DEAF BLIND COMMUNICATION

MINI PROJECT

Submitted by

ARUNVENKATESH M
(Reg.No. 720722208002)

in partial fulfillment for the award of the degree

of

MASTER OF ENGINEERING

in

EMBEDDED SYSTEMS

HINDUSTHAN COLLEGE OF ENGINEERING AND TECHOLOGY
Approved by AICTE, New Delhi, Accredited with 'A' Grade by NAAC
(An Autonomous Institution, Affiliated to Anna University, Chennai)
COIMBATORE - 641032

AUGUST 2023

HINDUSTHAN COLLEGE OF ENGINEERING AND TECHOLOGY

Approved by AICTE, New Delhi, Accredited with 'A' Grade by NAAC
(An Autonomous Institution, Affiliated to Anna University, Chennai)

BONAFIDE CERTIFICATE

Certified that this project report **"DEAF BLIND COMMUNICATION"** is the bonafide work of "**ARUNVENKATESH M**" who carried out the project work under my supervision.

Signature of the Supervisor

SUPERVISOR
Dr SEKAR K, M.E., Ph.D.,
Professor,
Department of Electrical and Electronics Engineering,
Hindusthan College of Engineering And Technology,
Coimbatore – 32.

Signature of the Head of the Department

HEAD OF DEPARTMENT
Dr.N.P.ANANTHAMOORTHY, M.E.,Ph.D,
Professor & Head
Department of Electrical and Electronics Engineering,
Hindusthan College of Engineering and Technology,
Coimbatore – 32.

Submitted for the Anna university Examination project Viva-voce examination conducted on …….

INTERNAL EXAMINER **EXTERNAL EXAMINER**

ACKNOWLEDGEMENT

We express our sincere thanks to **Hindusthan Educational and Charitable Trust** for providing us with the necessary facilities to bring out the project successfully. We feel grateful to our thanks to our **Chairmen Shri.T.S.R. KHANNAIYANN**, and **Smt. SARASUWATHI KHANNAIYANN,** HICET for all their support and ray of strengthening hope extended.

I extend my thanks to my **CEO Dr.K.KARUNAKARAN, Ph.D.,** for his constant support and motivation.

I am grateful to our **Principal Dr JAYA J, M.Tech., Ph.D.,** for her invaluable support in enabling us to come up with this project.

At this moment we take this opportunity to convey our deepest regards and sincere thanks to **Dr.N.P. ANANTHAMOORTHY, M.E., Ph.D., Head of the Department**, Electrical and Electronics Engineering for the technical guidance.

We are highly indebted to our **Project Supervisor Dr SEKAR K, M.E., Ph.D.,** Professor, Electrical and Electronics Engineering Department, for his cooperation and valuable guidance from the beginning of the project.

We express our grateful thanks to our **Department Project Co-Ordinator Dr MATHAN K, M.E., Ph.D.,** Associate Professor, Electrical and Electronics Engineering Department, for his timely suggestions and encouragement throughout the project.

We deeply express our gratitude to all the faculty members of the Department of Electrical and Electronics Engineering, for their encouragement which we received throughout the semester.

We thank all our non-teaching staffs, parents, friends and almighty god without whose support and blessings, we would never have completed this project.

TABLE OF CONTENTS

ABSTRACT

Image Processing includes changing the nature of an image in order to improve its pictorial information for human interpretation, for autonomous machine perception. Digital image processing is a subset of the electronic domain wherein the image is converted to an array of small integers, called pixels, representing a physical quantity such as scene radiance, stored in a digital memory, and processed by computer or other digital hardware. Interest in digital image processing methods stems from two principals' applications areas: improvement of pictorial information for human interpretation; and processing of image data for storage, transmission, and representation for autonomous machine perception. Edges characterize boundaries and edge detection is one of the most difficult tasks in image processing hence it is a problem of fundamental importance in image processing. In this paper investigates different steps of digital image processing. Like, a high-speed non-linear Adaptive median filter implementation is presented. Then Adaptive Median Filter solves the dual purpose of removing the impulse noise from the image and reducing distortion in the image.

LIST OF FIGURES:

CHAPTER 1

1.1 INTRODUCTION

Deaf-blindness is a combination of vision and hearing loss. These two senses can be completely or partially affected in impaired people where they face difficulty to communicate in everyday life. This is also known as Dual-sensory loss or multi-sensory impairment. There are 1.5 million people in the world who are suffering from this impairment i.e., 9 out of every 10,000 people. In few cases, people are born deafblind but majority of them loose their senses in latter part of their life. Some of the disorder which causes auditory and visual impairment in people is age-related hearing loss but disorders such as cerebral palsy, a problem with the brain and nervous system that mainly affects movement and co-ordination is due to abnormal brain even before birth. Down syndrome and Usher syndrome are few of those genetic conditions where a baby is infected in its womb. A person who has hearing and visual impairment encounters a unique experience of the world. As mentioned earlier there are numerous ways in which a specially impaired person can communicate. Few among them are sign language, tactile sign language, tadoma, print on palm, speech reading, Braille etc. Based on braille system, there are innumerable devices which are developed such as deaf blind communicator, braille notetaker, screen braille communicator, TTY (tele typewriter) etc. Most of these devices are expensive, not compact and few of them are outdated. There are professionals known as Support Service Providers (SSPs)who are skilled at tactile sign, close up visual sign and other communication strategies which acts as a translator. They are capable of assisting complex or lengthy translations. But these SSPs are difficult to locate and expensive to hire. The input is given as a captured image which is encoded in three vibrating modules. To encode the image captured, Convolutional Neural Network algorithms and OpenCV library is used for computation of the data received. These vibrating modules are placed inside an armband which can be wrapped easily around the arm.

1.2 MOTIVATION

Through this project we are working for the welfare of disabled people with knowledge and technology we have. There is no better motivation than helping the specially challenged people. The driving force for us to choose this project is that it involves a combination of various fields to build a model that can be applied effectively to make their life easy and better. Deaf-blind people use many different ways to communicate such as sign language, tactile sign language, tracking etc.

There are hand signs and gestures with the help of which a normal person can communicate with the deaf-blind people. But there are some drawbacks which brings difficulty for them to communicate. Deaf-blindness is a multi-sensory impairment. 0.2% of the world's population is living with severe deaf-blindness and 2% of world's population is living with milder forms deafblindness. A person who is deaf-blind has a unique experience of world. Deaf-blind people are sensitive to touch, because of which we have used vibrating motors as output devices to communicate with deaf-blind people. Deaf-blind people use many different ways to communicate such as sign language, tactile sign language, tracking, tactile finger spelling, print on palm, todama, brille, speech reading, etc. There are devices for deafblind communication like Screen Braille Communicator, Braille Notetaker, Deaf-Blind Communicator, Tyle, etc. All the devices which is available for deaf-blind people are expensive where these people cannot afford. Most of the devices are not portable. There are hand signs and gestures with the help of which a normal person can communicate with the deaf-blind people. But there are some drawbacks which brings difficulty for them to communicate. Our project aim for the communication through the hand gestures from a distance, which is cost-efficient and user-friendly. This model ensures that the communication is contactless and include internet to communicate with the deaf-blind person.

1.3 PROBLEM STATEMENT

The aim of our project is to provide assistance for deaf-blind people to communicate with people using hand gestures. As we know deaf-blind communicate through tactile language. Deaf-blind people are sensitive to touch, sense of touch is used to communicate with them.

