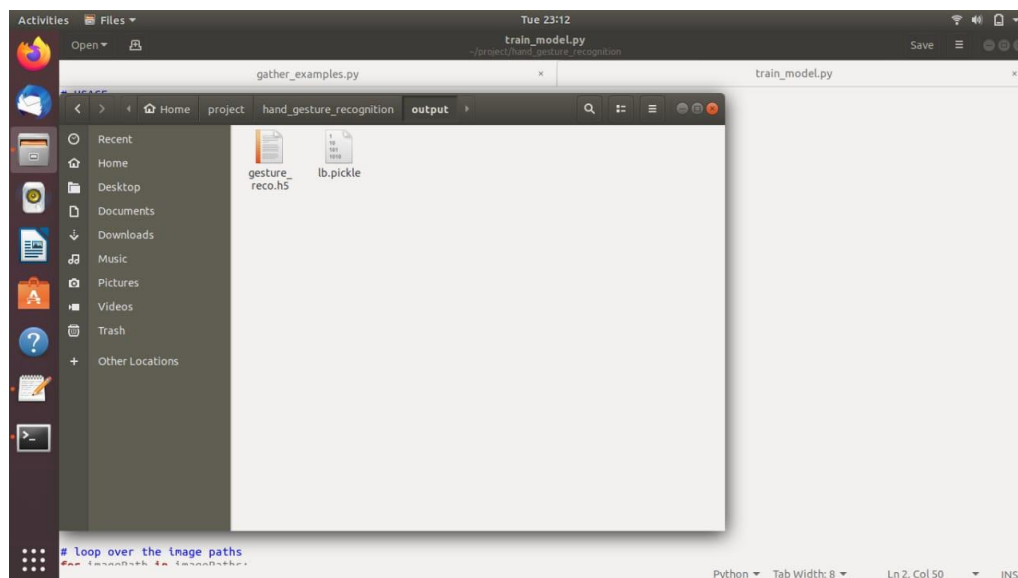
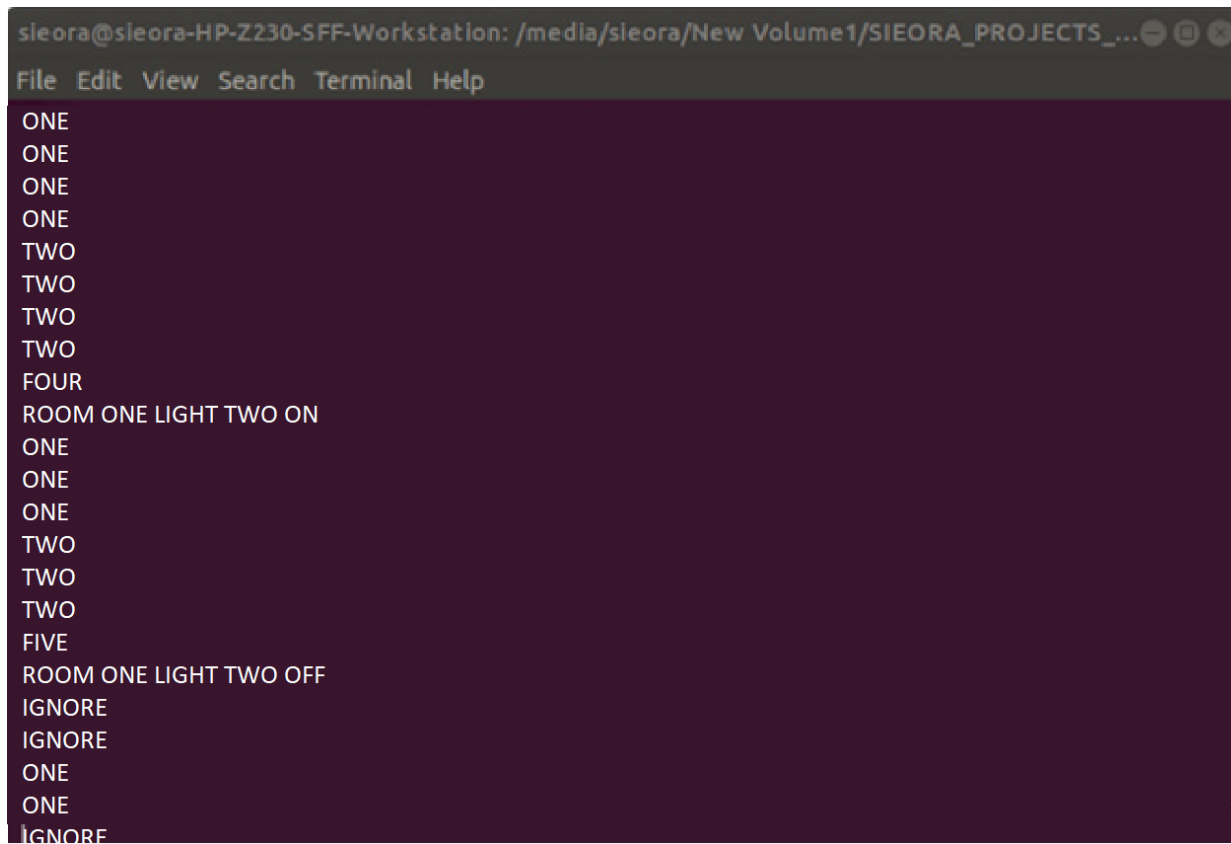


