

# Project Report

## A Real Time Communication System Powered By AI For Specially Abled

**Team ID:** PNT2022TMID16504

### **Team Members:**

- P Mukesh Kumar
- Naveenakrishnan S
- Ganeshamurthy K
- Sharath Kumar B

# CONTENTS

## **1.INTRODUCTION**

- 1.1 Project Overview
- 1.2 Purpose

## **2.LITERATURE SURVEY**

- 2.1 Existing problem
- 2.2 References
- 2.3 Problem Statement Definition

## **3.IDEATION & PROPOSED SOLUTION**

- 3.1 Empathy Map Canvas
- 3.2 Ideation & Brainstorming
- 3.3 Proposed Solution
- 3.4 Problem Solution fit

## **4.REQUIREMENT ANALYSIS**

- 4.1 Functional requirement
- 4.2 Non-Functional requirements

## **5.PROJECT DESIGN**

- 5.1 Data Flow Diagrams
- 5.2 Solution & Technical Architecture
- 5.3 User Stories

## **6.PROJECT PLANNING & SCHEDULING**

- 6.1 Sprint Planning & Estimation
- 6.2 Sprint Delivery Schedule

## 6.3 Reports from JIRA

## **7.CODING & SOLUTIONING**

### 7.1 Feature 1

### 7.2 Feature 2

### 7.3 Database Schema (if Applicable)

## **8.TESTING**

### 8.1 Test Cases

### 8.2 User Acceptance Testing

## **9.RESULTS**

### 9.1 Performance Metrics

## **10.ADVANTAGES & DISADVANTAGES**

## **11.CONCLUSION**

## **12.FUTURE SCOPE**

## **13.APPENDIX**

### 13.1 Source Code

### 13.2 GitHub & Project Demo Link

# 1. INTRODUCTION

## 1.1 Project Overview

The project's objective is to create an interactive system that can convert spoken English into understandable sign language for the deaf and dumb to communicate with common people.

To train a model on various hand movements, a convolution neural network is being employed. On top of this paradigm, a framework is created. Using signs that are translated into speech and words that people can understand, deaf or dumb users can communicate using this software.

This framework consists of a robust and user-friendly UI written in Python Flask that allows the user to switch between modes such as Speech to Sign and vice versa. Users can send the exact messages they want to convey with the help of a specially trained neural network. We will therefore make sure that this framework will aid in closing the communication gap between these individuals.

## **1.2 Purpose**

People who are visually impaired face unique obstacles when it comes to doing daily activities and chores such as ascending the stairs, using public transportation without the assistance of others, and many others. This category also includes people who have hearing loss and were not born with the ability to speak (specially abled). They encounter a number of difficulties, the most significant of which is a lack of communication with others. Although sign language can be used to communicate, only a small number of people use it due to a lack of awareness.

On account of sign language translation, an innovative approach that would effectively help the underprivileged has been presented. This service, which can function without an internet connection, can translate in real time from English to sign language, greatly assisting in closing the communication gap.

## **2. LITERATURE SURVEY**

### **2.1. Existing problem:**

When it comes to managing their daily activities and chores, people with special needs have various difficulties. These include climbing stairs, using public transportation alone, and a host of other things. This category also includes those who have hearing loss and lack the gift of speech (specially-abled). Numerous difficulties exist for them, but the absence of interpersonal communication is the biggest one. Although sign language can be used to communicate, due to a lack of awareness, relatively few people actually use it. Following are the existing systems which utilise various kinds of technology.

#### **2.1.1. 5DT Data Glove**

The following system uses position-sensing 5DT gloves and other machine learning techniques to convert hand movements made in sign language into words. There are fourteen high-precision joint-angle measurement sensors in the 5DT data glove. Data-gloves' direct measurement of finger movements and the abduction angle between the fingers are the major factors in this decision. Finger flexure and finger abduction are measured with the 5DT Data Glove 14 Ultra, which has two sensors per finger.

#### **Pros:**

- The designed system is precise and accurate.

#### **Cons:**

- The system's high level of sophistication and the use of various electronic components make it less portable.

### 2.1.2. Dynamic Interpreter

The system employs an interpreter that can reliably classify the letters a-z in American Sign Language. The system first converts videos into frames, which are then pre-processed to create greyscale images. Each frame is stored with a distinct frame ID and has a frame rate of one per second. To detect hand motions, skin filtering technology is applied to the input frames. The Scale Invariant Feature Transform (SIFT) characteristics of various ASL alphabets are used to construct a Convolutional Neural Network (CNN) classification model.

#### Pros:

- Training the model can be easily done using videos .

#### Cons:

- It is only limited to ASL.

### 2.1.3. Sign Language Translator using the Back Propagation Algorithm of an MLP.

Abou Haidar, G., Achkar, R., Salhab, D., Sayah, A., & Jobran, F. (2019)

**DOI:** 10.1109/ficloudw.2019.00019

The system is implemented with the help of a mobile app and a back-end procedure that operates on a dedicated server. The primary job of the mobile application is to accept data from the gloves and pass it to the back-end procedure. To process the input and release the output, the latter employs the MLP's (Multi-Layer Perceptron) Back-Propagation method.

#### Pros:

- a. The use of flex sensors make it a very cost effective system
- b. Back propagation algorithm is a very simple and easy to use algorithm that yield accurate results.

**Cons:**

- a. Flex sensors are prone to malfunction under moving and warping garments.
- b. The performance of a back-propagation algorithm is determined by its input data.

**2.1.4. Microsoft's kinect v2:**

The following system utilizes an innovative strategy for bridging the communication gap between hearing persons and everyday people. The system provides a dual way of communication between sign language speakers and speakers of natural language. It serves as a sign language interpreter and translator. It makes advantage of the most recent Microsoft Kinect for Windows V2 technology through Kinect's field of view (FOV), the system gets the gestures that are being executed. It then understands the motions by comparing them to the gestures that have already been pre-stored.

**Pros:**

- a. Kinect can easily detect the exact dimensions of the human body which can prove useful for gesture detection.

**Cons:**

- a. Can only be used in indoor applications, does not work in outdoor areas
- b. The distance depth detection of a Kinect is limited.

**2.1.5. Skin Segmentation and Image Category Classification with Convolutional Neural Network and Deep Learning:**

The existing system presents a method for automatically segmenting human skin based on color data. The YCbCr colour space is selected because it is frequently used in video coding and offers



a useful way to use microfinance data to mimic the colour of human skin. In the CbCr plane, it models the distribution of skin tones as a bi-variate normal distribution. By simulating the behavior of the algorithm on photos of people of various ethnic backgrounds, it is possible to demonstrate how well it performs. Following that, Deep Learning Method is used to train a classifier to recognize Sign Language, and Convolutional Neural Network (CNN) is utilized to extract features from the photos.

**Pros:**

- a. Skin Segmentation can identify gestures with precision as it is primarily used for face recognition.

**Cons:**

- a. Cannot differentiate between skin and skin color backgrounds.
- b. Results might not be consistent.

## **2.2. References:**

**1. Vijayalakshmi, P., & Aarthi, M. (2016). Sign language to speech conversion. 2016 International Conference on Recent Trends in Information Technology (ICRTIT).**

**DOI:**10.1109/icrtit.2016.7569545

**2. Abou Haidar, G., Achkar, R., Salhab, D., Sayah, A., & Jobran, F. (2019). Sign Language Translator using the Back Propagation Algorithm of an MLP.**

**DOI:** 10.1109/ficloudw.2019.00019

**3. Rosero-Montalvo, P. D., Godoy-Trujillo, P., Flores-Bosmediano, E., Carrascal-Garcia, J., Otero-Potosi, S., Benitez-Pereira, H., & Peluffo-Ordonez, D. H. (2018).**

**DOI:** 10.1109/etcm.2018.8580268

**4. Priya, L., Sathya, A., & Raja, S. K. S. (2020). Indian and English Language to Sign Language Translator- an Automated Portable Two Way Communicator for Bridging Normal and Deprived Ones.**

**DOI :** 10.1109/icpects49113.2020.9336983

**5. Rahaman, M. A., Jasim, M., Ali, M. H., & Hasanuzzaman, M. (2014). Real-time computer vision-based Bengali Sign Language recognition. DOI: 10.1109/iccitechn.2014.70731**

## **2.2. Problem Statement:**

There are handicapped people in our society. Although technology is constantly evolving, little is being done to improve the lives of these people. It has always been difficult to communicate with someone who is deaf-mute.

It is quite challenging for mute individuals to communicate with non-mute people as hand sign language is not taught to the general public. It might be quite challenging for them to communicate at times of crisis. In circumstances where other modes of communication, like speech, are not possible, the human hand has remained a common alternative for information transmission.