Until the 1970s, most people who were deaf and blind lived lives of isolation. As professionals became aware of this population, attempts were made to serve deafblind people by creating manual alphabets or modifying sign languages used by deaf-sighted people. Several methods of deafblind communication have been developed like Handover-hand, Tracking, Tactile fingerspelling, Lorm, Tracing, Braille signing, additionally, simple ways of responding, such as a tap for 'yes' or a rubbing motion for 'no', may be included. In Japan, a system developed by a deafblind woman is in use to represent the five vowels and five major consonants of the Japanese language on the fingers, where the signer 'types' onto a table and the receiver places their hands on top to 'listen'. The challenging part was

communicating with children or babies born deaf and blind who had not had an opportunity to learn a natural (spoken or signed) language. Some efforts made for those conditions are Co-active signing, which is the sender moves and manipulates the hands and arms of the Deafblind person to form sign shapes, or fingerspell words. This is often used with deafblind children to teach them signs, and with people with an intellectual disability and On-body signing, which is the body of the person who is deafblind is used to complete the sign formation with another person. E.g.: chin, palm, chest. Often used with people who also have an intellectual disability. As the decades progressed, deafblind people began to form communities where tactile language were born. Just as deaf people brought together in communities first used invented forms of spoken language and then created their own natural languages which suited the lives of deaf-sighted people. Deafblind people in communities first used modified forms of visual language and are now creating their own natural tactile languages. Through this project a normal person or even a dumb person can make gestures Infront of the camera embedded on the deaf-blind person or a webcam (if you are communicating from a distance). The model takes the input data in the form of image and predicts the gesture made by the person next to camera as hello, yes, no, where, etc and assigns the specific code to it. Then the vibrating modules vibrates in the pattern assigned to the gesture made. Our model is cost-efficient and user-friendly, and ensures that you can even communicate with the deaf-blind person at a distance. The battery, IoT device, vibrating modules are embedded in an armband, which is compact and can be wrapped around the arm. As our device communicate through HTTP protocols, The device requires active internet connection for its working. We have initially included 8 signs to communicate which include hello, yes, no, where, stop, food, talk, and none for no gesture.

1.4 OBJECTIVES

A person may be totally deaf, totally blind, low vision, hard of hearing, or any combination of these four states. Some deaf-blind individuals would have been deaf first and be familiar with alternative techniques which focus on vision, while others may have been blind first and are therefore more comfortable with solutions that rely on hearing. (Deaf-blind people may also have other advantages and disadvantages that must be considered when choosing a solution, such as poor reading skills or a higher level of tech savviness.). Non-technical solutions include a communication card which is a simple Braille/print or large print card, usually laminated, which asks a specific question or requests assistance concerning a

specific item or task, inexpensively, limited by their linear nature. Support Service Providers are the professionals who have been trained in skills such as tactile sign, close up visual sign, and other communication strategies like flexibility in receiving information, capable of assisting with very complex or lengthy transactions. SSPs can be difficult to locate and expensive to hire. Many deaf-blind people use computers to meet longer-distance communication needs by screen-access software, Braille display, or magnification package. Interpretype DBCS 2.0 package sold by Freedom Scientific. This package consists of a pair of laptop computers, a carrying case, a Focus 40 Blue Braille display, JAWS (installed on one computer), and the Interpretype software. The deaf-blind user must carry around a pair of computers and a Braille display. Electronic Braille notetakers replicate the basic functionality of a personal data assistant (PDA) by providing functions such as a calendar, an address book, a calculator, basic note taking, and book reading. Also offer instant messaging, email, basic web browsing, and other online services. These devices are highly portable. PAC Mate Bluetooth enabled computer. In the case of the PAC Mate and a computer, the computer is designated as a "server" and the PAC Mate is designated the "client". The PAC Mate user is able to send messages between the PAC Mate and the computer. This software allows the user to save conversations for later review by which a user can create pre-set messages in order to send frequently used questions or statements easily and quickly. But the PAC Mate is an older technology, missing many of the portability and hardware benefits of newer tools. The Deaf-Blind Communicator (DBC) is a package built by HumanWare intended to offer a simple solution to the need for portable face-to-face communications. It consists of two pieces of hardware: an older Windows Mobile 6 cell phone with a full QWERTY keyboard and a BrailleNote mPower. The deaf-blind user reads messages on the mPower's Braille display and can reply to the sighted user by typing on the notetaker's keyboard. While the sighted participant uses the screen and keyboard of the cell phone to read and write messages to the deaf-blind person. The notetaker can be left in Deaf-blind Communicator mode, which offers only the basic deaf-blind communication tools, or it can be set to a more advanced user mode, which allows the user to take advantage of the full range of software available to users of the BrailleNote, such as basic Internet browsing, email, notetaking, book reading, and personal management functions. DBC is unfortunately based heavily on outdated technology. A user can no longer purchase a Braille Note mPower from HumanWare, and Windows Mobile 6 has not been relevant or widely available for several years. Since this system offers a truly portable design and a simple setup

compared with many of the other tools on this list, it would be an excellent choice for users if the technology were not already so outdated.

HIMS offers its deaf-blind communications solutions. HIMS has chosen to offer the basic notetaker with a couple of relatively inexpensive add-ons and no specialty software. The Braille Sense U2 and U2 Mini are the most recent Braille notetakers to come out of HIMS and are equipped with all of the software that a user would come to expect from a recent notetaker, but they also have some unique features that make them compelling options for deaf-blind users. First, each of these devices is equipped with a vibration motor in order to provide tactile feedback, which can replace system sounds for all important notifications on the device. Second, the Braille Sense with a thirty-two-cell display and Perkins-style keyboard includes a small, one-line LCD, which can be flipped to face away from the user, thus allowing a sighted user to view what is being written on the notetaker. Unfortunately, the very small screen of the notetaker and the close proximity of the built-in LCD screen on the Perkins style U2 can make reading the conversation less comfortable for the sighted participant. Taking all the parameters of the above devices into considerations which didn't meet the desired expectations such as simple, portable, cost effective etc, we have designed a model which meets the expectations as required.

1.5 SUMMARY

Gesture recognition through Deep Learning can be serviceable to the people who are deaf as well as blind to communicate with the normal people and the dumb people through the sense of touch (vibrating motors). In the present era of Artificial Intelligence and Machine Learning, a device which takes the input in the form of images (hand gestures) and predicts the gesture made based on the Trained Model can be used for the welfare of the Physically Impaired people who can't speak and hear the world around them. This method has no ill effects on the user as well as the person who is trying to communicate with him. The device is less expensive and compact (embedded in armband), which makes it universal product for the people of all standards. All it takes a fully charged battery and an active internet connection

CHAPTER-2

LITERATURE SURVEY

2.1 INTRODUCTION

To support and care for the deafblind ppl they try to communicate in alternate methods such as hand signing, braille. More than 70 causes of deaf-blindness were identified in 2019 National deaf-blind child count. The most common complications of prematurity and CHARGE syndrome, with each causing approximately 10% of cases. 0.2% of the world's population is living with severe deaf-blindness and 2% of the world population is living with milder forms of deaf-blindness. For a young child who is deaf-blind, the world is much narrower. If the child is profoundly deaf and totally blind, his or her experience of the world extends only as far as the fingertips can reach. There are 9 out of 10,000 people who are deaf as well as blind in the world's population. We are utilizing the concept of deep learning as it's an integral part of machine learning which is equivalent to the human brain in its functions and designs. The neural networks are used in training the model to recognise images, voice, text and so on. The working of neural network can be explained using 3 layers namely input layer, output layer and in between there is at least one hidden layer present. To build a neural network, the input layer receives large amount of data which can be in the form of an image. In the hidden layers by applying appropriate activation functions, the predicted output can be generated. The data is processed by performing complex computation and then the feature extraction is carried out. After processing the data, the values are passed onto the outer layer. TensorFlow is used to build an image classification model. 2.2 LITERATURE SURVEY Braille is a tactile writing system used by people who are visually impaired. It is traditionally written with embossed paper. Braille users can read computer screens and other electronic supports using refreshable braille displays. They can write braille with the original slate and stylus or type it on a braille writer, such as a portable braille notetaker or computer that prints with a braille embosser. Braille equipment includes a variety of multipurpose devices, which enhance access to distance communication. Some can be used as stand-alone devices connected via Wi-Fi, while others are paired with a mobile device to provide tactile access to e-mail, text messaging, and other modern communication resources. To receive Braille equipment, an eligible consumer must be proficient in Braille and must have access to the Internet or cellular telephone service. Tele braille does not have a computer communications

modem, but does have a TTY (TDD) modem. It was designed as a TTY for deaf-blind people and is also useful for face-to-face conversation. It has two components: The sighted component is a modified SuperCom TTY device. It has a qwerty keyboard and a single-line LED display. The display is regular size and is not particularly suited to people with low vision. The SuperCom TTY can be connected directly to the telephone line using a conventional telephone jack or the telephone receiver can be coupled to the SuperCom on a cradle on top of the device. Text flows past the display in a continuous stream, like tickertape. The SuperCom is connected to the Braille portion of the device by a cable that is about 2 ft (0.6 m) long. The Braille display is about 15 characters in width, although a knockout allows additional characters to be installed, at considerable additional cost. The Telebraille is able to communicate in ASCII mode, but is not compatible with conventional computer modems. There is what looks like a RS-232 socket on the back of the Braille component, but the instructions for the Telebraille state that this jack is for "future use" and that no computer devices should be attached to it.

