"SIGN LANGUAGE CONVERTOR"

A Major Project Report Submitted to

Rajiv Gandhi Proudyogiki Vishwavidhyalaya, Bhopal

Towards Partial Fulfillment for the Award of

Bachelor of Technology

(Computer Science and Engineering)

Submitted By:

Akash Prajapat (0827CS171013)

Anurag Jain (0827CS171034)

Chanchal Agrawal (0827cs171058)

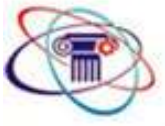
CS IV YEAR

Supervised By:

Prof. Shivshankar Rajput

Guided by:

Ms. Nisha Rathi



Department of Computer Science and Engineering

Acropolis Institute of Technology and Research, Indore

July-December 2020

CHAPTER 1
INTRODUCTION

A machine learning program that detects and recognizes various signs performed by the hand that are been captured and fed the program through a video stream (webcam or a particular IP webcam). This program works on a supervised learning algorithm which is used to create an initial logic tree which helps the algorithm to later on recognize those actions. The machine can recognize and identify only those signs that are fed into it by the user while training the module on a particular dataset which has a finite number of videos (showing signs) and all and all the objects that are stored in the logic tree from which the program derives its answers to recognize the signs and provide you with an audio output.
Sign gestures can be majorly classified into two types static and dynamic. Static gesture is simpler than dynamic gesture recognition, in static only single image is recognized at a time while on the other hand dynamic gesture is a moving gesture represented by various images. Sign language is used by people who have speaking or listening disability so that they can exchange information between other people and within their own community. This can be used on wide scale to make most of the public understand what a person is trying to convey to the world through sign language. This will prevent the disabled people for hiring other people who can speak and understand sign language just to be their communicator.

RATIONALE

Human beings interact with each other either using different language channel such as words, writing, or by gestures e.g. hand and head gestures, facial expression, lip movements and so on. Sign language is a natural language and as understanding natural language is important, understanding sign language is also worthwhile. Sign language is a communication method for people with hearing and speaking disability. People with hearing and speaking disabilities face problems in communicating with other hearing people without a translator. Even after decades of digitalization some abstract topics such as this one still remain unexplored. In this project, we have proposed an algorithm which is capable of extracting sign's from a live video stream. We are working towards enhancement of deaf people's social life by providing them this sign language orator which would help them to interact with people who are not familiar with sign language. This system can be used for professional conversations as wells

as for day to day life chores. This algorithm will help us in converting video of frequently used full sentences gesture into a transcript and then finally convert it into speech. So the signs and their corresponding meanings will be interpreted and identified. Other people who are not familiar with this non-verbal communication can understand easily which will ultimately prevent them to hire a person to communicate for them and save their money.

PROBLEM DEFINITION

According to the World Health Organization (WHO), around 5% of world population belongs to the people with the hearing and speaking disability that totals over 360 million people across the globe. The majority of these people live in countries with comparatively low incomes. Sign Language is an independent language which is different from spoken/written language, the basic difference is it has limited vocabulary compared to written/spoken. Sign language is not the same in every country, different sign languages are used in different countries or regions. As you may know there are two separate languages ASL and BSL among which ASL is the most widely used signed language.
This paper describes a technology in which real time videos are analysed and are used for hand movement detection and recognition, thus helping them to convey what they want to explain or tell, in the form of transcripts and converts it into audio. The system developed identifies sign language (non-verbal communication) done by using some hand gestures and the machine is trained to recognize some daily frequently used gestures and convert them into verbal communication so that the other people who are not familiar with this non-verbal communication can understand easily which will ultimately prevent them to hire a person to communicate for them and save their money.
We focus on mainly these two points:-
Persons who are disabled to speak or hear converse in sign language and have a problem in communicating what they actually mean.
A mediator is needed just for the communication between two people if one is disabled and the other is not.

PROCESS FLOW OF EXISTING SYSTEM

Figure 1.3: Process Flow of Existing System

LIMITATION OF EXISTING SYSTEM

Some limitations of the existing system are:
Can only work on images or videos.
Uses image processing algorithms.
Less efficient.
Time consuming.
Uses more memory as images and videos are loaded first.

PROPOSED SOLUTION

The current system that we are proposing would work on processing of live video stream rather than processing of a still image. It actually identifies the signs on real time basis which are in front of machine or host that has a web cam and tells us about the communications done in transcripts form to read. The current system is developed in python and works on machine learning algorithm that uses supervised learning as its basis. It uses a specified Dataset which has information (Q&A) about a finite number of sign videos provided, using this information only the algorithm recognizes the nonverbal communication done in front of the machine. It

also has the capability to recognize all types of characters from A to Z or numbers 0 to 9.

OBJECTIVE & SCOPE

The following objectives are achieved in the project:-
Real time translation.
Eliminates the need for an interpreter between sign language and common speech.
Does not require additional hardware.
Easy to incorporate in any OS
The project's scope is summarized as follows:-
This can be used to help disabled people.
This can be used by people to understand the sign language and help those people communicate through it.
This can be used on wide scale to make most of the public understand what a person is trying to convey to the world through sign language.
This will prevent the disabled people from hiring other people who can speak and understand sign language just to be their communicator.
This can be implemented with other technologies too like iot, android and thus has a lot of scope.