Figure 5.7: Pickle File generation

After this the execution of project takes place where the combination of gestures is done to switch on and off the room's appliance. This can be seen in the below figure.



```
sleora@sleora-HP-Z230-SFF-Workstation: /media/sleora/New Volume1/SIEORA_PROJECTS_...
File Edit View Search Terminal Help
ONE
ONE
ONE
ONE
TWO
TWO
TWO
TWO
FOUR
ROOM ONE LIGHT TWO ON
ONE
ONE
ONE
TWO
TWO
TWO
FIVE
ROOM ONE LIGHT TWO OFF
IGNORE
IGNORE
ONE
ONE
IGNORE
```

Figure 5.8: Automation of Lights

Thus, from the above results it is proved that we achieved home automation with the help of artificial intelligence where there will be no use of internet to achieve automation. By this project, the scope of the project is achieved.

5.2 COMPARISON BETWEEN EXISTING SYSTEM AND PROPOSED SYSTEM

In the existing system, the main disadvantage of this method is the existing system for automation has only hand gesture recognition, making it a very insecure. This method will lead in wrong results and will allow spoofing to take place. This method is not reliable in case of weak internet connection. There may be possible failure in sensor modules making it very inefficient. In the proposed system, the disadvantages of the existing system have been overcome by the successful implementation of a face recognition setup to recognize and receive gestures from authorized persons only. By doing this the efficiency and robustness will increase and the system becomes very secure.

FACTORS	EXISTING SYSTEM	PROPOSED SYSTEM
ALGORITHM	Uses egocentric vision based technologies	Uses Ultralight face detector and gesturenets.
ACCURACY	Accuracy in predicting the hand gestures are very poor	Achieves very good accuracy using the gesturenets.
SECURITY	Unauthorized persons can perform the hand gestures	Only authorized persons can perform the hand gestures, using face detection algorithms.
COST	Involves many sensor modules, making it very expensive for the end user	Uses only an usb camera and no sensors are used in this method. Making it affordable to the common man.

CHAPTER 6

CONCLUSION AND FUTURE WORK

6.1 CONCLUSION

In this project we will be using artificial intelligence which recognizes human gestures without the use of any kind of sensors with the help of a camera and acts accordingly to control any kind of home appliance. By this project physically challenged people can easily switch off the lights and fan or any home appliance. So, we reduce the time required for manual work.

6.2 SUMMARY

In the coming future, we review the application of the hand gesture technology in the automation technology field and it can promote for all type of home appliances with more accuracy. In this field one have more chance to develop or convert this project in many ways. It can be helpful in discovering an automation system without the use of sensors or a mobile app. This is achieved using artificial intelligence technology.