2.2 METHODOLOGY

2.2.1 DESCRIPTION

Deep learning application demands a higher level of computation for the training process which seems to be complicated. It involves numerous iterative processes, matrix multiplications, mathematical calculations and consumes more time due to huge data size. Neural network plays one of the important roles in deep learning. Through this project, deep learning and the concept of artificial neural network are used to implement a model which can convert the hand gesture made into move name (basic words such as hello, yes, no etc). The overview of the system is shown in the form of block diagram in Figure 2.1.

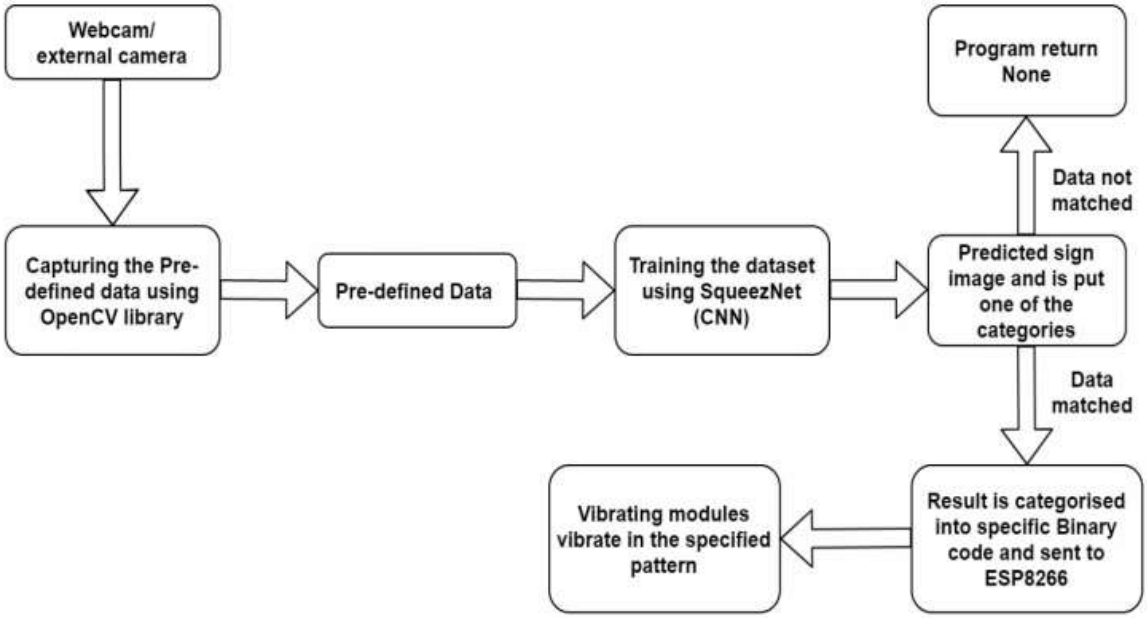


Figure 2.1: Block Diagram

2.3 EXTRACTING KNOWLEDGE FROM EXISTING SYSTEM:

The CNN or convolutional neural networks are the most commonly used algorithms for image classification problems. An image classifier takes a photograph or video as an input and classifies it into one of the possible categories that it was trained to identify. They have applications in various fields like driver less cars, defence, healthcare etc.

The image classifier model is trained to classify different possible categories based on the input in the form of a photograph or a video. There are many algorithms for image classification and in this project, we have used SqueezeNet by DeepScale, one of the most popular algorithms and has a smaller CNN architecture.

Smaller CNN architectures offer at least three advantages:

1. Smaller Convolutional Neural Networks (CNNs) require less communication across servers during distributed training.

2. Smaller CNNs require less bandwidth to export a new model from the cloud to an autonomous car.

3. Smaller CNNs are more feasible to deploy on FPGAs and other hardware with limited memory.

Image recognition process can be explained in 4 steps:

A. Classification of categories

B. Collection of pre-defined data

C. Training of the model

D. Prediction model

2.3.1. Classification of categories:

The application is designed in such a way that it can identify and classify the gestures captured as an input using webcam or an external camera into one of the categories in the data as shown in Figure 2.2. When the model fails to recognise the gesture made by the user or no gesture is made, then it will have another category (say None) to indicate that the user should provide an appropriate input. To overcome false predictions, datasets are captured from different backgrounds in this category.

Figure 2.2: Sign gestures

2.3.2. Collection of pre-defined data:

To capture the pre-defined images, OpenCV library is used. OpenCV library supports capturing of the frame with cv2.VideoCapture(0) function, then the captured images are written or saved using cv2.write() function. cv2.rectangle() is

being used to specify a boundary for the hand gesture in the form of rectangular frame within the main frame as shown in Figure 2.3.

Figure 2.3: Rectangular frame used to specify the boundary for the hand gesture made.

The model is programmed in such a way that the camera (using cv2 library) captures only the rectangular frame. The program manually takes the input for number of images to be captured and the 'label_name' for the particular category of images obtained from the command line arguments. To take number of samples and label_name from the command line argument, sys.argv[] function is used. Here the number of samples means, the number of images to be captured and label name indicates to specify the name of the category that these images belong to as shown in Figure 2.4

```
Usage: python gather_images.py <label_name> <num_samples>
```

Fig 2.4: Arguments for labelling the pre-defined data

The program ends after capturing the specified number of sample images. A sub-folder is created under the main folder (e.g., 'pre-defined data' folder) and stores all the captured images based on the category name which is illustrated in the Figure 2.5.

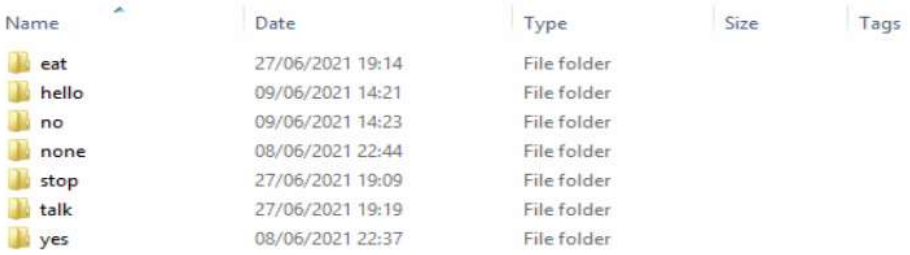


Figure 2.5: The pre-defined images categorised and is stored in these labelled names

2.3.3. Training of the model

In this project, Tensorflow acts as a backend engine where it receives data in the form of multi-dimensional arrays, which are of higher dimensions called tensors. These multidimensional arrays are handy in nature which means it has the capacity of handling large amount of data. The results executed using the Tensorflow code is graphically represented. The major benefit of using Tensorflow is that, it supports both GPUs and CPUs which also has a faster compilation time.

Keras library is used for training process which is simple and powerful Python library that is widely used in deep learning. It has made it really easy to train neural network models. It is required to add few layers to the Convolutional Neural Network (SqueezeNet) which is done using Keras library. The layers are arranged in a linear sequence using Sequential model. We have Added few layers to the existing Convolution Neural network, SqeezeNet imported from keras library. The layers in a model are added as arguments to this constructor. The shape of the input that the SqueezeNet requires should be at least 224x224 (with 3 channels for RGB). In this model, 227x227 is used as the input shape. The number of neurons/nodes in the outer layer is equal to the number of classes that the model is aimed to predict. Deep neural nets with a large number of parameters are very powerful machine learning systems. Overfitting might occur as a problem when the deep neural networks contain large number of parameters. Neural networks which are trained on smaller datasets often tend to over-fit and are less likely to be accurate on new datasets. Large networks are also slow to use, making it difficult to deal with overfitting by combining the predictions of many different large neural nets at test time. Theoretically, the best way to train a model could be to try out as