REPORT ORGANIZATION

Chapter 1 The introduction to the thesis is given in this section and it describes the objectives, motivation and justification.
Chapter 2 reviews numerous existing and emerging technologies that are related to the work presented in this thesis under the title literature survey
Chapter 3 Analysis of the whole work is done by explaining the detailed work performed in the project.
Chapter 4 System Design of the whole work which includes the UML diagrams such as Use Case, Activity diagram and Sequence diagram.
Chapter 5 describes the various results that are obtained after the complete implementation and screenshots of the project.
Chapter 6 includes the testing part in the project.
Chapter 7 includes the functional and non-functional requirements.
Chapter 8 draws conclusions from the work described in previous chapters and discusses possibilities for future development.

CHAPTER 2
LITERATURE SURVEY

According to the World Health Organization (WHO), around 5% of world population belongs to the people with the hearing and speaking disability that totals over 360 million people across the globe. The majority of these people live in countries with comparatively low incomes. Sign Language is an independent language which is different from spoken/written language, the basic difference is it has limited vocabulary compared to written/spoken. Sign language is not the same in every country, different sign languages are used in different countries or regions. As you may know there are two separate languages ASL and BSL among which ASL is the most widely used signed language.

The real time input video is initially segmented into frames and the systems that are developed works on images to recognize sign language, the main device used is camera, which helps to capture a gesture image which is an input data for Sign Language Recognition (SLR) and this image is then processed and the result is obtained. There is also other device like Microsoft Kinect which is used to capture image. It is widely used by researchers because of it features like color video stream and depth video stream.

A general framework for hand movement detection and analysis involves stages such as motion detection with the help of background modelling and foreground segmentation, object classification, motion tracking and activity recognition.

This project describes a technology in which real time videos are analyzed and are used for hand movement detection and recognition, thus helping them to convey what they want to explain or convey, in the form of transcripts and converts it into audio. The system developed identifies sign language (non-verbal communication) done by using some hand gestures and the machine is trained to recognize some daily frequently used gestures and convert them into verbal communication so that the other people who are not familiar with this non-verbal communication can understand easily which will ultimately prevent them to hire a person to communicate for them and save their money.

A Phoneme based method is also used to recognize the sign language. There are 44 phonemes in English language; therefore 44 gestures can be formed. There are 11 categories of gestures which are formed in ASL. The right hand shows categories, left hand shows sign in category. For prepossessing and filtering the image, RGB colour space is used. RGB algorithm shows the head and hands region. The next step consists of using the vertical interleaving method for image compression. The features of the image are extracted using the 2-D moment invariant which are then fed to the neural network which recognizes the equivalent text.

2.1 BACKGROUND

In recent years, many advanced classification approaches, such as artificial neural networks, fuzzy- sets, and expert systems, have been widely applied for image classification. Cihlar (2000 Cihlar, J. 2000. Land cover mapping of large areas from satellites: status and research priorities.. International Journal of Remote Sensing, 21: 1093–1114.

Discussed the status and research priorities of land- cover mapping for large areas. Franklin and Wulder (2002 Franklin, S. E. and Wulder, M. A. 2002. Remote sensing methods in medium spatial resolution satellite data land cover classification of large areas.. Progress in Physical

Geography, 26: 173–205.

One of the milestones in deep learning, this research paper “ImageNet Classification with Deep Convolutional Neural Networks” started it all. Even though deep learning had been around since the 70s with AI heavyweights Geoff Hinton, Yann LeCun and Yoshua Bengio working on Convolutional Neural Networks, AlexNet brought deep learning into the mainstream. Authored by Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton, this 2012 paper won the ImageNet Large Scale Visual Recognition Challenge with a 15.4% error rate. In fact, 2012 marked the first year when a CNN was used to achieve a top 5 test error rate of 15.4% and the next best research paper achieved an error rate of 26.2. the paper was groundbreaking in its approach and brought the many concepts of deep learning into the mainstream.

Inspired by the Inception thriller, GoogleNet proposes a deep convolutional neural network architecture codenamed “Inception”, which was responsible for setting the new state of the art for classification and detection in the ImageNet Large-Scale Visual Recognition Challenge. This Google project proposed a 22 layer convolutional neural network and was the winner of ILSVRC 2014 with a rate of 6.7%. According to experts, this CNN architecture was the first to propose a different approach from the general approach of simply stacking and pooling layers on top of each other.

Developed in response to index images, GoogleNet research project was undertaken by Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Drago Anguelov, Dumitru Erhan, Andrew Rabinovich and Christian Szegedy. At the core of the project was a reworked convolutional network architecture consisting of 100+ layers with a depth of 20 parameter layers) and is based Hebbian principle and scale invariance. Over the years, Google has been experimenting with neural networks to improve its image search ability and understand the content within Youtube videos.Google is leveraging these research advances and converting it into Google products such as in YouTube, image search and even self-driving cars.

The research paper authored by Matthew D Zeiler and Rob Fergus introduced a novel visualization technique that gave a peek into the functioning of intermediate feature layers and the operation of the classifier. This architecture was trained on 1.3 million images and it developed a visualization technique called De Deconvolutional Network that helped to examine different feature activations and their relation to the input space. The paper proposed to outperform Krizhevsky on the ImageNet classification benchmark.