REFERENCE

- [1] Abhijith Punnappurath, Ambasamudram Narayanan Rajagopalan, Senior Member, IEEE, Sima Taheri, Student Member, IEEE, Rama Chellappa, Fellow, IEEE, and Guna Seetharaman, Fellow, IEEE, “ Face Recognition Across Non-Uniform Motion Blur, Illumination, and Pose” VOL. 24, NO. 7, JULY 2015
- [2] Andrea Bandini, Member, IEEE, and Jos´e Zariffa, Senior Member, IEEE, ”Analysis of the hands in egocentric vision:A survey”[Vol 0162-8828,2020]
- [3] An-An Liu*, Ning Xu*, Wei-Zhi Nie*, Yu-Ting Su, and Yong-Dong Zhang,”Multi-Domain & Multi-Task Learning for Human Action Recognition”,IEEE Transactions on Image Processing,VOL.1057-7149,2018.
- [4] Aouaidjia Kamel, Bin Sheng , Po Yang, Ping Li , Ruimin Shen, and David Dagan Feng , Fellow, IEEE,”Deep Convolutional Neural Networks for Human Action Recognition Using Depth Maps and Postures”,IEEE transactions on systems, man, and cybernetics: Systems,VOL.2168-2216,2018.
- [5] Hongsong Wang, and Liang Wang, Senior Member, IEEE,”Beyond Joints: Learning Representations from Primitive Geometries for Skeleton-based Action Recognition and Detection”,IEEE Transactions on Image Processing,VOL.1057-7149,2018.
- [6] Hossein Rahmani, Ajmal Mian and Mubarak Shah,”Learning a Deep Model for Human Action Recognition from Novel Viewpoints”,IEEE Transactions on Pattern Analysis and Machine Intelligence ,VOL.0162-8828,2018.
- [7] Jian Cheng, Haijun Liu, Feng Wang, Hongsheng Li, and Ce Zhu, Senior Member, IEEE,”Silhouette Analysis for Human Action Recognition Based on Supervised Temporal t-SNE and Incremental Learning”,IEEE transactions on image processing, Vol. 24, No. 10, October 2016.

[8] Jiwen Lu, Member, IEEE, Venice Erin Liong, Xiuzhuang Zhou, Member, IEEE, and Jie Zhou, Senior Member, IEEE, "Learning Compact Binary Face Descriptor for Face Recognition"[Vol 0162-8828,2015]

[9] Nour El Din Elmadany, Student Member, IEEE, Yifeng He, Member, IEEE, and Ling Guan, Fellow, IEEE, "Information Fusion for Human Action Recognition via Biset/Multiset Globality Locality Preserving Canonical Correlation Analysis", IEEE Transactions on Image Processing, VOL.1057-7149,2018.

[10] Ping Wei, Member, IEEE, Hongbin Sun, Member, IEEE, and Nanning Zheng, Fellow, IEEE, "Learning Composite Latent Structures for 3D Human Action Representation and Recognition", VOL.1520-9210,2018.

[11] Qiang Nie, Jiangliu Wang, Xin Wang, Yunhui Liu, Fellow, IEEE, "View-invariant Human Action Recognition Based on A 3D Bio-constrained Skeleton Model", IEEE Transactions on Image Processing, Vol. 14, No. 8, August 2019.

[12] Renliang Weng, Jiwen Lu, Senior Member, IEEE, and Yap-Peng Tan, Senior Member, IEEE, "Robust Point Set Matching for Partial Face Recognition" [Vol 1057-7149,2016]

[13] Sid Ahmed Walid Talha¹, Mounir Hammouche^{1,2}, Enjie Ghorbel^{1,3}, Anthony Fleury^{*1}, Member, IEEE,, Sébastien Ambellouiss, "Features and Classification Schemes for View-Invariant and Real-Time Human Action Recognition", IEEE Transactions on Cognitive and Developmental Systems, VOL.2379-8920,2018.

[14] Sunpill Choi, Seongwook Park, and Hoi-Jun Yoo, "A Memory-efficient Hand Segmentation Architecture for Hand Gesture Recognition in Low-power Mobile Devices" [Vol 1057-7149, 2017]

[15] Zhigang Tu, Member, IEEE, Hongyan Li, Dejun Zhang, Member, IEEE, Justin Dauwels, Senior Member, IEEE Baoxin Li, Senior Member, IEEE and Junsong Yuan,

Senior Member, IEEE ,“Action-Stage Emphasized Spatio-Temporal VLAD for Video Action Recognition”, IEEE Transactions on Image Processing,VOL.1057-7149,2018.