To have a proper communication between a normal person and a handicapped person in any language, a voice conversion system with hand gesture recognition and translation will be very helpful.

The project's objective is to create an interactive system that can convert spoken English into understandable sign language for the deaf and dumb as well as Sign Language into a human hearing voice in English to communicate with common people.

## 3. IDEATION AND PROPOSED SOLUTION

### 3.1. Empathy Map Canvas

An empathy map is a straightforward, simple-to-understand picture that summarises information about a user's actions and views.

It is a helpful tool that enables teams to comprehend their users more fully. It's important to comprehend both the actual issue and the individual who is experiencing it in order to develop a workable solution. Participants learn to think about situations from the user's perspective, including goals and problems, through the exercise of constructing the map.

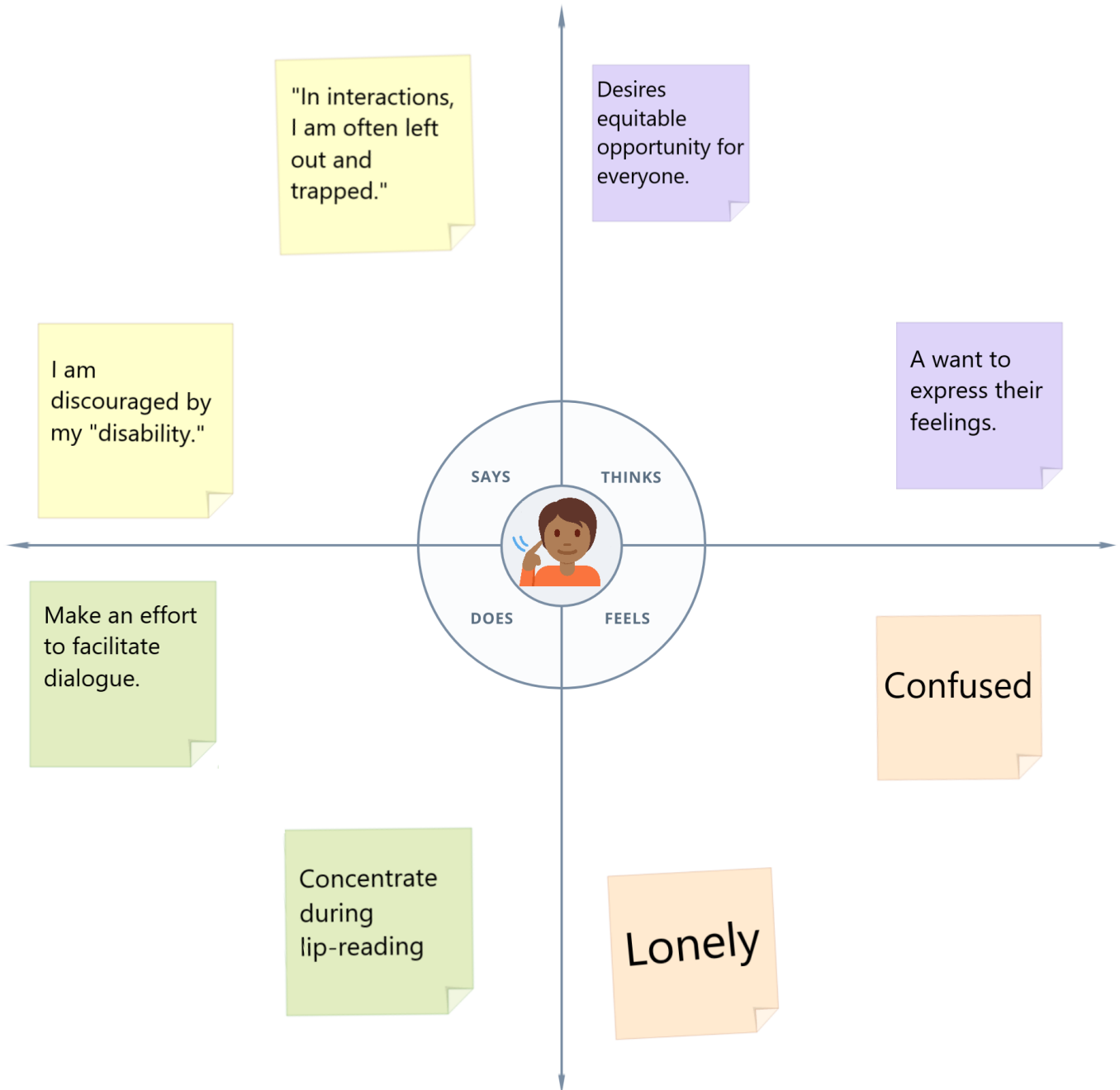
Empathy maps, which are neither chronological nor sequential, offer a glimpse into a user's character as a whole. The user's actual spoken words during an interview or other usability study are recorded in the Says quadrant.

It should ideally include exact quotes from studies. What the user is thinking during the encounter is captured in the Thinks quadrant. The user's emotional state is conveyed in the Hear quadrant by an adjective and a brief statement for context. The user's actions are contained within the See quadrant.

Any qualitative research technique can serve as the basis for empathy mapping. They can assist team members in determining which aspects of their user they are already familiar with and which areas require additional user data collection. It is best to employ empathy maps right away throughout the design phase. The project gains significant advantages from both the empathy map-making process and the final product.

# EMPATHY MAP

Real Time Communication System for the Specially Abled

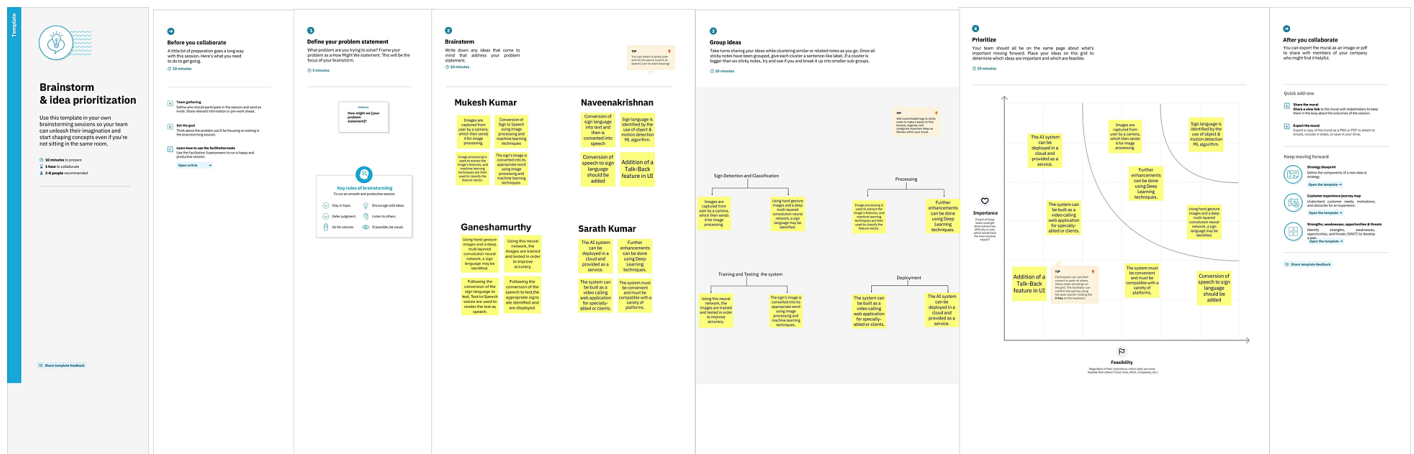


## 3.2. Ideation & Brainstorming

Ideation is the process through which you come up with concepts and answers using tools like the empathy map canvas, brainstorming, etc. The third step of the Design Thinking process is also known as ideation.

Brainstorming can be used to come up with potential answers to straight forward issues, it is unreasonable to expect it to solve most issues or fulfil most planning requirements. The investigation phase of a new project begins with brainstorming, therefore it's critical to be open to all ideas and options.

During a brainstorming session, everyone on a team is encouraged to engage in the process of original thought that results in problem solving. Volume over quality is prioritised, unconventional ideas are welcomed and developed upon, and everyone is urged to participate in order to produce a wealth of original solutions.



### 3.3. Proposed Solution

This paper describes a system that solves the problem of speech and hearing impairment. The following are the study's objectives:

1. To detect the sign language (ISL) using gesture detection algorithm.
2. To recognise the speech using speech recognition algorithm.
3. To convert the ISL to English / Tamil language and vice versa.
4. To create an interactive and responsive User Interface.