many different combination of different parameter values and then take an average of those individual results to come up with a generalized result. But this would require a lot of time and computational resources in training the model multiple times with multiple combinations of these parameters. To address this issue, dropout rate is implemented. The main objective of this technique is to drop units randomly from the neural network during the training which prevents the units from getting accustomed more. Dropout is a technique for addressing this problem. The key idea is to randomly drop units (along with their connections) from the neural network during training. This prevents units from coadapting too much. Few units/nodes in a layer (from input layers or hidden layers, but not from the output layer) are 'dropped', due to this the node's incoming and the outgoing connections tends to disappear. Simply put, when this is done multiple times during training, different number of node(s) are dropped from the layer making the layer appear differently in terms of number of nodes and its connections to former layer (roughly simulating multiple models with different layer configurations). This significantly reduces overfitting and gives major improvements over other regularization methods. This in turn reduces overfitting and provides major improvements over other regularization methods. Generally, an acceptable value of dropout rate for a layer is between 0.5-0.8. In this project, 0.5 is found to be the suitable value. Function: ReLU is the activation function being used, which is the most common activation function in neural networks due to its simplicity in its computation. For negative inputs given, the function returns zero and for the positive inputs it returns the value itself. For most of the modern neural networks, ReLU is the default activation function. In CNN, it is a common practice to add a pooling layer after the convolution & activation layer. A pooling layer is added in order to eliminate the minor details. The input image is down sampled or converted to a low resolution which retains only the significant data. Right before the output layer, an activation layer is used for multi class classification. The activation layer used here is the SoftMax layer which provides the plausibility of the input image that is belonging to a particular category. Adam optimizer is being used and a loss function is selected based on the type of the problem. The loss function is chosen based on the type of problem. For e.g.: - for a binary classification problem, [loss='binary_crossentropy'] is better suited and for a multi-class classification problem, [loss='categorical_crossentropy'] is chosen. Whereas for a regression problem, [loss='mse'] could be chosen. Since ours is a multi-class classification problem, we will use categorical_crossentropy. Outer layering for the CNN is done. Sequential: We will use a sequential model, meaning that the layers are

arranged in a linear stack (sequence). The layers in a model are added as arguments to this constructor. Dropout Rate: Neural networks trained on smaller datasets often tend to over-fit and are therefore less likely to be accurate on new data. Theoretically, the best way to train a model could be to try out as many different combination of different parameter values and then take an average of those individual results to come up with a generalized result. But this would require a lot of time and computational resources in training the model multiple times with multiple combinations of these parameters. To get around this, the dropout rate was introduced. Here, some units/nodes in a layer (from input layers or hidden layers, but not from the output layer) are 'dropped' which makes that node's incoming and outgoing connections disappear. Simply put, when this is done multiple times during training, different number of node(s) are dropped from the layer making the layer appear differently in terms of number of nodes and its connections to former layer (roughly simulating multiple models with different layer configurations). A good value for Dropout rate for a layer is between 0.5 – 0.8. In my case, I have found 0.5 to be giving me the best result after trying out few different values. Do note that dropout is used to avoid over fitting. If the model's accuracy is low, this parameter could be avoided. Nodes: Number of neurons/nodes in our output layer is equal to the number of classes we are trying to predict. Input Shape: The input shape that SqueezeNet requires is at least 224 X 224 (with 3 channels for RGB). In this program, we have used 225 X 225. Number of Epochs: The number of epochs is the number of times the entire dataset is passed through the neural network during training. There is no ideal number for this and it depends on the data. In this case, I started with 10, I have used 15. Higher number indicates longer training time. Activation Function: We are using the ReLU activation function, which is the most common activation function that is used in neural networks due to its computational simplicity (among other advantages). This function returns a zero for negative inputs and the value itself for positive inputs. For most of the modern neural networks, ReLU is the default activation function. Other activation functions that are also used are Sigmoid, Tanh etc. Pooling: In CNN, it is a common practice to add a pooling layer after the convolution & activation layer. The input image is down sampled or converted to a low-resolution version in order to keep only the significant details and remove the finer less important details. Softmax: The softmax layer is used in multi-class classification right before the output layer. It gives the likelihood of the input image belonging to a particular category. After training we will store the trained model parameters into a file (gesture-model05_20.h5) which we will be using later

during model testing. Dropout: Deep neural nets with a large number of parameters are very powerful machine learning systems. However, overfitting is a serious problem in such networks. Large networks are also slow to use, making it difficult to deal with overfitting by combining the predictions of many different large neural nets at test time. Dropout is a technique for addressing this problem. The key idea is to randomly drop units (along with their connections) from the neural network during training. This prevents units from coadapting too much. During training, dropout samples from an exponential number of different networks. At test time, it is easy to approximate the effect of averaging the predictions of all these thinned networks by simply using a single unthinned network that has smaller weights. This significantly reduces overfitting and gives major improvements over other regularization methods. We show that dropout improves the performance of neural networks on supervised learning tasks in vision, speech recognition, document classification and computational biology, obtaining state-of-the-art results on many benchmark data sets. Using the pre-defined data, the neural network assigns specific move code to each category. The model is trained to predict the “move name” based on the gesture made. After training the model, we store its parameters into a file name with .h5 extension.

2.3.4. Prediction Model

The trained model classifies the images based on the category and predicts the “move name” on the basis of the move code assigned to the gesture made (Infront of the camera). "load_model” function which is obtained from Keras library is used to load the model. model.predict() function is used to predict the gesture made. Using boltiot library, the predicted “move name” which is in the form of string is returned to IoT device.

CHAPTER – 3

PROPOSED METHODOLOGY:

3.1 INTRODUCTION

To support and care for the deafblind ppl they try to communicate in alternate methods such as hand signing, braille. More than 70 causes of deaf-blindness were identified in 2019 National deaf-blind child count. The most common complications of prematurity and CHARGE syndrome, with each causing

approximately 10% of cases. 0.2% of the world's population is living with severe deaf-blindness and 2% of the world population is living with milder forms of deaf-blindness.

3.2 DESCRIPTION OF PROPOSED MODEL:

ESP8266-BOLT IoT:

It is a microcontroller with built-in Wi-Fi module that is integrated with the TCP/IP protocol stack. We are using the Bolt IoT board in our project as it has its own IoT cloud where the Cloud API controls and monitors the Bolt IoT device over the internet. The Bolt IoT Cloud API gives us the interface for communication between 3rd party systems like mobile app, web server, python programs etc with the Bolt IoT Devices. The Bolt Cloud uses HTTP protocol for communication where HTTP is a TCS/IP based communication and it is used for exchanging of the data over the World Wide Web. The data is exchanged by the HTTP GET and HTTP POST methods. The GET method is used for the receiving of the data and POST method is used for the sending data form the 3rd party systems. The board has 5 DIgital I/O pins for connecting the sensor. The operating voltage of the device is 3.3V which makes it power efficient. The board is in the dimension of 35mm x 35mm. Which makes it compact and portable. The Bolt IoT device is shown in Figure 2.6.

Figure 2.6: ESP8266 Bolt IoT device

3.2.1 WORKING:

Working of ESP8266-BOLT IoT:

As we know, the Python program connects to the Bolt device through the internet. It is done through importing of a library file in the python program. The library “boltiot” is installed and imported in the program. This library helps in connecting the program and the board through the Uniq API Key and Device ID that is assigned to an IoT device. With the help of DigitalWrite command and the pin number, the vibrating motor connected to the board can be controlled. The core program of our project is to identify the type of input that is given to the deaf-blind person. The input is a predefined sign language that have to be conveyed to the disabled person. When the input is given to the camera the received data is analysed and compared with the predefined training data. The result of this is taken as an input for the IoT device. In the program the inputs of the IoT are assigned with different working sequences of the vibrating modules. When the input is given, the vibrating motor vibrates in the particular pattern assigned to them with an intensity of 250 Hz which is considered to be an ideal intensity for sensing the vibration. As we know, the level of sensation or the response for the vibration may vary from person to person and the intensity for which a human being's response ranges from 20 Hz - 1000 Hz. Depending upon the comfort of the user the intensity of the vibration can be changed in the program.

ARMBAND:

We are choosing the armband as our output which is a wearable component as it is compact and portable. The Bolt IoT Device with the battery is placed in one of the compartments of the armband. The vibrating motors are attached to the inner layer of the armband through which they make indirect contact to the skin. The armband we are using comes with the Velcro as shown in Figure 2.8, so by this the person who uses the device can wear the armband according to their own comfort. So the IoT device is switched on and connected to the Wi-Fi of the user's mobile. Then the armband is wrapped around the upper limb of the user. This allows the user to conveniently use the wearables device. As mentioned before the intensity of the vibration can be changed according to the user requirement.