APPENDIX

```
# USAGE

# python train_model.py --conf config/config.json

# construct the argument parser and parse the arguments
ap = argparse.ArgumentParser()
ap.add_argument("-c", "--conf", required=True,
                help="path to the input configuration file")
args = vars(ap.parse_args())

# load the configuration file
conf = Conf(args["conf"])

for imagePath in imagePaths:
    # extract the class label from the filename
    label = imagePath.split(os.path.sep)[-2]

    # load the image, convert it to grayscale, and resize it to be a
    # fixed 64x64 pixels, ignoring aspect ratio
    image = cv2.imread(imagePath)
    image = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
    image = cv2.resize(image, (64, 64))

    # update the data and labels lists, respectively
```

```

data.append(image)

labels.append(label)


# convert the data into a NumPy array, then preprocess it by scaling
# all pixel intensities to the range [0, 1]
data = np.array(data, dtype="float") / 255.0


# reshape the data matrix so that it explicitly includes a channel
# dimension

# the data for training and the remaining 25% for testing
(trainX, testX, trainY, testY) = train_test_split(data, labels,
                                                  test_size=0.25, stratify=labels, random_state=42)


# construct the training image generator for data augmentation
aug = ImageDataGenerator(rotation_range=20, zoom_range=0.15,
                          width_shift_range=0.2, height_shift_range=0.2, shear_range=0.15,
                          horizontal_flip=True, fill_mode="nearest")


# initialize our gesture recognition CNN and compile it
model = GestureNet.build(64, 64, 1, len(lb.classes_))

opt = Adam(lr=conf["init_lr"],
           decay=conf["init_lr"] / conf["num_epochs"])

model.compile(loss="categorical_crossentropy", optimizer=opt,

```

```

        metrics=["accuracy"])

# train the network

H = model.fit_generator(
    aug.flow(trainX, trainY, batch_size=conf["bs"]),
    validation_data=(testX, testY),
    steps_per_epoch=len(trainX) // conf["bs"],
    epochs=conf["num_epochs"])

# evaluate the network

print("[INFO] evaluating network...")

predictions = model.predict(testX, batch_size=conf["bs"])

print(classification_report(testY.argmax(axis=1),
    predictions.argmax(axis=1), target_names=lb.classes_))

print("[INFO] serializing label encoder...")

f = open(str(conf["lb_path"]), "wb")

f.write(pickle.dumps(lb))

f.close()

# USAGE

# python recognize.py --conf config/config.json

# import the necessary packages

from pyimagesearch.utils import Conf

```

```
#from pyimagesearch.notifications import TwilioNotifier

from imutils.video import VideoStream

from imutils import paths

from threading import Thread

from tensorflow.keras.preprocessing.image import img_to_array

from tensorflow.keras.models import load_model

from datetime import datetime

from datetime import date

import RPi.GPIO as GPIO

import numpy as np

import argparse

import imutils

import pickle

import time

import cv2

import os


# set the Pi Traffic Light GPIO pins

red = 9

yellow = 10

green = 11


# setup the Pi Traffic Light
```

```
GPIO.setmode(GPIO.BCM)
```

```
GPIO.setup(red, GPIO.OUT)
```

```
GPIO.setup(yellow, GPIO.OUT)
```

```
GPIO.setup(green, GPIO.OUT)
```

```
def correct_passcode(p):
```

```
    # actuate a lock or (in our case) turn on the green light
```

```
    GPIO.output(green, True)
```

```
    # print status and play the correct sound file
```

```
    print("[INFO] everything is okay :-)")
```

```
    play_sound(p)
```

```
def incorrect_passcode(p, tn):
```

```
    # turn on the red light
```

```
    GPIO.output(red, True)
```

```
    # print status and play the incorrect sound file
```

```
    print("[INFO] security breach!")
```

```
    play_sound(p)
```

```
    # alert the homeowner
```

```
    hhmmss = (datetime.now()).strftime("%I:%M%p")
```

```

today = date.today().strftime("%A, %B %d %Y")

msg = "An incorrect passcode was entered at " \
 "{} on {} at {}".format(conf["address_id"], today, hhmmss)

tn.send(msg)