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	To develop an interactive communication system for people with special needs that converts sign language to voice in real time.
2.	Idea / Solution description	<p>1. At first, we will pre-process the images which will be used for building the model. Image pre-processing includes zooming, shearing, flipping to increase the robustness of the model after it is built. The Keras package will be utilized for picture pre-processing.</p> <p>2. Then, we start building our model by initializing the model, adding Convolution layers, adding Pooling layers, Flatten layer and Full connection layers which include hidden layers.</p> <p>3. We compile the model with layers that we added to complete the neural network structure.</p> <p>4. We create an interactive and responsive User Interface.</p>
3.	Novelty / Uniqueness	<p>1. We intend to create a communication system that is robust and user friendly so that both normal and specially abled people can access it freely.</p> <p>2. The model is trained by the latest available convolution neural network with maximum efficiency.</p>

4.	Social Impact / Customer Satisfaction	This allows specially abled people to interact freely and not be dependant on anybody. It also bridges the communication gap which allows specially abled people to integrate into normal society.
5.	Business Model (Revenue Model)	The system will be marketed as a web app with freemium features, starting with a free trial period for a specific period of time. Once this period ends, access to the service can be renewed for a marginal price.
6.	Scalability of the Solution	Initially launching the firm with venture capital, but later reinvesting profits obtained from subscription service.

### 3.3. Problem Solution Fit

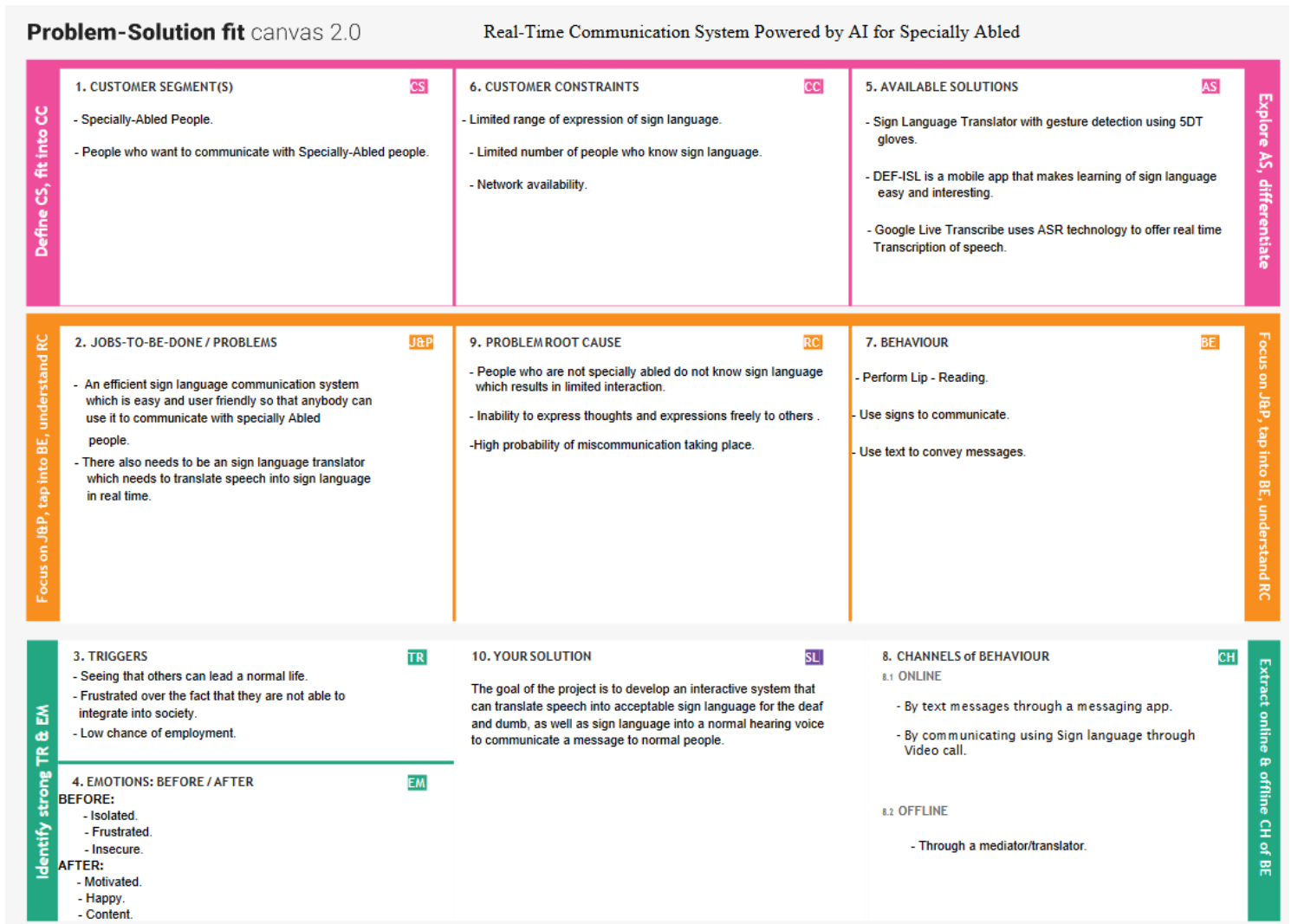
The Problem-Solution Relationship Fit simply means that you identified a problem with your customer and that the solution you devised solves the customer's problem. It assists entrepreneurs, marketers, and corporate innovators in identifying behavioral patterns and determining what works and why.

#### Purpose:

- Solve complex problems in a way that is appropriate for your customers' current situation.
- Tap into existing mediums and channels of behaviour to accelerate success and increase solution adoption.
- With the right triggers and messaging, you can sharpen your communication and marketing strategy
- Increase touch-points with your company by identifying the best problem-behavior fit

and establishing trust by resolving common annoyances, as well as urgent or costly problems.

- Understand the current situation in order to improve it for your target audience.





## 4. REQUIREMENT ANALYSIS

### 4.1. Functional Requirements:

Functional Requirements are the requirements that the end user expressly requests as basic services that the system provide. All of these functions must be incorporated into the system as part of the design. These are represented or stated in the form of input to the system, operation performed, and expected output. In contrast to non-functional requirements, they are essentially the user-stated requirements that can be seen directly in the final design.

Following are the functional requirements of the proposed solution.

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration	Registration through Form. Registration through Gmail. Registration through LinkedIN.
FR-2	User Confirmation	Confirmation via Email. Confirmation via OTP.
FR-3	System Prerequisites	Webcam with a high resolution on a desktop. Microphone to record audio input. Audio output is provided by speakers.
FR-4	Authorization Levels	In terms of authorization, there are two levels: standard access level and advanced access level.
FR-5	Real-time Translator	In real time, it translates sign language into speech and vice versa.
FR-6	User Interface	Easy to use with an intuitive design along with a talkback feature.

## 4.2. Non Functional Requirements:

Non-functional requirements are essentially the quality restrictions that the system must meet in order to follow the design plan. Depending on the project, these criteria may be prioritised differently or applied to a different degree. Additionally known as non-behavioral requirements. Basically, they deal with the following factors:

- Usability.
- Security.
- Reliability.
- Performance.
- Availability.
- Scalability.

Following are the non-functional requirements of the proposed solution.

FR No.	Non-Functional Requirement	Description
NFR-1	<b>Usability</b>	The system must be simple to use and accessible for everyone, including those with special needs.
NFR-2	<b>Security</b>	Only the authorised user has access to the information present in the application.
NFR-3	<b>Reliability</b>	For people who need it, such as those with special needs, the application must prove to be a necessary tool.
NFR-4	<b>Performance</b>	The translation system needs to be quick and responsive.
NFR-5	<b>Availability</b>	It must be accessible at all times and across several platforms.
NFR-6	<b>Scalability</b>	The application framework can be modified and developed in many ways like adding regional language support.