Figure 2.7: Armband where vibrating modules are embedded inside the band

When the input is given, the vibrating modules vibrate in a particular pattern assigned to them with an intensity of 200Hz which is considered to be an ideal intensity for sensing the vibration. The user must be instructed with some demonstration to make them familiar with the message conveyed to them with the variation in the sequence of vibrating modules. These vibrating modules are small in size which has a dimension of 10mm x 2.8mm as shown in the Figure2.8. The overview of the practical model is shown in Figure 2.9.

Figure 2.8: Vibrating module

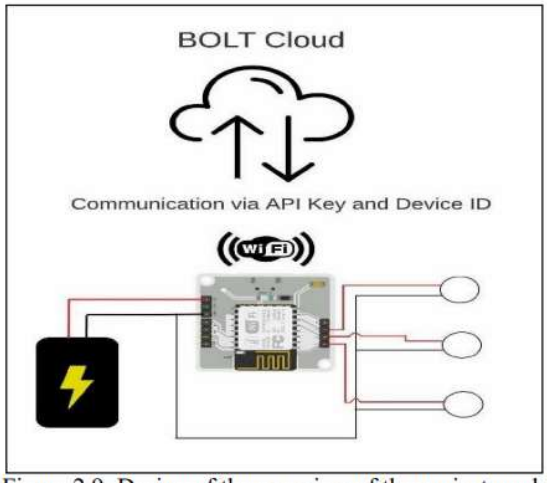
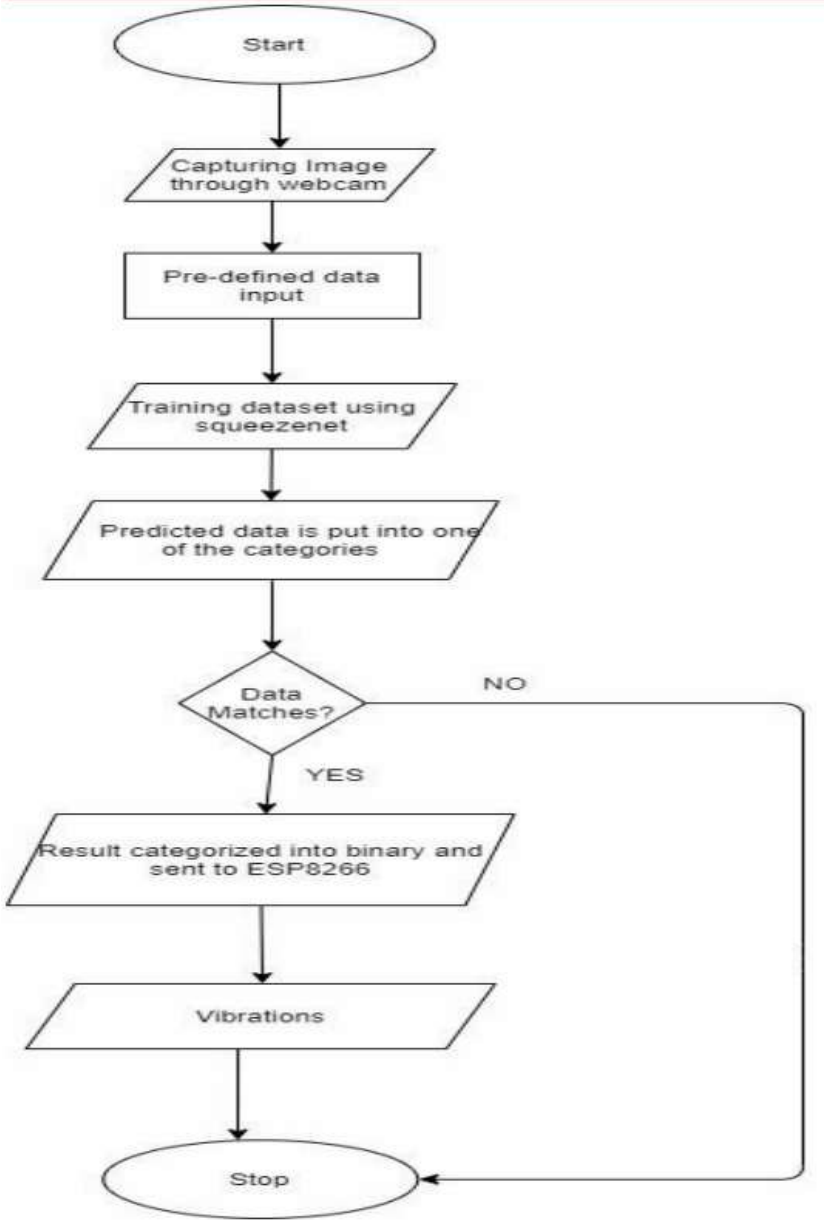


Figure 2.9: Design of the overview of the project model

3.3 FLOW CHART



3.4 CODING

3.4.1 CODE OF THE PROJECT

Code to capture images using OpenCV:

```
desc = “”
Script to gather data images with a particular label.
Usage: python gather_images.py.
<label_name><num_samples><background_name>
The script will collect <num_samples> number of images and store them in its own directory.
 Only the portion of the mage within the box displayed will be captured and stored.
Press 's' to start/pause the image collecting process.
Press 'q' to quit.
'''
import cv2
import os
import sys
def deleteprefiles(file_name):
try:
        img_counter = 0
        for img_counter in range(1, 201):
                os.remove("E:/OpenCV/reference/"+label_name+"/{}.jpg".format(img_counter))
                img_counter += 1
        except: pass
try:
        label_name = sys.argv[1]
        num_samples = int(sys.argv[2])
```

```
        background = sys.argv[3]
except:
        print("Arguments missing.")
        print(desc)
        exit(-1)
IMG_SAVE_PATH = 'E:\OpenCV\Reference'
IMG_CLASS_PATH = os.path.join(IMG_SAVE_PATH, label_name)
try:
        os.mkdir(IMG_SAVE_PATH)
except FileExistsError:
        pass
try:
        os.mkdir(IMG_CLASS_PATH)
except FileExistsError:
        print("{} directory already exists.".format(IMG_CLASS_PATH))
        print("Deleting existing items in this folder")
        #deleteprefiles(label_name)
cap = cv2.VideoCapture(0)
cv2.namedWindow("Capturing...")
start = False
count = 0
while True:
        ret, frame = cap.read()
        if not ret:
                continue
```

```
        if count == num_samples:
        break
cv2.rectangle(frame, (50, 50), (270, 270), (255, 0, 0), 2)
if start:
        roi = frame[50:270, 50:270]
        save_path = os.path.join(IMG_CLASS_PATH, background +
'{}.jpg'.format(count + 1))
        cv2.imwrite(save_path, roi)
        count += 1
        font = cv2.FONT_HERSHEY_SIMPLEX
        cv2.putText(frame, "Collecting {}".format(count),(5, 50), font, 0.7, (0, 255,
255), 2, cv2.LINE_AA)
        cv2.imshow("Collecting images", frame)
        k = cv2.waitKey(10)
        if k == ord('s'):
                start = not start
        if k == ord('q'):
                break
        print("\n{} image(s) saved to {}".format(count, IMG_CLASS_PATH))
        cap.release()
        cv2.destroyAllWindows()
```