```

```

def reset_lights():

    # turn off the lights

    GPIO.output(red, False)

    GPIO.output(yellow, False)

    GPIO.output(green, False)

```

```

def play_sound(p):

    # construct the command to play a sound, then execute the command

    command = "aplay {}".format(p)

    os.system(command)

    print(command)

```

```

# construct the argument parser and parse the arguments

ap = argparse.ArgumentParser()

ap.add_argument("-c", "--conf", required=True,

    help="path to the input configuration file")

args = vars(ap.parse_args())

```

```

# load the configuration file and initialize the Twilio notifier

conf = Conf(args["conf"])

#tn = TwilioNotifier(conf)


# grab the paths to gesture icon images and then initialize the icons

# dictionary where the key is the gesture name (derived from the image
# path) and the key is the actual icon image

print("[INFO] loading icons...")

imagePaths = paths.list_images(conf["assets_path"])

icons = {}


# loop over the image paths
for imagePath in imagePaths:

    # extract the gesture name (label) the icon represents from the
    # filename, load the icon, and then update the icons dictionary

    label = imagePath.split(os.path.sep)[-1].split(".")[0]

    icon = cv2.imread(imagePath)

    icons[label] = icon


# grab the top-left and and bottom-right (x, y)-coordinates for the
# gesture capture area

TOP_LEFT = tuple(conf["top_left"])

BOT_RIGHT = tuple(conf["bot_right"])

```

```

# load the trained gesture recognizer model and the label binarizer

print("[INFO] loading model...")

model = load_model(str(conf["model_path"]))

lb = pickle.loads(open(str(conf["lb_path"]), "rb").read())


# start the video stream thread

print("[INFO] starting video stream thread...")

vs = VideoStream(src=0).start()

#vs = VideoStream(usePiCamera=True).start()

time.sleep(2.0)


# initialize the current gesture, a bookkeeping variable used to keep
# track of the number of consecutive frames a given gesture has been
# classified as

currentGesture = [None, 0]


# initialize the list of input gestures recognized from the user
# along with the timestamp of when all four gestures were entered

gestures = []

enteredTS = None


# initialize two booleans used to indicate (1) whether or not the

```

```

# alarm has been raised and (2) if the correct pass code was entered

alarm = False

correct = False


# loop over frames from the video stream
while True:

    # grab the frame from the threaded video file stream and grab the
    # current timestamp
    frame = vs.read()
    timestamp = datetime.now()


    # resize the frame and then flip it horizontally
    frame = imutils.resize(frame, width=300)
    frame = cv2.flip(frame, 1)


    # clone the original frame and then draw the gesture capture area
    clone = frame.copy()
    cv2.rectangle(clone, TOP_LEFT, BOT_RIGHT, (0, 0, 255), 2)


    # only perform hand gesture classification if the current gestures
    # list is not already full
    # if len(gestures) < 4:

        # extract the hand gesture capture ROI from the frame, convert

```

```

    # the ROI to grayscale, and then threshold it to reveal a
    # binary mask of the hand

roi = frame[TOP_LEFT[1]:BOT_RIGHT[1], TOP_LEFT[0]:BOT_RIGHT[0]]
roi = cv2.cvtColor(roi, cv2.COLOR_BGR2GRAY)
roi = cv2.threshold(roi, 75, 255, cv2.THRESH_BINARY)[1]
visROI = roi.copy()

    # now that we have the hand region we need to resize it to be
    # the same dimensions as what our model was trained on, scale
    # it to the range [0, 1], and prepare it for classification

roi = cv2.resize(roi, (64, 64))
roi = roi.astype("float") / 255.0
roi = img_to_array(roi)
roi = np.expand_dims(roi, axis=0)

# do a bit of cleanup
cv2.destroyAllWindows()
vs.stop()

```