## 5. PROJECT ANALYSIS

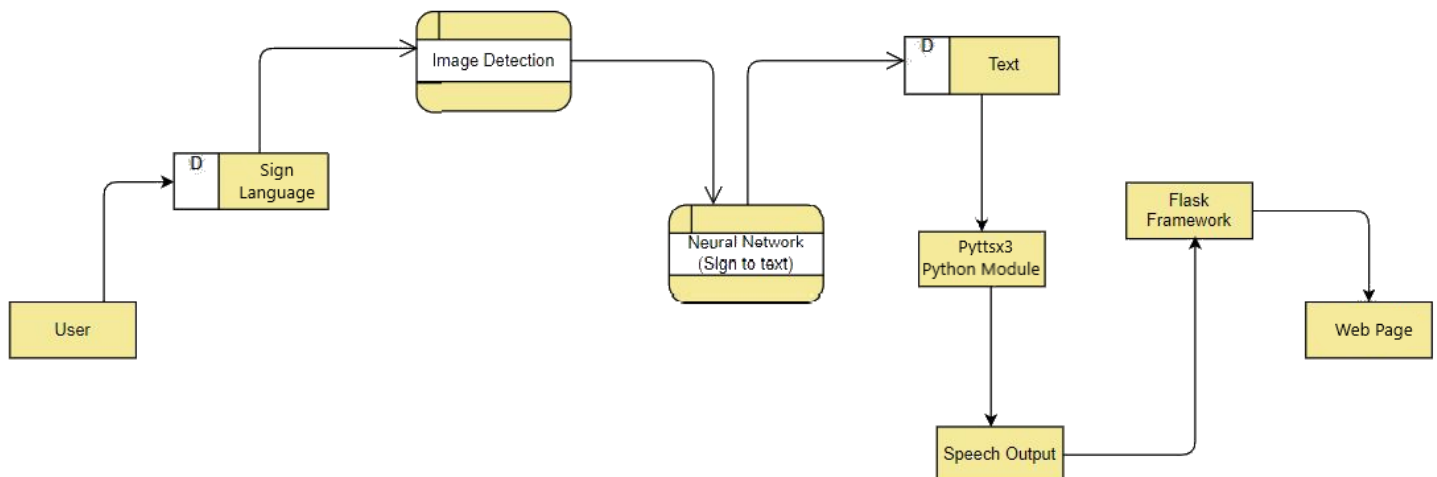
### 5.1. Data Flow Diagram:

A data flow diagram is the standard graphical illustration of how information flows through a system (DFD). A clear and unambiguous DFD can graphically express the right amount of system need. It illustrates how data enters and departs the system, where it is stored, and what changes the data.

It makes it simpler to comprehend how the design that is being built will flow.

It displays data inputs, outputs, storage locations, and routes between each destination using predefined symbols such rectangles, circles, and arrows as well as brief text labels.

The design team can quickly comprehend the flow thanks to those symbols. According to the specifications, we created these data flow diagrams.



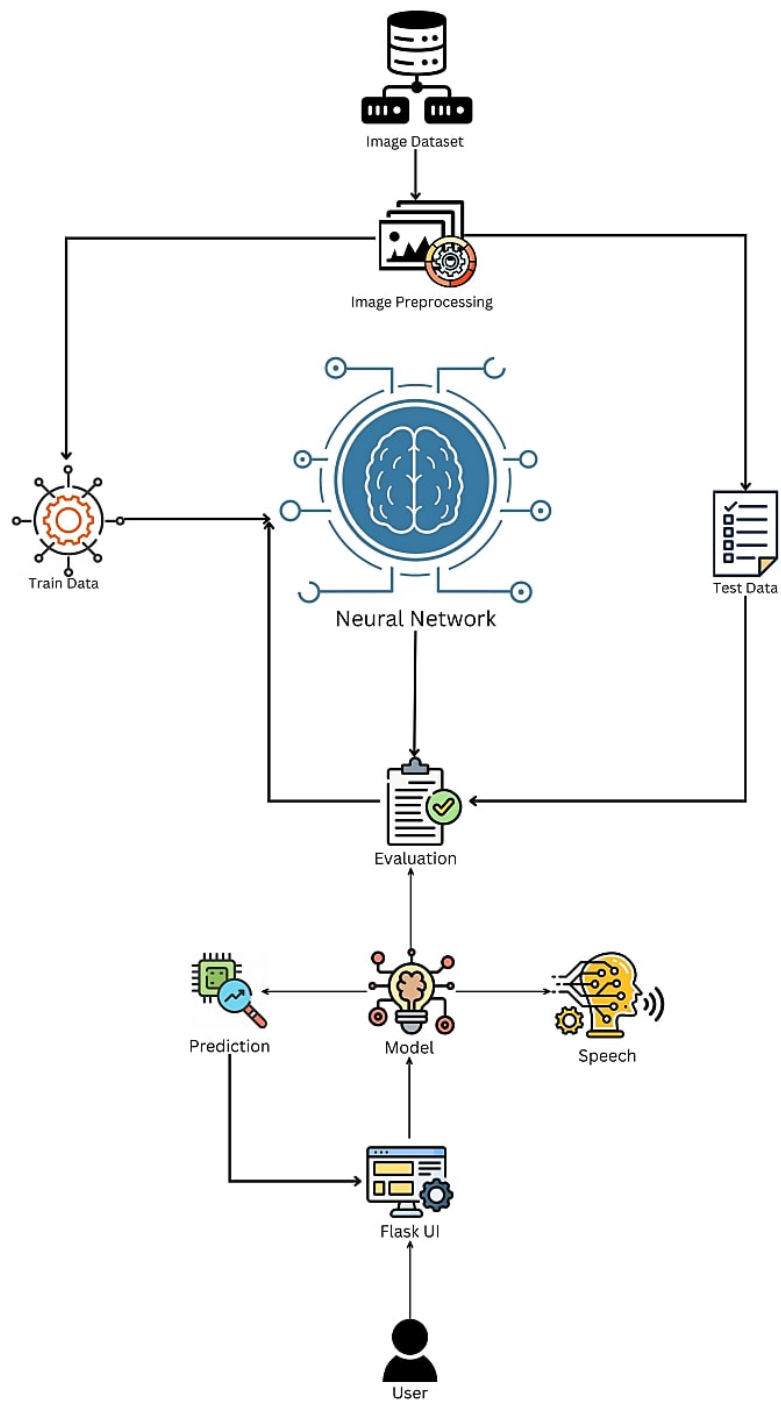
## **5.2. Solution And Technical Architecture:**

### **5.2.1 Solution Architecture:**

Solution architecture is the process of creating solutions using established procedures, rules, and best practises with the goal of ensuring that the created solution is compatible with the enterprise architecture in terms of information architecture, system portfolios, integration needs, and other factors.

The mix of roles, procedures, and documentation can thus be seen as being used to build and develop applications and information systems in order to handle certain business needs, requirements, or challenges. Its objectives are to:

- Find the finest technological solution to address current business issues.
- Explain to project stakeholders the structure, qualities, behaviour, and other features of the software.
- Define the solution's requirements, development stages, and features.
- Give guidelines for how the solution will be defined, managed, and delivered.



## 5.2.2 Technical Architecture:

Systems are designed using technical architecture, a type of information technology (IT) architecture. It entails creating a technical blueprint for how all components should be arranged, work together, and depend on one another in order to satisfy system-related criteria. The use of a technological architecture diagram is widespread because it offers precise information about how the system was designed and the components that were used.

Every component uses a different technology and has a unique application. The following are the elements that were utilised in this design:

S.No	Component	Description	Technology
1.	User Interface	User interacts with the neural network through Home page	HTML, CSS,
2.	Image Detection	The user data is captured by the camera in which the signs are interpreted from the input using Image detection.	Python,Keras, TensorFlow
3.	Neural Network	The detected signs are then sent to the neural network where the signs are converted to text	CNN,Image classification(Keras)
4.	File Storage	File storage requirements	IBM Cloud Storage
5.	Text to Speech API	The text is then converted to speech through an API	IBM Watson API or Pyttsx3 (Python Module)
6.	Machine Learning Model	Performs Image Classification and Recognition	CNN Model
7.	Infrastructure (Server / Cloud)	Application Deployment on Local System / Cloud Local Server Configuration: Cloud Server Configuration :	IBM Cloud

It is crucial to look at the design's qualities when creating a system.

The following characteristics should always be included in any design:

S.No	Characteristics	Description	Technology
1.	Open-Source Frameworks	The model is constructed using open-source frameworks. Techniques for picture categorization and gesture recognition are also employed with it.	Python Flask, OpenCV, Keras, Numpy, Pandas, Matplotlib.
2.	Security Implementations	List all the security / access controls implemented, use of firewalls etc.	IBM Watson cloud Security.
3.	Scalable Architecture	When needed, IBM Cloud Bare Metal servers assist in achieving scalability.	IBM Cloud.
4.	Availability	Global load balancing is used by IBM Cloud to guarantee the availability of a redundant, highly available platform to host the workloads and applications.	IBM Cloud
5.	Performance	Utilizing the data centre and IBM Cloud APM, workloads and cloud infrastructure are with cognitive intelligence, handled. Slowdowns and outages can be minimised and continuously avoided in a hybrid as Cloud APM supports the application world moving on from performance evaluation difficulty in identifying the source of the issue occurrences and identifying problems before the Affected application.	IBM Cloud APM

### 5.3. User Stories:

A user story is a casual, all-inclusive description of a design element written from the viewpoint of the end user. Its objective is to explain how a design will benefit the end user. Putting people first is a crucial aspect of agile software development, and a user narrative places end users at the forefront of the discussion. These narratives give context for the development team's work in non-technical terms.