3.4.2 CODE TO TRAIN MODEL:

```
import cv2
import numpy as np
from keras_squeezenet import SqueezeNet
from keras.optimizers import Adam
from keras.utils import np_utils
from keras.layers.core import Activation
from keras.layers.core import Dropout
from keras.layers import Convolution2D
from keras.layers import GlobalAveragePooling2D
from keras.models import Sequential
import tensorflow as tf
import os
IMG_SAVE_PATH = 'E:\OpenCV\Reference'
CLASS_MAP = {
        "hello": 0,
        "yes": 1,
        "no": 2,
        "where": 3,
        "food": 4,
        "talk": 5,
        "stop": 6,
        "none": 7
}
NUM_CLASSES = len(CLASS_MAP)
```

```
    def mapper(val):
        return CLASS_MAP[val]
def get_model():
    model = Sequential([
        SqueezeNet(input_shape=(227, 227, 3), include_top=False),
        Dropout(0.5),
        Convolution2D(NUM_CLASSES, (1, 1), padding='valid'),
        Activation('relu'),
        GlobalAveragePooling2D(),
        Activation('softmax')
    ])
    return model
    # load images from the directory
    dataset = []
    for directory in os.listdir(IMG_SAVE_PATH):
        path = os.path.join(IMG_SAVE_PATH, directory)
        if not os.path.isdir(path):
        continue
    for item in os.listdir(path):
        # to make sure no hidden files get in our way
        if item.startswith("."):
            continue
        img = cv2.imread(os.path.join(path, item))
        img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
```

```
img = cv2.resize(img, (227, 227))
dataset.append([img, directory])
'''
dataset = [
[[...], 'hello'],
[[...], 'yes'],
        ...
]
'''
data, labels = zip(*dataset)
labels = list(map(mapper, labels))
'''
labels: hello,yes,yes,no,hello...
one hot encoded: [1,0,0], [0,1,0], [0,1,0], [0,0,1], [1,0,0]...
'''
# one hot encode the labels
labels = np_utils.to_categorical(labels)

# define the model
model = get_model()
model.compile( optimizer=Adam(lr=0.0001),
loss='categorical_crossentropy',
metrics=['accuracy'] )
# start training
model.fit(np.array(data), np.array(labels), epochs=10)
```

```
        # save the model for later use
        model.save("blind-deaf-communication-pro-max-model.h5")
```

3.4.3 PREDICTION MODEL CODE:

```
from keras.models import load_model
import cv2
import numpy as np
import os, time
from boltiot
import Bolt api_key = "d415a360-01d3-4a46-a3d1-0fa2f5498dd6"
device_id = "BOLT6555568"
mybolt = Bolt(api_key,device_id)
REV_CLASS_MAP = {
        0: "hello",
        1: "yes",
        2: "no",
        3: "none" }
def mapper(val):
        return REV_CLASS_MAP[val]
        model = load_model("blind-deaf-communication-test-model.h5")
        #capture the sign and the path of sign image is automatically given
        IMG_SAVE_PATH = 'E:\OpenCV\Sign'
try:
        os.remove("E:/OpenCV/Sign/sample_sign.jpg")
        except: pass
try:
```

```
        os.mkdir(IMG_SAVE_PATH)
        except FileExistsError:
pass
        cap = cv2.VideoCapture(0)
        cv2.namedWindow("Capturing...")
        start = False
        count = 0
while True:
        ret, frame = cap.read()
if not ret:
        continue
if count == 1:
        break
cv2.rectangle(frame, (50, 50), (270, 270), (255, 255, 255), 2)
if start:
        roi = frame[50:270, 50:270]
        save_path = os.path.join(IMG_SAVE_PATH, 'sample_sign.jpg')
        cv2.imwrite(save_path, roi)
        count += 1 font = cv2.FONT_HERSHEY_SIMPLEX
        cv2.putText(frame, "Capturing Sign", (5, 50), font, 0.7, (0, 255, 255), 2,
        cv2.LINE_AA) cv2.imshow("Collecting Sign image", frame)
        k = cv2.waitKey(10)
        if k == ord('s'):
                start = not start
        if k == ord('q'):
                break
```

```
print("\n{} image saved to {}".format(count, IMG_SAVE_PATH))
cap.release()
cv2.destroyAllWindows()
#filepath = 'E:\OpenCV\Sign\sample.jpg'
# prepare the image img = cv2.imread(save_path)
img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
img = cv2.resize(img, (227, 227))
# predict the move made
pred = model.predict(np.array([img]))
move_code = np.argmax(pred[0])
move_name = mapper(move_code)
print("Predicted: {}".format(move_name))
#Logic for implementation of task
if (move_code == 0):
        response = mybolt.analogWrite('0','200')
        response = mybolt.analogWrite('1', '200')
elif (move_code == 1):
        response = mybolt.analogWrite('0','200')
        response = mybolt.analogWrite('1', '0')
elif (move_code == 2):
        response = mybolt.analogWrite('0','0')
        response = mybolt.analogWrite('1', '200')
else:
        print("Invalid Input! Please make sure you are indicating the valid
Sign.")
print(response)
```

```
time.sleep(1)
response = mybolt.analogWrite('0','0')
response = mybolt.analogWrite('1','0')
print(response)
```

CHAPTER 4

RESULTS AND DISCUSSION

4.1 EVALUATION AND RESULT

When the application starts or code is being executed, a capturing window appears on the screen. A rectangular white box is constructed inside the window which starts from the point (50, 50) and ends at the point (270, 270). The program is designed in such a way that, it captures only the gestures made in the white box. 1000 reference images are used for each of the category to train the model, which are captured from five different backgrounds, exposure of light and various patterns of hand gesture as shown in the Figure 2.10.

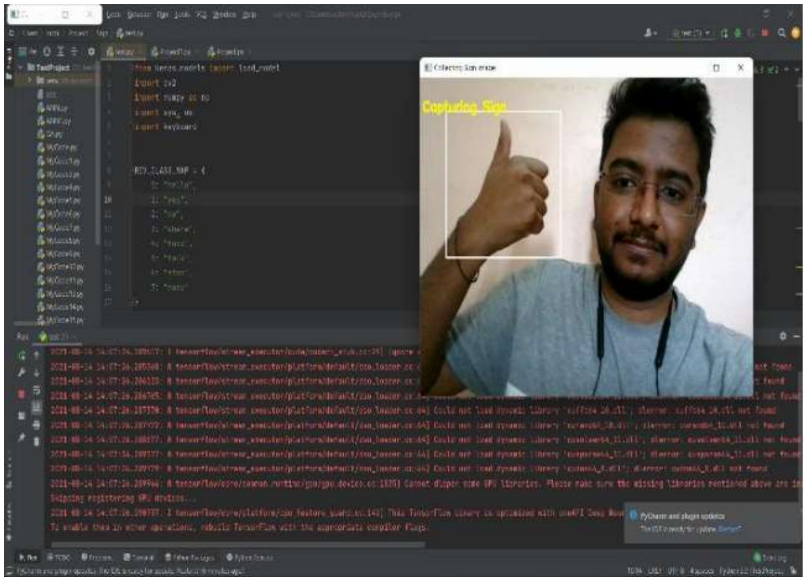


Figure 2.10: Capturing of the sign gesture.

Consider the phrase ‘YES’ given as an input to which the assigned pattern is 101. Therefore, vibrating module connected to Pin 0 and Pin 2 vibrates whereas Pin 1 remains in the off condition as shown in fig11. In this manner different signs are assigned with different combinations of the vibration.

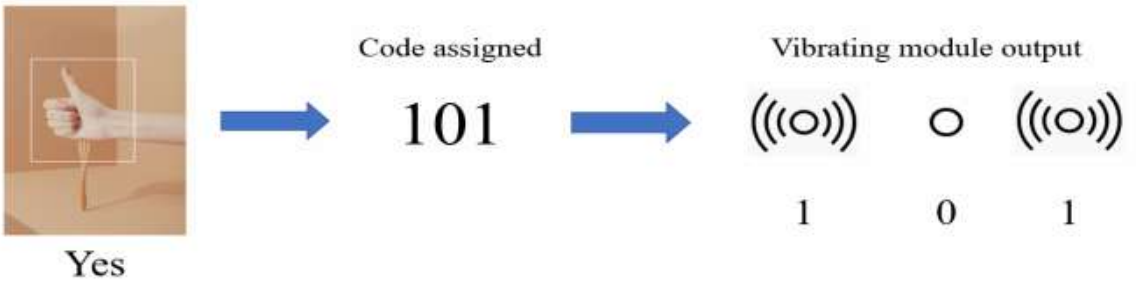


Figure 2.11: Process of assigning code to the word “YES”

The intensity of the vibrating modules can be changed according to the user’s preference. As shown in the table below for the SIGN GESTURES shown the CODE is assigned to each GESTURE NAME, according to which the vibrating motors vibrate. Here the blue dot indicates that the vibrating motor will vibrate and whereas the red dot indicates the motor does not vibrate. For each GESTURE the green and red dots are assigned with the specified pin numbers i.e., P0, P1, & P2 based on which we can understand the specific pattern for each SIGN GESTURE as shown in Figure 2.12.

The accuracy rate of our model is 75-78% with 10-12% of network issues as it requires active internet connection and 3- 5% of unsuccessful attempts due to improper gesture and other part is responsible for the wrong prediction which occurs due to different background or exposure of light. These statistics are represented graphically in the Figure 2.13. The accuracy can be improved by increasing the dataset of the reference images with variations. There is a need of strong internet connection in order to run the application.