Authored by two University of Maryland researchers Rama Chellappa, Swami Sankaranarayanan and GE Global researchers Arpit Jain and Ser Nam Lim proposed a simple learning algorithm that leveraged perturbations of intermediate layer activation to provide a stronger regularization while improving the robustness of deep network to adversarial data. The research dealt with the behaviour of CNNs as related to adversarial data and the intrigue it had generated in computer vision. However, the effects of adversarial data on deeper networks had not been explored well. The paper cited results of adversarial perturbations for hidden layer activations across different samples and leveraged this observation to devise an efficient adversarial training approach that could be used to train deep architectures.

The latest research paper proposed a “Residual Attention Network” – a convolutional neural network that leverages attention mechanism which can incorporate feed forward network architecture in an end-to-end training fashion. Authored by Fei Wang, Mengqing Jiang, Chen Qian, Shuo Yang, Cheng Li, Honggang Zhang, Xiaogang Wang, Xiaoou Tang, their research methodology achieved a 0.6% top-1 accuracy improvement with 46% trunk depth and 69% forward FLOPs comparing to ResNet-200. The experiment also demonstrates that the neural network is robust vis-à-vis noisy labels.

Assessed land- cover classification approaches with medium spatial resolution remotely

sensed data. Books by Tso and Mather (2001 Tso, B. and Mather, P. M. 2001. Classification Methods for Remotely Sensed Data, New York: Taylor and Francis Inc.
And Landgrebe (2003 Landgrebe, D. A. 2003. Signal Theory Methods in Multispectral Remote Sensing, Hoboken, NJ: John Wiley and Sons.
Specifically focus on image- processing approaches and classification algorithms. In general, image classification approaches can be grouped as supervised and unsupervised, or parametric and non- parametric, or hard and soft (fuzzy) classification, or per- pixel, subpixel, and per- field. Table 1 provides brief descriptions of these categories. For the sake of convenience, this paper group's classification approaches as per- pixel, subpixel, per- field, contextual- based, knowledge- based, and a combination of multiple classifiers.The table below lists major advanced classification approaches that have appeared in recent literature. A brief description of each category is provided in the following subsection.
Figure 2.1: Research Paper Summary

2.2 RELATED WORK

Machine Learning

Artificial intelligence (AI) is a subarea of computer science that emphasizes the creation of automated machines that work and reacts like humans. Machine learning system is a branch of artificial intelligence based on an idea that a system can produce general hypothesis by learning from data provided, identify patterns and make decisions with minimal human intervention. Machine learning is important because as models are exposed to new data, they are able to create a predictive model capable of inferring annotations for future data. They learn from

previous computations to produce reliable, repeatable decisions and results without needing multiple manual edits to the program.

Supervised Learning

Supervised Learning is a computational task of automated data (the training data set) to produce general hypothesis. The training data set comprises of input and output pairs which are used to train the model and determine a hidden pattern. This hidden pattern is then used for recognition of specific patterns when working data is provided in future.
Figure 2.2(a): Supervised Learning
Supervised learning is in which you have input variable (a) and an output variable (b), and you apply an algorithm to learn the mapping function from the input to the output.
$Y = f(A)$
The aim is to approximate the mapping function so well that when you have new input data (a), you can predict the output variables (B) for that data. Supervised learning is called so because the method of algorithm learning from the training dataset can be thought of as a trainer supervising the learning process. We know the accurate results; the algorithm iteratively makes predictions on the training data and is corrected by the trainer. Learning stops when the algorithm achieves an agreeable level of performance.

Deep Neural Networks

Neural networks is a set of algorithm which is used to recognize the relationship between the underlying set of data in way that a human brain works which constantly try to recognize patterns and categorize and classify information. It is an excellent tool for finding patterns which are complex as well as numerous for programmers to teach the machine to recognize. It usually involves a large number of processors that are operating in parallel and are arranged in

tiers. Hidden layer is in between the two input and output layers.

Deep neural network is similar to deep learning, with certain level of complexity and has more than two layers. The different layers of such a system could be seen as a nested hierarchy of related concepts or decision trees. Deep neural network systems need large quantities of data in order to be trained as the systems learns from exposure to huge number of data points. Google Brain learning to recognize cats after being shown over ten million images can be seen as an early example of this. Figure 2.2(c): Multi- layered Deep Neural Network

TensorFlow

TensorFlow is a free and open-source software library for dataflow and differentiable programming across a range of tasks. It is a symbolic math library, and is also used for machine learning applications such as neural networks. It is used for both research and production at Google. It is a standard expectation in the industry to have experience in TensorFlow to work in machine learning. TensorFlow was developed by the Google Brain team for internal Google use. It was released under the Apache 2.0 open-source license on November 9, 2015.

OpenCV

OpenCV (Open Source Computer Vision Library) is a library developed by Intel of programming functions and is released under BSD licence hence it's free for both academic and commercial use. It is a powerful library designed to work on real time applications with a strong real-time efficiency. It is written in C++ and it's primay interface is based on C++ this makes OpenCV portable to almost any commercial system includesPython, JAVA, MATLAB/OCTAVE interfaces which are supported by Windows, Mac OS, iOS, Linux, FreeBSD, OpenBSD and Android as it was designed to be a cross-platform, the library can take advantage on multi level processing thus makes it easy for businesses to utilize and modify the code.