The development team understands their goals, the nature of their work, and the value it adds after reading a user story.

One of the essential elements of an agile programme is the user story. They aid in creating a framework for daily work that is user-focused, which encourages cooperation, innovation, and better design in general.

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Mobile user)	Registration	USN-1	As customer, I could able to register for the app by entering my E-mail and proper password.	I could able to access my registered account.	High	Sprint 1
		USN-2	As a user, I'll get the acknowledgement verification email once after my registration has been done for the app	I can get verification email and click ok to confirm it..	High	Sprint 1
		USN-3	As a customer, I could able to register for application via their official websites and social media.	I could able to register and access my account by using their website & social media.	Medim	Sprint 2



		USN-4	As a customer, I could able to register for application through Gmail	via some third parties link	Low	Sprint 2
	Login	USN-5	As a customer, I could able to login into application by entering already registered email and password	I can type manually and also can use saved login credentials	High	Sprint 1
	Dashboard	USN-6	As a customer, I can get all services and help in dashboard	I can access my dashboard and change profile	Medium	Sprint 2
Customer (Webuser)	Registration	USN-7	As a customer, I could able to login through registered phone number by using otp instead of Gmail	I could able to register & login via phone number to access my account	High	Sprint 2
Customer Care Executive	Service	USN-8	Can avail the service by calling customer care or reaching through E-mail.	Can avail the service by calling customer care or reaching through E-mail.	Medium	Sprint 1
Administrator		USN-9	Respective person in the company should take care of all of this.	All the requirements are there.	High	Sprint 2
	Sign up	USN-10	Customer have to sign-up to use these things and all	Have to enter valid credentials.	High	Sprint 2

## 6. PROJECT PLANNING AND SCHEDULING

Planning and scheduling are two separate but equally important parts of project management. In order to accomplish the project's goals, the planning phase is largely concerned with choosing the necessary policies and procedures. The project action plans for scope, time, cost, and quality are transformed into an operating timeline through scheduling.

### 6.1 Sprint Planning And Estimation

The process for estimating the time needed to perform a prioritised item from the product backlog is called sprint planning and estimation. This effort is typically assessed in relation to the amount of time needed to finish the task, which enables precise sprint planning.

Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date(Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint-1	10	6 Days	24 Oct 2022	29 Oct 2022	8	29 Oct 2022
Sprint-2	10	6 Days	31 Oct 2022	04 Nov 2022	5	04 Nov 2022
Sprint-3	10	6 Days	07 Nov 2022	11 Nov 2022	7	11 Nov 2022
Sprint-4	10	6 Days	14 Nov 2022	18 Nov 2022	5	18 Nov 2022

Velocity:

$$AV = \frac{\text{sprint duration}}{\text{velocity}}$$

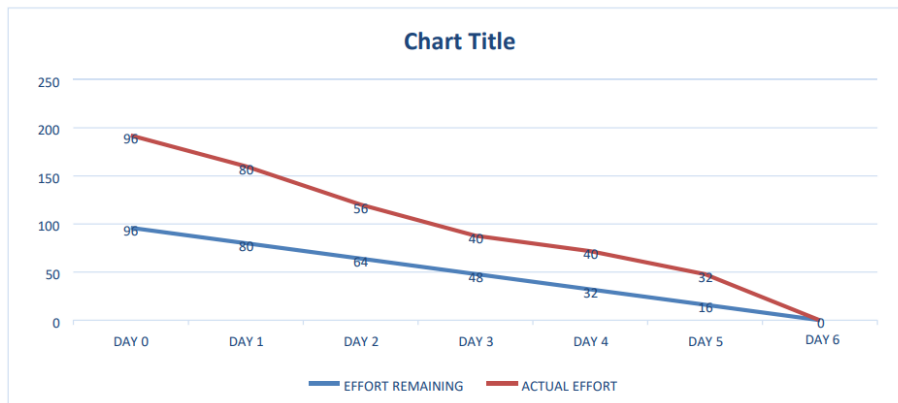
$$AV = 6/10 = 0.6$$

## 6.2 Sprint Delivery Schedule

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Data Collection	USN-1	Collect Data-set.	9	High	Naveenakrishnan S Ganeshamurthy K Sharath Kumar B P Mukesh Kumar
Sprint-1	Image Preprocessing	USN-2	Preprocessing the collected data-set	8	Medium	Naveenakrishnan S Ganeshamurthy K Sharath Kumar B P Mukesh Kumar
Sprint-2	Model Building	USN-3	Import the required libraries, add the necessary layers and compile the model	10	High	Naveenakrishnan S Ganeshamurthy K Sharath Kumar B P Mukesh Kumar
Sprint-2	Model Training	USN-4	Training the image classification model using CNN	7	Medium	Naveenakrishnan S Ganeshamurthy K Sharath Kumar B P Mukesh Kumar
Sprint-3	Model Testing	USN-5	Training the model and testing the model's performance	9	High	Naveenakrishnan S Ganeshamurthy K Sharath Kumar B P Mukesh Kumar
Sprint-4	Implementation of the application	USN-6	Converting the input sign language images into English alphabets	8	Medium	Naveenakrishnan S Ganeshamurthy K Sharath Kumar B P Mukesh Kumar

## 6.3 Reports from JIRA

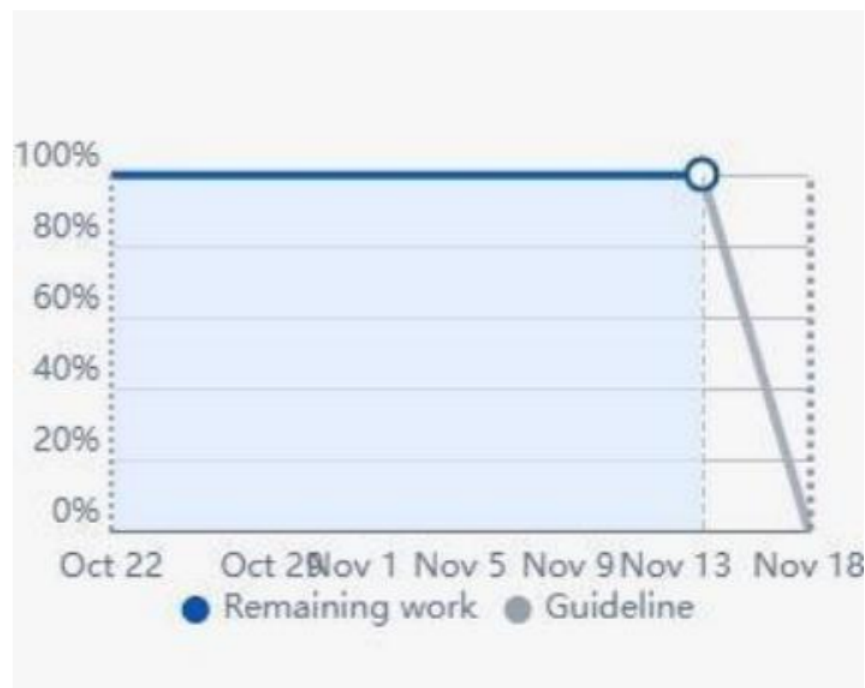
Burndown chart:



## Burndown Chart:

A burndown chart plots the amount of work remaining against the amount of time. In agile software development approaches like Scrum, it is frequently employed. Burn down charts, however, can be used for any project that makes observable progress over time.

### SPRINT BURNDOWN CHART:



	16	17	18	19
Sprints				
>  RTCSPBAFSA-7 Data Collection				
>  RTCSPBAFSA-10 MODEL BUILDNG				
>  RTCSPBAFSA-13 Training and Testing				
>  RTCSPBAFSA-15 Implementation of the application				

## **7. CODING AND SOLUTIONING**

### **7.1 Features And Methodology**

Our project phases are defined broadly in four categories, according to the scope of work involved in each: Data Collection, Model Training, Deployment of Model and Final Testing.

#### **PHASE 1: Data Collection**

This phase consists of collecting data for our model. We will make two folders: one for training and one for testing. The images in the training folder will be used to build the model, while the images in the testing folder will be used to validate our model.

We will pre-process the images which will be used for building the model. Image pre-processing includes zooming, shearing, flipping to increase the robustness of the model after it is built. The Keras package will be utilised for picture pre-processing.