INPUT		OUTPUT		
SIGN GESTURE	SIGN GESTURE NAME	PIN0	PIN1	PIN2
	HELLO			
	YES			
	NO			
	WHERE			
	FOOD			
	TALK			
	STOP			
	NONE			

Figure 2.12: Patterns used for sign gestures in this project

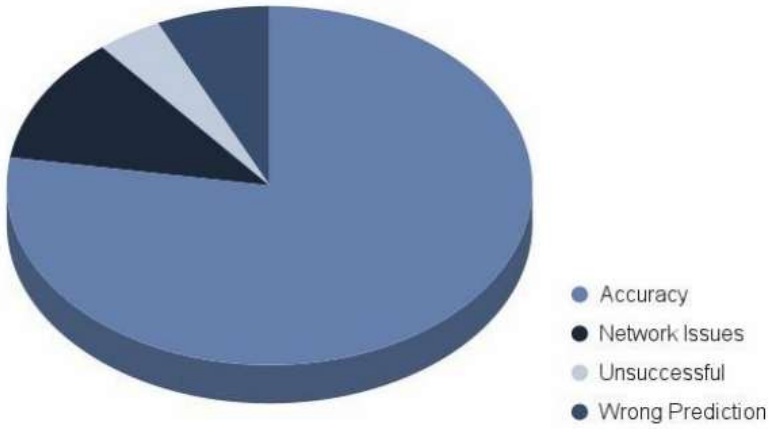


Figure 2.13: Graphical representation of the accuracy rate

Buzzer is used to indicate if any unseemly gesture is made to save time and increase accuracy. The total time taken to predict a gesture is about 2-3 seconds but to restart and run the model it requires nearly 20 seconds. The program captures one image after the other in a loop until it is terminated. The time consumed between two successive images is 5-8 seconds.

4.2 SUMMARY

This model mainly concentrates on enhancing an easy lifestyle for the impaired people. These people are gifted with sensitive touch which helps in recognizing and understanding their surrounding environment. Our model is designed in such a way that, we can communicate with the target audience even from a distance through a stable internet connection and the application converts the sign language to touch, which widens the range of users. In addition to the Gesture Recognition using the Deep Learning the model also assigns a specific move code to each category of hand gesture i.e., Hello is encoded as 0 0 1. The move code basically is a 3-bit binary code assigned to the move name i.e., Hello. Devices are developed using the high-end technologies available in order to provide assistance to impaired individuals but most of them are expensive, not portable and few of them are commercially not available. This paper solely aims to overcome these issues by implementing user friendly armband which consists of vibrating modules that conveys the input information to the user. Further improvement can be made by increasing the number of datasets and by training the model more in order to achieve 100% accuracy.

CHAPTER 5

CONCLUSION AND EXTENSION

5.1 CONCLUSION:

Image recognition for the blind is a mini project that aims to help visually impaired individuals to recognize images using technology. The project involves the use of machine learning algorithms and computer vision techniques to identify and describe images to the blind.

The project is designed to be user-friendly and accessible to individuals with visual impairments. The system uses a camera or a smartphone to capture an image, which is then processed by the machine learning algorithm. The algorithm analyzes the image and provides a description of the objects and their location within the image.

The system can be used in various settings, such as in public places, museums, and educational institutions. It can also be used in personal settings, such as at home or while traveling. The system can help visually impaired individuals to navigate their surroundings, identify objects, and learn about their environment.

The project has several benefits, including increased independence and improved quality of life for visually impaired individuals. It can also help to reduce the stigma associated with visual impairments and promote social inclusion.

In conclusion, image recognition for the blind is a promising mini project that has the potential to improve the lives of visually impaired individuals. The project combines the power of machine learning and computer vision to provide a user-friendly and accessible system that can help visually impaired individuals to recognize images and navigate their surroundings. With further development and refinement, this project could have a significant impact on the lives of millions of visually impaired individuals around the world.

5.2 EXTENSION:

An extension to the image recognition for the blind mini project could be the integration of a text-to-speech feature. This feature would allow the system to not only identify and describe images to the blind but also read out the description aloud. This would make the system even more accessible and user-friendly for visually impaired individuals.

The text-to-speech feature could be integrated into the existing system using speech synthesis technology. The system could use natural language processing algorithms to convert the image description into speech. The speech output could be customized to suit the user's preferences, such as the language, speed, and volume.

The integration of a text-to-speech feature would enhance the usability and effectiveness of the image recognition system. It would enable visually impaired individuals to receive real-time feedback on the images they encounter, allowing them to navigate their surroundings more confidently and independently. The feature would also make the system more inclusive, as it would cater to individuals who may have difficulty reading or comprehending text.

In conclusion, the integration of a text-to-speech feature would be a valuable extension to the image recognition for the blind mini project. It would enhance the accessibility and usability of the system, enabling visually impaired individuals to recognize images and navigate their surroundings more effectively. With further development and refinement, this extension could have a significant impact on the lives of visually impaired individuals around the world.

APPENDIX

Code to capture images using OpenCV:

```
desc = “”
Script to gather data images with a particular label.
Usage: python gather_images.py.
<label_name><num_samples><background_name>
The script will collect <num_samples> number of images and store them in its own directory.
 Only the portion of the mage within the box displayed will be captured and stored.
Press 's' to start/pause the image collecting process.
Press 'q' to quit.
'''
import cv2
import os
import sys
def deleteprefiles(file_name):
try:
        img_counter = 0
        for img_counter in range(1, 201):
                os.remove("E:/OpenCV/reference/"+label_name+"/{}.jpg".format(img_counter))
                img_counter += 1
        except: pass
try:
        label_name = sys.argv[1]
        num_samples = int(sys.argv[2])
```

```
        background = sys.argv[3]
except:
        print("Arguments missing.")
        print(desc)
        exit(-1)
IMG_SAVE_PATH = 'E:\OpenCV\Reference'
IMG_CLASS_PATH = os.path.join(IMG_SAVE_PATH, label_name)
try:
        os.mkdir(IMG_SAVE_PATH)
except FileExistsError:
        pass
try:
        os.mkdir(IMG_CLASS_PATH)
except FileExistsError:
        print("{} directory already exists.".format(IMG_CLASS_PATH))
        print("Deleting existing items in this folder")
        #deleteprefiles(label_name)
cap = cv2.VideoCapture(0)
cv2.namedWindow("Capturing...")
start = False
count = 0
while True:
        ret, frame = cap.read()
        if not ret:
                continue
```

```
        if count == num_samples:
        break
cv2.rectangle(frame, (50, 50), (270, 270), (255, 0, 0), 2)
if start:
        roi = frame[50:270, 50:270]
        save_path = os.path.join(IMG_CLASS_PATH, background +
'{}.jpg'.format(count + 1))
        cv2.imwrite(save_path, roi)
        count += 1
        font = cv2.FONT_HERSHEY_SIMPLEX
        cv2.putText(frame, "Collecting {}".format(count),(5, 50), font, 0.7, (0, 255,
255), 2, cv2.LINE_AA)
        cv2.imshow("Collecting images", frame)
        k = cv2.waitKey(10)
        if k == ord('s'):
                start = not start
        if k == ord('q'):
                break
        print("\n{} image(s) saved to {}".format(count, IMG_CLASS_PATH))
        cap.release()
        cv2.destroyAllWindows()
```