CHAPTER 3
SYSTEM REQUIREMENT SPECIFICATION

3.1 HARDWARE REQUIREMENTS

Processor: Pentium, AMD or Higher Version
Operating System: Windows XP/ Windows 7/ Linux
RAM: 256 MB, 2GB recommended
Hardware Devices: Keyboard with mouse
Hard disk: 10 GB or more
Display: Standard Output Display
Camera: Webcam

3.2 SOFTWARE REQUIREMENTS

Technology Implemented: Sublime Text
Language Used: Python
User Interface Design: Terminal/CMD
Model: MobileNetSSD (Caffe Model, Prototxt)

CHAPTER 4
SYSTEM ANALYSIS & DESIGN

Based on the user requirements and the detailed analysis of a system, the system is designed. This is the phase of system designing. It is a crucial phase in the development of a system. It includes the UML diagrams.

4.1 SYSTEM ANALYSIS

System development has two major component- system analysis and system design. It refers to the process of examining a business situation with the intent of improving it methods and procedures.

4.1.1 REQUIREMENT ANALYSIS

Requirements analysis, also called requirements engineering, is the process of determining user expectations for a new or modified product. These features, called requirements, must be quantifiable, relevant and detailed. In software engineering, such requirements are often called functional specifications. This section describes the overall functional requirements of this system which is needed to operate this system successfully. In Software engineering and
A -function is described as a set of inputs, the behaviour, and outputs. Non-functional requirements include constraints and qualities. System qualities are properties or characteristics of the system that its stakeholders care about and hence will affect their degree systems engineering, a functional requirement defines a function of a system or its component. of satisfaction with the system. A constraint is a restriction on the degree of freedom we have in. Non-functional requirements define the needs in terms of performance, logical database requirements, design constraints, standard compliance, reliability, availability, security, maintainability and portability.

4.1.2 OBJECT ORIENTED ANALYSIS

Object Oriented Analysis is the first technical activity that is performed as a part of object oriented software engineering. Object Oriented Analysis introduces new concepts for

examining a problem. It is grounded in a set of basic principles, which are as follows –
The information domain is modelled.
Behaviour is represented.
Function is described.
Data, functional and behavioural models are divided to uncover greater detail.
Early models represent the essence of the problem, while later ones provide implementation details.

4.1.3 ARCHITECTURAL SPECIFICATION

A description of a software and electronics system in terms of its hardware and software components and their interactions. In (hardware, software, or enterprise) systems development, an architectural specification is the set of documentation that describes the structure, behaviour, and more views of that system. The architectural specification include the report organization of the project.
The role of the system architectural structure that serve as the blueprint for the target system.
Allocate required system functionality to hardware and software.
Document crucial design constraints, assumptions and rationales.
Support early analysis to make sure that the design approach is highly visible and open to peer review and progressive improvement.
Demonstrate compliance with system requirements. The SAAS provides an authoritative reference for detailed traceability analysis.
Explain how the end product will exhibit required qualities such as usability, performance, modifiability, safety and security.
Describe design management strategies to be used to control the development of the design including nomination of design patterns and rules.
Support project planning and budgeting.
Support preparation of acquisition documents (for example, requests for proposal and statements of work)
Support on-going maintenance and enhancement.

4.1.4 FEASIBILITY ANALYSIS

The use of technology in the field of event management would make it more time efficient and reliable. Taking economic feasibility into consideration our project is cost efficient as it would cost very minimal in its development also it would be easily available to the customer.

TECHNICAL FEASIBILITY

Since the security departments are ever researching for to techniques to tackle fraud this project is meant to suit their needs perfectly. This project is compatible to run on any system which satisfies the basic hardware and software requirements. It could be easily embedded into any banks existing system's backend to provide efficient and easy results.

BEHAVIORAL FEASIBILITY

By behavioural feasibility, we find that whether the users will accept the changes and whether our project is behaviourally feasible or not. Since the technology is booming and card frauds are increasing day by day, there is always a requirement of new technology which could be deployed to stop the frauds. Fraudsters are getting smarter day by day and adapting to the new security systems. Our system is desired to be used by banks or finance companies as it proposes a new model which would provide a latest approach to fraud which would try to stop the fraud at the initial stages, i.e., the application frauds.

OPERATIONAL FEASIBILITY

This software can run on Windows operating system (platform) or iOS having python installed. The only requirement for the proper functioning of this software is the minimum

software and hardware requirements that have been mentioned.

Time

Completion of any project in time is a very important. Our project is feasible in time. We have presented the project in allotted time.

Resource

Our project required least resource to run it successfully, no special or costly resources, no extra furniture; extra peripherals required just have a simple configuration system

4.1.5 DEVELOPMENT METHOD

After designing the new system, the whole system is required to be converted into computer understanding language. Coding the new system into computer programming language does this. It is an important stage where the defined procedures are transformed into control specifications by the help of a computer language. This is also called the programming phase in which the programmer converts the program specifications into computer instructions, which refer as programs. The programs coordinate the data movements and control the entire process in a system.

My development occurred in following phases

Platform selection: HTML, CSS, Java Script, Adobe Photoshop, My SQL.

Observed a tutorial site of w3schools.com.

Created drawings showing the future system.

Created new design of the system.

Received user reviews.

4.2 SYSTEM DESIGN

4.2.1 USE CASE DIAGRAM

A use case diagram in the Unified Modelling Language (UML) is a type of behavioural diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted. Use Case diagrams are formally included in two modelling languages defined by the OMG: the Unified Modelling Language (UML) and the Systems.