---

## Image Preprocessing

### Import ImageDataGenerator Library And Configure It

```
In [1]: from tensorflow.keras.preprocessing.image import ImageDataGenerator

In [2]: import tensorflow as tf
import os
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Conv2D, Flatten, Dropout, MaxPooling2D
from tensorflow.keras.preprocessing.image import ImageDataGenerator
import numpy as np
import matplotlib.pyplot as plt
import IPython.display as display
from PIL import Image
import pathlib
```

### Apply ImageDataGenerator Functionality To Train And Test set

```
In [3]: train_datapath = "dataset/train_set/"
test_datapath = "dataset/test_set/"

In [4]: train_datagen = ImageDataGenerator(rescale=1./255, zoom_range=0.2, horizontal_flip=True, vertical_flip=False)

In [5]: test_datagen = ImageDataGenerator(rescale=1./255)

In [6]: x_train = train_datagen.flow_from_directory(train_datapath, target_size=(64,64), batch_size=300,
class_mode='categorical', color_mode = "grayscale")

Found 15750 images belonging to 9 classes.

In [7]: x_test = test_datagen.flow_from_directory(test_datapath, target_size=(64,64), batch_size=300,
class_mode='categorical', color_mode = "grayscale")

Found 2250 images belonging to 9 classes.

In [8]: x_train.class_indices

Out[8]: {'A': 0, 'B': 1, 'C': 2, 'D': 3, 'E': 4, 'F': 5, 'G': 6, 'H': 7, 'I': 8}

In [9]: x_test.class_indices

Out[9]: {'A': 0, 'B': 1, 'C': 2, 'D': 3, 'E': 4, 'F': 5, 'G': 6, 'H': 7, 'I': 8}
```

## PHASE 2: Machine Learning Model Training

We start building our model by initializing the model, adding Convolution layers, adding Pooling layers, Flatten layer and Full connection layers which include hidden layers. At last, we compile the model with layers we added to complete the neural network structure.

## Model Creation

```
In [6]: # Importing Libraries
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Convolution2D, MaxPooling2D,
from tensorflow.keras.layers import Flatten, Dense
```

```
In [7]: # Creating Model
model=Sequential()
```

```
In [8]: # Adding Layers
model.add(Convolution2D(32,(3,3),activation='relu',input_shape=(64,64,3)))
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Flatten())

# Adding Hidden Layers
model.add(Dense(300,activation='relu'))
model.add(Dense(150,activation='relu'))

# Adding Output Layer
model.add(Dense(9,activation='softmax'))
```

```
In [13]: model.summary()
```

Model: "sequential"

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 62, 62, 32)	896
max_pooling2d (MaxPooling2D)	(None, 31, 31, 32)	0
flatten (Flatten)	(None, 30752)	0
dense (Dense)	(None, 300)	9225900
dense_1 (Dense)	(None, 150)	45150
dense_2 (Dense)	(None, 9)	1359

=====  
Total params: 9,273,305  
Trainable params: 9,273,305  
Non-trainable params: 0  
=====

```
In [9]: # Compiling the Model
model.compile(loss='categorical_crossentropy',optimizer='adam',metrics=['accuracy'])
```

```
In [10]: # Fitting the Model Generator
model.fit_generator(x_train,steps_per_epoch=len(x_train),epochs=10,validation_data=x_test,validation_steps=len(x_test))
```

C:\Users\sname\AppData\Local\Temp\ipykernel\_3100\1042518445.py:2: UserWarning: `Model.fit\_generator` is deprecated and will be removed in a future version. Please use `Model.fit`, which supports generators.

```
model.fit_generator(x_train,steps_per_epoch=len(x_train),epochs=10,validation_data=x_test,validation_steps=len(x_test))
Epoch 1/10
18/18 [=====] - 29s 1s/step - loss: 0.9233 - accuracy: 0.6873 - val_loss: 0.3513 - val_accuracy: 0.8898
Epoch 2/10
18/18 [=====] - 17s 914ms/step - loss: 0.1772 - accuracy: 0.9458 - val_loss: 0.2505 - val_accuracy: 0.9542
Epoch 3/10
18/18 [=====] - 17s 923ms/step - loss: 0.0780 - accuracy: 0.9775 - val_loss: 0.1611 - val_accuracy: 0.9702
Epoch 4/10
18/18 [=====] - 17s 929ms/step - loss: 0.0404 - accuracy: 0.9890 - val_loss: 0.1985 - val_accuracy: 0.9760
Epoch 5/10
18/18 [=====] - 17s 930ms/step - loss: 0.0249 - accuracy: 0.9944 - val_loss: 0.2219 - val_accuracy: 0.9764
Epoch 6/10
18/18 [=====] - 17s 909ms/step - loss: 0.0177 - accuracy: 0.9961 - val_loss: 0.2412 - val_accuracy: 0.9773
Epoch 7/10
18/18 [=====] - 17s 920ms/step - loss: 0.0109 - accuracy: 0.9979 - val_loss: 0.2459 - val_accuracy: 0.9773
Epoch 8/10
18/18 [=====] - 17s 912ms/step - loss: 0.0100 - accuracy: 0.9975 - val_loss: 0.1562 - val_accuracy: 0.9782
Epoch 9/10
18/18 [=====] - 16s 907ms/step - loss: 0.0058 - accuracy: 0.9992 - val_loss: 0.2264 - val_accuracy: 0.9782
Epoch 10/10
18/18 [=====] - 16s 913ms/step - loss: 0.0041 - accuracy: 0.9992 - val_loss: 0.2430 - val_accuracy: 0.9782
```

Out[10]:

## Saving the Model

```
In [16]: model.save('ASL_Model.h5')
```

## PHASE 3: Testing The Model

By sending an image to the model and obtaining predictions, we test it. When testing the model, we must ensure that the test image will fit the model's goal dimensions and that it will be rescaled before being sent to the model.

We import packages that are used to load the model and extend the image's dimension as the initial stage in the prediction process. The model that was saved when we built the model is loaded using the Keras package.

The image is pre-processed by being converted to an array and resized in accordance with the model.

### Test the Model

```
In [1]: from tensorflow.keras.models import load_model
        from tensorflow.keras.preprocessing import image
        import numpy as np
        import os
        import cv2
```

```
In [2]: model = load_model('./ASL_Model.h5')
```

```
In [3]: img = image.load_img('./Dataset/test_set/E/200.png',target_size = (100,100))
        img
```



```
In [4]: from skimage.transform import resize
        def detect(frame):
            img=image.img_to_array(frame)
            img = resize(img,(64,64,3))
            img = np.expand_dims(img,axis=0)
            pred=np.argmax(model.predict(img))
            op=['A','B','C','D','E','F','G','H','I']
            print("THE PREDICTED LETTER IS ",op[pred])
```

```
In [5]: img=image.load_img("./Dataset/test_set/E/200.png")
        detect(img)
```

```
1/1 [=====] - 3s 3s/step
THE PREDICTED LETTER IS  E
```

```
In [6]: img = image.load_img('./Dataset/test_set/A/112.png')
        pred=detect(img)
```

```
1/1 [=====] - 0s 24ms/step
THE PREDICTED LETTER IS  A
```

```
In [7]: img = image.load_img('./Dataset/test_set/I/12.png')
        pred=detect(img)
```

```
1/1 [=====] - 0s 22ms/step
THE PREDICTED LETTER IS  I
```

```
In [8]: img = image.load_img('./Dataset/test_set/C/57.png')
        pred=detect(img)
```

```
1/1 [=====] - 0s 24ms/step
THE PREDICTED LETTER IS  C
```