CODE TO TRAIN MODEL:

```
        import cv2
        import numpy as np
        from keras_squeezenet import SqueezeNet
```

```
    from keras.optimizers import Adam
    from keras.utils import np_utils
    from keras.layers.core import Activation
    from keras.layers.core import Dropout
    from keras.layers import Convolution2D
    from keras.layers import GlobalAveragePooling2D
    from keras.models import Sequential
    import tensorflow as tf
    import os
    IMG_SAVE_PATH = 'E:\OpenCV\Reference'
    CLASS_MAP = {
        "hello": 0,
        "yes": 1,
        "no": 2,
        "where": 3,
        "food": 4,
        "talk": 5,
        "stop": 6,
        "none": 7
    }
    NUM_CLASSES = len(CLASS_MAP)
    def mapper(val):
        return CLASS_MAP[val]
def get_model():
    model = Sequential([
```

```
        SqueezeNet(input_shape=(227, 227, 3), include_top=False),
        Dropout(0.5),
        Convolution2D(NUM_CLASSES, (1, 1), padding='valid'),
        Activation('relu'),
        GlobalAveragePooling2D(),
        Activation('softmax')
])
return model
# load images from the directory
dataset = []
for directory in os.listdir(IMG_SAVE_PATH):
        path = os.path.join(IMG_SAVE_PATH, directory)
        if not os.path.isdir(path):
        continue
for item in os.listdir(path):
        # to make sure no hidden files get in our way
        if item.startswith("."):
                continue
        img = cv2.imread(os.path.join(path, item))
        img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
        img = cv2.resize(img, (227, 227))
        dataset.append([img, directory])
        '''
        dataset = [
        [[...], 'hello'],
```

```
[[...], 'yes'],
        ...
]
'''
data, labels = zip(*dataset)
labels = list(map(mapper, labels))
'''
labels: hello,yes,yes,no,hello...
one hot encoded: [1,0,0], [0,1,0], [0,1,0], [0,0,1], [1,0,0]...
'''
# one hot encode the labels
labels = np_utils.to_categorical(labels)

# define the model
model = get_model()
model.compile( optimizer=Adam(lr=0.0001),
loss='categorical_crossentropy',
metrics=['accuracy'] )
# start training
model.fit(np.array(data), np.array(labels), epochs=10)
# save the model for later use
model.save("blind-deaf-communication-pro-max-model.h5")
```

PREDICTION MODEL CODE:

```
from keras.models import load_model
import cv2
```

```
import numpy as np
import os, time
from boltiot
import Bolt api_key = "d415a360-01d3-4a46-a3d1-0fa2f5498dd6"
device_id = "BOLT6555568"
mybolt = Bolt(api_key,device_id)
REV_CLASS_MAP = {
        0: "hello",
        1: "yes",
        2: "no",
        3: "none" }
def mapper(val):
        return REV_CLASS_MAP[val]
        model = load_model("blind-deaf-communication-test-model.h5")
        #capture the sign and the path of sign image is automatically given
        IMG_SAVE_PATH = 'E:\OpenCV\Sign'
try:
        os.remove("E:/OpenCV/Sign/sample_sign.jpg")
        except: pass
try:
        os.mkdir(IMG_SAVE_PATH)
        except FileExistsError:
pass
        cap = cv2.VideoCapture(0)
        cv2.namedWindow("Capturing...")
```

```
    start = False
    count = 0
while True:
    ret, frame = cap.read()
if not ret:
    continue
if count == 1:
    break
cv2.rectangle(frame, (50, 50), (270, 270), (255, 255, 255), 2)
if start:
    roi = frame[50:270, 50:270]
    save_path = os.path.join(IMG_SAVE_PATH, 'sample_sign.jpg')
    cv2.imwrite(save_path, roi)
    count += 1 font = cv2.FONT_HERSHEY_SIMPLEX
    cv2.putText(frame, "Capturing Sign", (5, 50), font, 0.7, (0, 255, 255), 2,
    cv2.LINE_AA) cv2.imshow("Collecting Sign image", frame)
    k = cv2.waitKey(10)
    if k == ord('s'):
        start = not start
    if k == ord('q'):
        break
    print("\n{} image saved to {}".format(count, IMG_SAVE_PATH))
    cap.release()
    cv2.destroyAllWindows()
    #filepath = 'E:\OpenCV\Sign\sample.jpg'
    # prepare the image img = cv2.imread(save_path)
```

```
img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
img = cv2.resize(img, (227, 227))
# predict the move made
pred = model.predict(np.array([img]))
move_code = np.argmax(pred[0])
move_name = mapper(move_code)
print("Predicted: {}".format(move_name))
#Logic for implementation of task
if (move_code == 0):
        response = mybolt.analogWrite('0','200')
        response = mybolt.analogWrite('1', '200')
elif (move_code == 1):
        response = mybolt.analogWrite('0','200')
        response = mybolt.analogWrite('1', '0')
elif (move_code == 2):
        response = mybolt.analogWrite('0','0')
        response = mybolt.analogWrite('1', '200')
else:
        print("Invalid Input! Please make sure you are indicating the valid
Sign.")
print(response)
time.sleep(1)
response = mybolt.analogWrite('0','0')
response = mybolt.analogWrite('1','0')
print(response)
```

REFERENCES

[1] O. Ozioko, P. Karipoth, M. Hersh and R. Dahiya, "Wearable Assistive Tactile Communication Interface Based on Integrated Touch Sensors and Actuators," in IEEE Transactions on Neural Systems and Rehabilitation Engineering, vol. 28, no. 6, pp. 1344- 1352, June 2020, doi: 10.1109/TNSRE.2020.2986222.

[2] Y. Matsuda and T. Isomura, "Emotion Feature Vector of Finger Braille," 2009 Fifth International Conference on Intelligent Information Hiding and Multimedia Signal Processing, 2009, pp. 877-880, doi: 10.1109/IIHMSP.2009.125.

[3] C. Giulia, D. V. Chiara and H. Esmailbeigi, "GLOS: GLOve for Speech Recognition," 2019 41st Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC), 2019, pp. 3319- 3322, doi: 10.1109/EMBC.2019.8857927.

[4] O. Ozioko, W. Taube, M. Hersh and R. Dahiya, "SmartFingerBraille: A tactile sensing and actuationbased communication glove for deafblind people," 2017 IEEE 26th International Symposium on Industrial Electronics (ISIE), 2017, pp. 2014-2018, doi: 10.1109/ISIE.2017.8001563.

[5] H. Culbertson, C. M. Nunez, A. Israr, F. Lau, F. Abnousi and A. M. Okamura, "A social haptic device to create continuous lateral motion using sequential normal indentation," 2018 IEEE Haptics Symposium (HAPTICS), 2018, pp. 32-39, doi: 10.1109/HAPTICS.2018.8357149.

[6] S. Casini, M. Morvidoni, M. Bianchi, M. Catalano, G. Grioli and A. Bicchi, "Design and realization of the CUFF - clenching upper-limb force feedback wearable device for distributed mechano-tactile stimulation of normal and tangential skin forces," 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2015, pp. 1186-1193, doi: 10.1109/IROS.2015.7353520.

[7] S. (2021, March 24). What is Tensorflow: DeepLearningLibrariesandProgramElementsExplained. Simplilearn.Com. https://www.simplilearn.com/tutorials/deep-learningtutorial/what-is-tensorflow.

[8] GeeksforGeeks. (2019, December 27). Python: ConvertSpeechtotextandtexttoSpeech. https://www.geeksforgeeks.org/python-convert-speech-totext-and-text-to-speech/

[9] Subramanian, D. (2020, July 9). EasySpeech-toTextwithPython - TowardsDataScience. Medium. Easy Speech-to-Text with Python. Speech to Text | by Dhilip Subramanian | Towards Data Science

[10] B, A. (2020, December 15). GestureRecognitionforBeginnerswithCNN - TowardsDataScience. Medium. Gesture Recognition for Beginners with CNN | by That Data Bloke | Towards Data Science

[11] Tsang, S. (2019, April 22). Review: SqueezeNet (Image Classification):Review: SqueezeNet (Image Classification) | by Sik-Ho Tsang | Towards Data Science

[12] Rosebrock, A. (2021, April 17). ImageclassificationwithKerasanddeeplearning. PyImageSearch. https://www.pyimagesearch.com/2017/12/11/imageclassification-withkeras-and-deep-learning/

[13] APIIntroduction. (2019). Bolt IoT: A Fully Integrated IoT Platform. https://docs.boltiot.com/docs/introduction.

[14] GettingstartedwithBoltIoT. (n.d.). Bolt IoT: A Fully Integrated IoT Platform. Retrieved July 2, 2021, from https://docs.boltiot.com/docs/getting-started-with-boltiot

[15] NHS website. (2019, July 23). Deafblindness. Nhs.Uk. https://www.nhs.uk/conditions/deafblindness/

[16] Deaf-Blind Communication Technology. (n.d.). Nfb. Retrieved July 5, 2021, from https://nfb.org//sites/default/files/images/nfb/publications/m/bm14/bm1409/bm140906.html

[17] Li, A., & Li, A. (2017, June 1). Google's speech recognition is now almost as accurate as humans. 9to5Google. https://9to5google.com/2017/06/01/googlespeechrecognition-humans