4.2.2 USE CASE SPECIFICATION

Application: Sign Language Orator

Use-case name: Gesture Recognition

Use-case description: This use case details the category for verifying the machine, dataset and video source

Primary actor: Developer/User

Precondition: The user/administrator successfully runs the application to load the database.

Post-condition: The user successfully portrays a category of gestures that the machine can detect. In this particular case American Sign Language (ASL) A-Z and 0-9 are preferred.

Basic Flow:

Run the application

Load the database

Work the neural network

Select the video input source

Display gestures in front of web cam

4.2.3 ACTIVITY DIAGRAM

Activity diagram is another important diagram in UML to describe dynamic aspects of the

system. Activity diagram is basically a flow chart to represent the flow form one activity to another activity. The activity can be described as an operation of the system. Activity diagrams provide a way to model the workflow of a business process. Activity diagrams can also be used to model code-specific information, such as a class operation. Activity diagrams are very similar to a flowchart because of modelling a workflow from activity to activity.

4.2.4 SEQUENCE DIAGRAM

A sequence diagram is a graphical view of a scenario that shows object interaction in a time-based sequence, what happens first, what happens next. Sequence diagrams establish the roles of objects and help provide essential information to determine class responsibilities and interface. This type of diagram is best used during early analysis phases because they are simple and easy to comprehend. A sequence diagram has two dimensions: typically, vertical placement represents time and horizontal placement represents different objects.

4.2.5 ATTACH DATA FLOW DIAGRAM

A Data Flow Diagram (DFD) is a structured analysis and design tool that can be used for flowcharting. A DFD is a network that describes the flow of data and the processes that change or transform the data throughout a system. This network is constructed by using a set of symbols that do not imply any physical implementation. It is the starting point of the design phase that functionally decomposes the requirements specifications down to the lowest level of detail. DFD can be considered to an abstraction of the logic of information-oriented or a process-oriented system flow-chart.

CHAPTER 5
PROJECT IMPLEMENTATION AND OUTPUT SCREEN

5.1 SCREENSHOTS OF YOUR PROJECT

Figure 5.1(a): ZERO
SIGN LANGUAGE ORATOR
Figure 5.1(b): K Alphabet
Figure 5.1(c): F Alphabet
Figure 5.1(d): U Alphabet
Figure 5.1(e) : Setting Histogram
SIGN LANGUAGE ORATOR
Figure 5.1(f): Display all Gestures
Figure 5.1(g): Detected ‘S’ alphabet