## PHASE 4 : Setting Up Flask UI

We will now create a Flask application that will be used to create our user interface and which, on the backend, will interface with the model to obtain predictions. A Python file is needed for the backend, which manages the interaction with the model, and an HTML page is needed for the frontend of a Flask application.

```
125 lines (105 sloc) | 4.8 KB
Raw Blame

1  # import the necessary packages
2  from flask import Flask,render_template
3  # Flask-It is our framework which we are going to use to run/serve our application.
4  #request-for accessing file which was uploaded by the user on our application.
5  import cv2 # opencv library
6  from tensorflow.keras.models import load_model#to load our trained model
7  import numpy as np
8  import pyttsx3 #to convert text to speech
9  from skimage.transform import resize
10
11
12  app = Flask(__name__,template_folder="templates") # initializing a flask app
13  # Loading the model
14  model=load_model(r".\ASL_Model.h5")
15  print("Loaded model from disk")
16
17  background = None
18  accumulated_weight = 0.5
19  ROI_top = 100
20  ROI_bottom = 300
21  ROI_right = 150
22  ROI_left = 350
23
24  word_dict = { 0:'A', 1:'B', 2:'C', 3:'D', 4:'E', 5:'F', 6:'G',7:'H', 8:'I' }
25  vals = ['A', 'B','C','D','E','F','G','H','I']
26
27  @app.route('/', methods=['GET'])
28  def index():
29      return render_template('home.html')
30  @app.route('/home', methods=['GET'])
31  def home():
32      return render_template('home.html')
```

```

33 @app.route('/upload', methods=['GET', 'POST'])
34 def predict():
35
36
37     def cal_accum_avg(frame, accumulated_weight):
38
39         global background
40
41         if background is None:
42             background = frame.copy().astype("float")
43             return None
44
45         cv2.accumulateWeighted(frame, background, accumulated_weight)
46
47     def segment_hand(frame, threshold=25):
48         global background
49         diff = cv2.absdiff(background.astype("uint8"), frame)
50         _, thresholded = cv2.threshold(diff, threshold, 255, cv2.THRESH_BINARY)
51         # Fetching contours in the frame (These contours can be of hand or any other object in foreground) ...
52         contours, hierarchy = cv2.findContours(thresholded.copy(), cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)
53         # If length of contours list = 0, means we didn't get any contours...
54         if len(contours) == 0:
55             return None
56         else:
57             # The largest external contour should be the hand
58             hand_segment_max_cont = max(contours, key=cv2.contourArea)
59
60             # Returning the hand segment (max contour) and the thresholded image of hand...
61             return (thresholded, hand_segment_max_cont)
62

```

```

63 cam = cv2.VideoCapture(0)
64 num_frames = 0
65 while True:
66     ret, frame = cam.read()
67     # flipping the frame to prevent inverted image of captured frame...
68     frame = cv2.flip(frame, 1)
69     frame_copy = frame.copy()
70     # ROI from the frame
71     roi = frame[ROI_top:ROI_bottom, ROI_right:ROI_left]
72     gray_frame = cv2.cvtColor(roi, cv2.COLOR_BGR2GRAY)
73     gray_frame = cv2.GaussianBlur(gray_frame, (9, 9), 0)
74     if num_frames < 70:
75         cal_accum_avg(gray_frame, accumulated_weight)
76         cv2.putText(frame_copy, "FETCHING BACKGROUND...PLEASE WAIT", (80, 400),
77                     cv2.FONT_HERSHEY_SIMPLEX, 0.9, (0, 0, 255), 2)
78     else:
79         # segmenting the hand region
80         hand = segment_hand(gray_frame)
81         # Checking if we are able to detect the hand...
82         if hand is not None:
83             thresholded, hand_segment = hand
84             # Drawing contours around hand segment
85             cv2.drawContours(frame_copy, [hand_segment + (ROI_right, ROI_top)],
86                             -1, (255, 0, 0), 1)
87             cv2.imshow("Thresholded Hand Image", thresholded)
88             thresholded = cv2.resize(thresholded, (64, 64))
89             thresholded = cv2.cvtColor(thresholded, cv2.COLOR_GRAY2RGB)
90             thresholded = np.reshape(thresholded, (1, thresholded.shape[0], thresholded.shape[1], 3))
91             pred = model.predict(thresholded)
92             result = word_dict[np.argmax(pred)]
93             cv2.putText(frame_copy, result, (170, 45),
94                         cv2.FONT_HERSHEY_SIMPLEX, 1, (0, 0, 255), 2)
95             text_speech = pyttsx3.init()
96             rate = 100
97             text_speech.setProperty("rate", rate)
98             text_speech.say(result)
99             text_speech.runAndWait()
100

```

```

101     # Draw ROI on frame_copy
102     cv2.rectangle(frame_copy, (ROI_left, ROI_top),
103                   (ROI_right, ROI_bottom), (255, 128, 0), 3)
104     # incrementing the number of frames for tracking
105     num_frames += 1
106     # Display the frame with segmented hand
107     cv2.putText(frame_copy, "DataFlair hand sign recognition __",
108                 (10, 20), cv2.FONT_ITALIC, 0.5, (51, 255, 51), 1)
109     cv2.imshow("Sign Detection", frame_copy)
110     # Close windows with Esc
111     k = cv2.waitKey(1) & 0xFF
112     if k == 'q':
113         break
114
115     # Release the camera and destroy all the windows
116     cam.release()
117     cv2.destroyAllWindows()
118
119
120     return render_template("upload.html")
121
122
123 if __name__ == '__main__':
124     app.run(host="0.0.0.0", port=8000, debug=False)
125

```

## 8.TESTING

### 8.1 Test Cases

The following test cases must be taken into account when testing the model:

- Check to see that the options are visible when the user hits the URL.
- Check to see if the UI elements are properly shown.
- Check to see if the user has any language options.
- Check to see if the user is being forwarded to the page for sign language.
- Check to see if the software can translate sign language into voice.
- Check to see if the user can leave the page with sign language.
- Check to see if the user is being forwarded to the page for speech to sign.

- Check to see if the UI elements are visible.
- Check to see if the voice to text button on the application can convert speech to text.
- Check to make sure the user can leave the speech-to-sign page.

## 8.2 User Acceptance Testing

User acceptance testing (UAT), also known as application testing or end-user testing, is a stage of the software development process when the target user group tests the product in the real world. Before the tested software is made available to its intended market, UAT is frequently the final stage of the software design process.

### 1. Defect Analysis

This report shows the number of resolved or closed bugs at each severity level, and how they were resolved

Resolution	Severity 1	Severity 2	Severity 3	Severity 4	Subtotal
By Design	11	7	4	2	24
Duplicate	1	0	2	0	3
External	2	3	2	1	8
Fixed	10	5	3	14	32
Not Reproduced	0	0	1	0	1
Skipped	0	0	1	1	2
Won't Fix	1	0	0	0	1
Totals	25	15	13	18	71

## 2. Test Case Analysis

This report shows the number of test cases that have passed, failed, and untested

Section	Total Cases	Not Tested	Fail	Pass
Print Engine	7	0	0	7
Client Application	15	0	0	15
Security	2	0	0	2
Outsource Shipping	2	0	0	2
Exception Reporting	9	0	0	9
Final Report Output	4	0	0	4
Version Control	2	0	0	2

# 9.RESULT

## 9.1 Performance Metrics

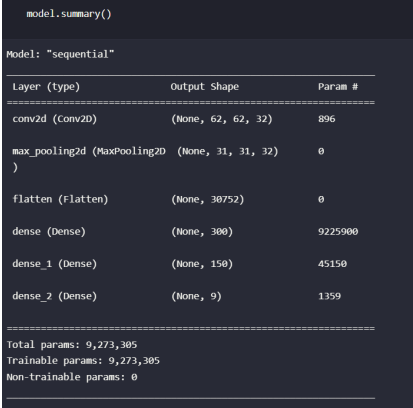
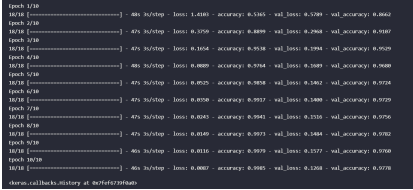
One of the crucial phases in creating a successful machine learning model is evaluating its performance. Different metrics—also referred to as performance metrics or evaluation metrics—are used to assess the effectiveness or quality of the model.

These performance indicators enable us to evaluate how well our model handled the supplied data. By adjusting the hyper-parameters, we can make the model perform better. Performance metrics assist measure how well a machine learning (ML) model generalises on new or previously unexplored data.

We must exercise caution while selecting the measures to measure the effectiveness of ML because:

- The statistic you select will determine exactly how the effectiveness of ML algorithms is assessed and compared.
- The metric you select will have a significant impact on how you weight the relative value of different variables in the outcome.