5.2 IMPORTANT CODING

```
import cv2, pickle
import numpy as np
import tensorflow as tf
from cnn_tf import cnn_model_fn
import os
import sqlite3
from keras.models import load_model
os.environ['TF_CPP_MIN_LOG_LEVEL'] = '3'
tf.logging.set_verbosity(tf.logging.ERROR)
classifier = tf.estimator.Estimator(model_dir="tmp/cnn_model2", model_fn=cnn_model_fn)
```

```
prediction = None
model = load_model('cnn_model_keras2.h5')
def get_image_size():
img = cv2.imread('gestures/0/100.jpg', 0)
return img.shape
image_x, image_y = get_image_size()
def tf_process_image(img):
img = cv2.resize(img, (image_x, image_y))
img = np.array(img, dtype=np.float32)
np_array = np.array(img)
return np_array
def tf_predict(classifier, image):
'''
need help with prediction using tensorflow
'''
global prediction
processed_array = tf_process_image(image)
pred_input_fn = tf.estimator.inputs.numpy_input_fn(x={"x":processed_array},
shuffle=False)
pred = classifier.predict(input_fn=pred_input_fn)
prediction = next(pred)
print(prediction)
def keras_process_image(img):
img = cv2.resize(img, (image_x, image_y))
img = np.array(img, dtype=np.float32)
img = np.reshape(img, (1, image_x, image_y, 1))
return img
def keras_predict(model, image):
processed = keras_process_image(image)
pred_probab = model.predict(processed)[0]
pred_class = list(pred_probab).index(max(pred_probab))
return max(pred_probab), pred_class
def get_pred_text_from_db(pred_class):
conn = sqlite3.connect("gesture_db.db")
cmd = "SELECT g_name FROM gesture WHERE g_id="+str(pred_class)
cursor = conn.execute(cmd)
for row in cursor:
return row[0]
def split_sentence(text, num_of_words):
'''
Splits a text into group of num_of_words
'''
list_words = text.split(" ")
length = len(list_words)
splitted_sentence = []
b_index = 0
e_index = num_of_words
while length > 0:
```

```
part = ""
for word in list_words[b_index:e_index]:
part = part + " " + word
splitted_sentence.append(part)
b_index += num_of_words
e_index += num_of_words
length -= num_of_words
return splitted_sentence
def put_splitted_text_in_blackboard(blackboard, splitted_text):
```

```
y = 200
for text in splitted_text:
cv2.putText(blackboard, text, (4, y), cv2.FONT_HERSHEY_TRIPLEX, 2,
(255, 255, 255))
y += 50
def get_hand_hist():
with open("hist", "rb") as f:
hist = pickle.load(f)
return hist
def recognize():
global prediction
cam = cv2.VideoCapture(1)
if cam.read()[0] == False:
cam = cv2.VideoCapture(0)
hist = get_hand_hist()
x, y, w, h = 300, 100, 300, 300
while True:
text = ""
img = cam.read()[1]
img = cv2.flip(img, 1)
img = cv2.resize(img, (640, 480))
imgCrop = img[y:y+h, x:x+w]
imgHSV = cv2.cvtColor(img, cv2.COLOR_BGR2HSV)
dst = cv2.calcBackProject([imgHSV], [0, 1], hist, [0, 180, 0, 256], 1)
disc = cv2.getStructuringElement(cv2.MORPH_ELLIPSE,(10,10))
cv2.filter2D(dst,-1,disc,dst)
blur = cv2.GaussianBlur(dst, (11,11), 0)
blur = cv2.medianBlur(blur, 15)
thresh =
cv2.threshold(blur,0,255,cv2.THRESH_BINARY+cv2.THRESH_OTSU)[1]
thresh = cv2.merge((thresh,thresh,thresh))
thresh = cv2.cvtColor(thresh, cv2.COLOR_BGR2GRAY)
thresh = thresh[y:y+h, x:x+w]
contours = cv2.findContours(thresh.copy(), cv2.RETR_TREE,
cv2.CHAIN_APPROX_NONE)[0]
if len(contours) > 0:
contour = max(contours, key = cv2.contourArea)
```

```
#print(cv2.contourArea(contour))
if cv2.contourArea(contour) > 10000:
x1, y1, w1, h1 = cv2.boundingRect(contour)
save_img = thresh[y1:y1+h1, x1:x1+w1]
if w1 > h1:
save_img = cv2.copyMakeBorder(save_img, int((w1-
h1)/2) , int((w1-h1)/2) , 0, 0, cv2.BORDER_CONSTANT, (0, 0, 0))
elif h1 > w1:
save_img = cv2.copyMakeBorder(save_img, 0, 0,
int((h1-w1)/2) , int((h1-w1)/2) , cv2.BORDER_CONSTANT, (0, 0, 0))
pred_probab, pred_class = keras_predict(model, save_img)
if pred_probab*100 > 80:
text = get_pred_text_from_db(pred_class)
print(text)
blackboard = np.zeros((480, 640, 3), dtype=np.uint8)
splitted_text = split_sentence(text, 2)
put_splitted_text_in_blackboard(blackboard, splitted_text)
#cv2.putText(blackboard, text, (30, 200), cv2.FONT_HERSHEY_TRIPLEX,
1.3, (255, 255, 255))
cv2.rectangle(img, (x,y), (x+w, y+h), (0,255,0), 2)
res = np.hstack((img, blackboard))
cv2.imshow("Recognizing gesture", res)
cv2.imshow("thresh", thresh)
if cv2.waitKey(1) == ord('q'):
break
keras_predict(model, np.zeros((50, 50), dtype=np.uint8))
recognize()
```

CHAPTER 6
TESTING

Software testing is an investigation conducted to provide stakeholders with information about the quality of the software product or service under test. Software testing can also provide an objective, independent view of the software to allow the business to appreciate and understand the risks of software implementation. Test techniques include the process of executing a program or application with the intent of finding software bugs (errors or other defects), and verifying that the software product is fit for use.

Software testing involves the execution of a software component or system component to evaluate one or more properties of interest. In general, these properties indicate the extent to which the component or system under test

meets the requirements that guided its design and development,

responds correctly to all kinds of inputs,

performs its functions within an acceptable time,
is sufficiently usable,
can be installed and run in its intended environments
As the number of possible tests for even simple software components is practically infinite, all software testing uses some strategy to select tests that are feasible for the available time and resources. As a result, software testing typically (but not exclusively) attempts to execute a program or application with the intent of finding software bugs (errors or other defects). The job of testing is an iterative process as when one bug is fixed, it can illuminate other, deeper bugs, or can even create new ones. Software testing can provide objective, independent information about the quality of software and risk of its failure to users or sponsors. Software testing can be conducted as soon as executable software exists. The overall approach to software development often determines when and how testing is conducted. For example, in a phased process, most testing occurs after system requirements have been defined and then implemented in testable programs. In contrast, under an agile approach, requirements, programming, and testing are often done concurrently.

6.1 TEST APPROACH

A test strategy is an outline that describes the testing approach of the software development cycle. It is created to inform project managers, testers, and developers about some key issues of the testing process. Test strategies describe how the product risks of the stakeholders are mitigated at the test-level, which types of testing are to be performed, and which entry and exit criteria apply. They are created based on development design documents. System design documents are primarily used and occasionally, conceptual design documents may be referred to. Design documents describe the functionality of the software to be enabled in the upcoming release. For every stage of development design, a corresponding test strategy should be created to test the new feature sets.

6.2 TEST PLAN

Unit testing is a software testing method by which individual units of source code, sets of one or more computer program modules together with associated control data, usage procedures, and operating procedures, are tested to determine whether they are fit for use. Intuitively, one can view a unit as the smallest testable part of an application. In procedural programming, a unit could be an entire module, but it is more commonly an individual function or procedure. For this we had divided into multiple parts limiting the code lines per part. Each of these parts was tested separately checking if there are any syntactical errors present or not.

6.3 FEATURES TO BE TESTED

Various features which we tested are as follows:
Dataset checking
The dataset should be compatible with the system. If an erroneous dataset is used the system may crash or produce invaluable outputs.
Compatibility of your system is a very important testing aspect. The model should be compatible and easy to embed with the existing company systems.
Web Cam testing
The primary reason for testing the web cam of a system is to identify a source of video stream input.

6.4 FEATURES NOT TO BE TESTED

Since the model is created on a python backend most of the code is to be tested before deploying. No external code template, APIs or links are used which could be pardoned while performing tests. Each section of the code is equally important and vulnerable to errors if not

properly tested hence a thorough test method is to be applied. The only thing to make sure is that the model shouldn't be tested on improper dataset as this may disrupt the model and produce erroneous results.

CHAPTER 7
FUNCTIONAL & NON FUNCTIONAL REQUIREMENT

7.1 PERFORMANCE REQUIREMENTS

The Sign Language Orator is based on python and has to be run from a host that has a webcam. The program shall take initial load time depending on internet connection strength. The performance shall also depend upon hardware components of the client.

7.2 RELIABILITY

Python is very reliable. The language has been around for over 20 years, and is in use in tens of thousands if not more sites. Python is actively maintained, so when problems are reported, they are dealt with promptly. But of course it is a programming language, which means the reliability of code you write depends on your skill at programming. If you write buggy code, Python cannot save you from your own errors.

7.3 RESPONSE TIME

Response time is the total amount of time it takes to respond to a request for service. That service can be anything from a loading of dataset to feedback from selected video source. Ignoring transmission time for a moment, the response time is the sum of the service time and wait time.

7.4 ROBUSTNESS

Robustness is the ability of a computer system to cope with errors during execution. Robustness can also be defined as the ability of an algorithm to continue operating despite abnormalities in input, calculations, etc. Robustness can encompass many areas of computer science, such as robust programming, robust machine learning, and Robust Security Network, the more robust the software. Formal techniques, such as fuzz testing, are essential to showing robustness since this type of testing involves invalid or unexpected inputs. Alternatively, fault injection can be used to test robustness. The Sign Language Orator will have robustness because it is based on python and whenever error will come in this it will show what wrong happened.

7.5 SCALABILTY

Python is the fastest-growing data science language and is used in production at many of the Fortune 500 companies for everything from software engineering to data engineering to rapid analytics. Despite its easy-to-learn nature and its simple syntax, Python packs surprising amounts of power and performance right out of the box. For instance, many of the newest innovations in the big data ecosystem, such as columnar storage, dataflow programming, and stream processing, can all be expressed in a relatively straightforward manner using Python. Unfortunately, it is also very easy to implement Python in ways that impede its ability to scale. For instance, many Hadoop practitioners fail to consider the implications of serialization overhead when interfacing with tools like R and Python. Others may simply be unaware of the facilities in Python to manage multicore and larger-than-memory workloads and assume that they have to move to complex distributed computing the instant they hit a memory barrier.

7.6 SECURIITY

There is nothing inherently insecure about python. However Python can be used to make a vulnerability. For instance it would be easy to make a web portal that executes anything that is typed in. This is true for any programming language, but python makes it extra easy. And I know, *you* would know better, but IT departments see all their users as idiots (often with good reason).

7.7 STABILITY

Python is very stable. New, stable releases have been coming out roughly every 6 to 18 months since 1991, and this seems likely to continue. Currently there are usually around 18 months between major releases.

With the introduction of retrospective "bug fix" releases the stability of existing releases is being improved. Bug fix releases, indicated by a third component of the version number (e.g. 2.1.3, 2.2.2), are managed for stability; only fixes for known problems are included in a bug fix release, and it's guaranteed that interfaces will remain the same throughout a series of bug fix releases.

7.8 SUPPORTABILITY

Python is not only popular but it is used broadly based on surveys conducted by the GitHub. Python could run on almost all operating systems either it is major or minor. Even a plenty of libraries and APIs have Python wrappers or bindings, letting Python interface to integrate freely with those APIs and libraries. The only disadvantage of the programming language is its speed. Python is not the fastest programming language still it is versatile and adopted broadly.

7.9 TESTABILITY

Test strategies describe how the product risks of the stakeholders are mitigated at the test-level, which types of tests are to be performed, and which entry and exit criteria apply. They are created based on development design documents. System design documents are primarily used and occasionally, conceptual design documents may be referred to. Design documents describe the functionality of the software to be enabled in the upcoming release. For every stage of development design, a corresponding test strategy should be created to test the new feature sets.

7.10 FAULT TOLERANCE

Fault tolerance is the property that enables a system to continue operating properly in the event of the failure of some (one or more faults within) of its components. If its operating quality decreases at all, the decrease is proportional to the severity of the failure, as compared to a naively designed system in which even a small failure can cause total breakdown. Fault tolerance is particularly sought-after in high-availability or life-critical systems.

CHAPTER 8
CONCLUSION

8.1 CONCLUSION

This is to conclude that the project that we undertook was worked upon with a sincere effort. Most of the requirements have been fulfilled up to the mark and the requirements which have been remaining, can be completed with a short extension. This project would definitely satisfy all the needs and help to the community of deaf people and people who are disabled to speak.

8.2 FUTURE ENHANCEMENT

A large number of identifiable nonverbal communication through series of videos.
An android application to run this on.
Make it portable.
Good GUI.

APPENDIX A: GLOSSARY

SR NO.	WORDS	MEANINGS
1	Source Webcam	Camera that is attached to the personal computer or laptop
	IP Webcam	This is a remote camera that sends a video feed to the source machine (on which the algorithm runs) through Internet.
	Database	A collection of interrelated data values that may be integrated permanently into a single connected structure or integrated temporarily for each interrogation, known as a query.
	GUI	Graphical User Interface. Defines a format for scroll bars, buttons, menus, etc., and how they respond to the user.
	Error message	A message that reports the detection of an error.
	Execute	To interpret a computer instruction and carry out the operations specified in the instruction.
	Crash	A computer system is said to crash when it stops working for some reason and must be restarted.
	Database	A collection of interrelated data values that may be integrated permanently into a single connected structure or integrated temporarily for each interrogation, known as a query.

	Neural Network	A computer system modelled on the human brain and nervous system.
	Utility	A specialized program that performs a frequently required everyday task such as sorting, report program generation, or file updating.
	ANN	It stands for Artificial Neural Network. An Artificial Neural Network is an information processing paradigm which is inspired by the way biological nervous systems, such as how the brain processes an information. It comes under the category of supervised learning models.

Table:- Glossary

USER MANUAL

Admin can only upload the database and access the software.
The graph can be shown created by SOM.
Admin can select the number of epochs for which the model would be trained. The more

the number of epochs the more the accuracy of the model.
User can have organized list of potential frauds.
Admin can check fraud possibility for a single customer.
Admin can change the training dataset.

APPENDIX B:
REFRENCES

[1] Priyanka Mekala, Ying Gao, Jeffrey Fan, Asad Davari “Real-time Sign Language Recognition based on Neural Network Architecture“April 2011 International Journal of Information and Communication Technology
[2] Suharjitoa*, Ricky Andersonb , Fanny Wiryanab , Meita Chandra Ariestab , Gede Putra Kusumaa “Sign Language Recognition Application Systems for Deaf-Mute People: A Review Based on Input-Process-Output” Suharjitoa*, Ricky Andersonb , Fanny Wiryanab , Meita Chandra Ariestab. ScienceDirect Procedia Computer Science 116 (2017) 441–448
[3] Tharadevi. R.V “Hand Gesture Recognition using Sign Language” Imperial Journal of Interdisciplinary Research (IJIR) Vol-3, Issue-9, 2017 ISSN: 2454-1362
[4] Fabio Fabris, Joa˜o Pedro de Magalha˜es, Alex A. Freitas. “A review of supervised machine learning research” 28 October 2016 / Accepted: 21 February 2017.
[5] Osisanwo F.Y., Akinsola J.E.T., Awodele O., Hinmikaiye J. O., Olakanmi O.,Akinjobi J. “Supervised Machine Learning Algorithms: Classification and Comparison” International Journal of Computer Trends and Technology (IJCTT) – Volume 48 Number 3 June 2017.
[6] Rashmi B. Hiremath, Ramesh M. Kagalkar “Review Paper on Sign Language Recognition Techniques”. International Journal of Computer Applications (0975 – 8887) National Conference on Advances in Computing (NCAC 2015) .
[7] Esam Kamal MARIED, Mansour Abdalla ELDALI ,Osama Omar ZIADA,” A Literature Survey of Deep learning and its application in Digital Image Processing”, UNIVERSITY OF TURKISH AERONAUTICAL ASSOCIATION DEPARTMENT OF ELECTRICAL AND ELECTRONICS ENGINEERING : (EEE 573),available : https://www.researchgate.net/publication/317346042_A_Literature_Study_of_Deep_learning_and_its_application_in_Digital_Image_Processing
[8] Akshay Jadhav, Gayatri Tatkar, Gauri Hanwate, Rutwik Patwardhan. “Sign Language Recognition” International Journal of Advanced Research in Computer Science and Software Engineering.

[9] Ms. Sonali. B. Maind, Ms. Priyanka Wankar “Research Paper on Basic of Artificial Neural Network” International Journal on Recent and Innovation Trends in Computing and Communication ISSN: 2321-8169 Volume: 2 Issue: 1.
[10] OpenCV Documentation https://opencv.org/about.html
[11] Theodoros Evgeniou and Massimiliano Pontil “SUPPORT VECTOR MACHINES: THEORY AND APPLICATIONS”. https://www.researchgate.net/publication/221621494_Support_Vector_Machines_Theory_and_Applications.

BIBLIOGRAPHY

[1] Learning Python – O’Reilly
[2] John A. Bullinaria,” Self-Organizing Maps: Fundamentals”, Introduction to Neural

Networks: Lecture 16,2004
[3] Self-organizing maps- https://www.springer.com/in/book/9783540679219
[4] Krishna Kumar Tripathi , Mahesh A. Pavaskar," Survey on Credit Card Fraud Detection Methods", International Journal of Emerging Technology and Advanced Engineering Website: www.ijetae.com (ISSN 2250-2459, Volume 2, Issue 11, November 2012)Noam Nisan, "The Elements of Computing Systems" Shimon Schocken - MIT Press , 2005
[5] Hands-On Machine Learning with Scikit-Learn and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems - Aurélien Géron
[6] Python GUI Programming with Tkinter: Develop Responsive and Powerful GUI Applications with Tkinter – Alan D Moore