To comprehend the performance metrics of our system, the following characteristics can be examined:

S.No.	Parameter	Values	Screenshot
1.	Model Summary	Total params: 9,273,305  Trainable params: 9,273,30  Non Trainable params: 0  : No  0	 <pre> model.summary()  Model: "sequential" Layer (type)                Output Shape              Param # ----- conv2d (Conv2D)              (None, 62, 62, 32)       896 max_pooling2d (MaxPooling2D) (None, 31, 31, 32)       0 Flatten (Flatten)            (None, 30752)             0 dense (Dense)                 (None, 300)               9225900 dense_1 (Dense)              (None, 150)              45150 dense_2 (Dense)              (None, 9)                 1359 ----- Total params: 9,273,305 Trainable params: 9,273,305 Non-trainable params: 0           </pre>
2.	Accuracy	Training Accuracy - 0.9985  Validation Accuracy -0.9778	 <pre> Epoch 1/10: 40s/30s - loss: 1.4183 - accuracy: 0.5165 - val_loss: 0.5289 - val_accuracy: 0.9862 Epoch 2/10: 40s/30s - loss: 0.3758 - accuracy: 0.8889 - val_loss: 0.2984 - val_accuracy: 0.9587 Epoch 3/10: 40s/30s - loss: 0.3054 - accuracy: 0.9518 - val_loss: 0.1984 - val_accuracy: 0.9628 Epoch 4/10: 40s/30s - loss: 0.0889 - accuracy: 0.9794 - val_loss: 0.1889 - val_accuracy: 0.9688 Epoch 5/10: 40s/30s - loss: 0.0825 - accuracy: 0.9858 - val_loss: 0.1862 - val_accuracy: 0.9784 Epoch 6/10: 40s/30s - loss: 0.0788 - accuracy: 0.9917 - val_loss: 0.1888 - val_accuracy: 0.9779 Epoch 7/10: 40s/30s - loss: 0.0843 - accuracy: 0.9941 - val_loss: 0.1938 - val_accuracy: 0.9794 Epoch 8/10: 40s/30s - loss: 0.0849 - accuracy: 0.9971 - val_loss: 0.1884 - val_accuracy: 0.9782 Epoch 9/10: 40s/30s - loss: 0.0816 - accuracy: 0.9971 - val_loss: 0.1757 - val_accuracy: 0.9788 Epoch 10/10: 40s/30s - loss: 0.0807 - accuracy: 0.9985 - val_loss: 0.1708 - val_accuracy: 0.9778  keras.callbacks.History at 0x7f6c0c000000           </pre>
3.	Confidence Score (Only Yolo Projects)	Class Detected - NA  Confidence Score - NA	NA

## 10. ADVANTAGES AND DISADVANTAGES

### 10.1 ADVANTAGES

- The model is quite accurate.
- Conversion of sign to speech in real time.
- Convenient UI
- Data security

## **10.2 DISADVANTAGES**

- The current model is limited to the letters A through I.
- There is no option to customise sign language.
- The app does not allow note-taking while being used.
- The app does not allow users to make calls.

## **11.CONCLUSION**

The ability to express oneself requires communication. Additionally, it satisfies one's needs. Career advancement requires effective communication. Good communication abilities can improve your personal skills.

By promoting mutual understanding, you may make your life simpler and your interactions with others better. As part of our effort, a system that converts speech into suitable sign language for the deaf and dumb has been created. In order to converse with regular people, it also transforms sign language into a human hearing voice.

The suggested method converts sign language into human-understandable English speech. With the help of this technology, the model receives hand gestures, recognises them, and then shows the corresponding Alphabet on the screen. This initiative allows deaf-mute people to perform sign language with their hands, which will later be translated into alphabets.

## **12.FUTURE SCOPE**

For persons who are deaf or dumb, having technology that can convert



hand sign language to its matching alphabet is a game changer in the fields of communication and AI. The web programme can be readily developed to recognise letters other than "I," numerals, and other symbols, as well as control of software/hardware interfaces, with the inclusion of gesture recognition.

An ISL word and phrase recognition model can be constructed. For this, a system that can detect changes in a brief amount of time will be required. We can also develop a finished product that will help mute or deaf persons.

The functionalities that can be included in our application include the following:

- Using the current model as a framework, a communication app can be created.
- The appropriate mode (speech to sign or sign to speech) can be selected by the user.
- The model's precision can be further improved and refined in future iterations.
- The addition of language customization is planned.
- Another element that may be introduced to signage is localization.
- Implementation of a TalkBack functionality could improve user navigation within the app.

# 13.APPENDIX

## Source Code

### Model Building

#### Loading the Dataset & Image Data Generation

```
In [1]: from tensorflow.keras.preprocessing.image import ImageDataGenerator

In [2]: # Training Datasets
train_datagen = ImageDataGenerator(rescale=1/255, zoom_range=0.2, horizontal_flip=True, vertical_flip=False)
# Testing Datasets
test_datagen = ImageDataGenerator(rescale=1/255)

In [3]: # Training Dataset
x_train=train_datagen.flow_from_directory(r'.\Dataset\training_set', target_size=(64,64), class_mode='categorical', batch_size=900)
# Testing Dataset
x_test=test_datagen.flow_from_directory(r'.\Dataset\test_set', target_size=(64,64), class_mode='categorical', batch_size=900)

Found 15750 images belonging to 9 classes.
Found 2250 images belonging to 9 classes.

In [4]: print("Len x-train : ", len(x_train))
print("Len x-test : ", len(x_test))

Len x-train : 18
Len x-test : 3

In [5]: # The Class Indices in Training Dataset
x_train.class_indices

Out[5]: {'A': 0, 'B': 1, 'C': 2, 'D': 3, 'E': 4, 'F': 5, 'G': 6, 'H': 7, 'I': 8}
```

### Model Creation

```
In [6]: # Importing Libraries
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Convolution2D, MaxPooling2D,
from tensorflow.keras.layers import Flatten, Dense

In [7]: # Creating Model
model=Sequential()

In [8]: # Adding Layers
model.add(Convolution2D(32, (3,3), activation='relu', input_shape=(64,64,3)))
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Flatten())

# Adding Hidden Layers
model.add(Dense(300, activation='relu'))
model.add(Dense(150, activation='relu'))

# Adding Output Layer
model.add(Dense(9, activation='softmax'))

In [13]: model.summary()

Model: "sequential"

```

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 62, 62, 32)	896
max_pooling2d (MaxPooling2D)	(None, 31, 31, 32)	0
flatten (Flatten)	(None, 30752)	0
dense (Dense)	(None, 300)	9225900
dense_1 (Dense)	(None, 150)	45150
dense_2 (Dense)	(None, 9)	1359

```

Total params: 9,273,305
Trainable params: 9,273,305
Non-trainable params: 0
```

```
In [9]: # Compiling the Model
model.compile(loss='categorical_crossentropy',optimizer='adam',metrics=['accuracy'])
```

```
In [10]: # Fitting the Model Generator
model.fit_generator(x_train,steps_per_epoch=len(x_train),epochs=10,validation_data=x_test,validation_steps=len(x_test))
```

C:\Users\snavel\AppData\Local\Temp\ipykernel\_3100\1042518445.py:2: UserWarning: `Model.fit\_generator` is deprecated and will be removed in a future version. Please use `Model.fit`, which supports generators.

```
model.fit_generator(x_train,steps_per_epoch=len(x_train),epochs=10,validation_data=x_test,validation_steps=len(x_test))
Epoch 1/10
18/18 [=====] - 29s 1s/step - loss: 0.9233 - accuracy: 0.6873 - val_loss: 0.3513 - val_accuracy: 0.8898
Epoch 2/10
18/18 [=====] - 17s 914ms/step - loss: 0.1772 - accuracy: 0.9458 - val_loss: 0.2505 - val_accuracy: 0.9542
Epoch 3/10
18/18 [=====] - 17s 923ms/step - loss: 0.0780 - accuracy: 0.9775 - val_loss: 0.1611 - val_accuracy: 0.9702
Epoch 4/10
18/18 [=====] - 17s 929ms/step - loss: 0.0404 - accuracy: 0.9890 - val_loss: 0.1985 - val_accuracy: 0.9760
Epoch 5/10
18/18 [=====] - 17s 930ms/step - loss: 0.0249 - accuracy: 0.9944 - val_loss: 0.2219 - val_accuracy: 0.9764
Epoch 6/10
18/18 [=====] - 17s 909ms/step - loss: 0.0177 - accuracy: 0.9961 - val_loss: 0.2412 - val_accuracy: 0.9773
Epoch 7/10
18/18 [=====] - 17s 920ms/step - loss: 0.0109 - accuracy: 0.9979 - val_loss: 0.2459 - val_accuracy: 0.9773
Epoch 8/10
18/18 [=====] - 17s 912ms/step - loss: 0.0100 - accuracy: 0.9975 - val_loss: 0.1562 - val_accuracy: 0.9782
Epoch 9/10
18/18 [=====] - 16s 907ms/step - loss: 0.0058 - accuracy: 0.9992 - val_loss: 0.2264 - val_accuracy: 0.9782
Epoch 10/10
18/18 [=====] - 16s 913ms/step - loss: 0.0041 - accuracy: 0.9992 - val_loss: 0.2430 - val_accuracy: 0.9782
```

Out[10]:

## Saving the Model

```
In [16]: model.save('ASL_Model.h5')
```

125 lines (105 sloc) | 4.8 KB

Raw

Blame



```
1 # import the necessary packages
2 from flask import Flask,render_template
3 # Flask-It is our framework which we are going to use to run/serve our application.
4 #request-for accessing file which was uploaded by the user on our application.
5 import cv2 # opencv library
6 from tensorflow.keras.models import load_model#to load our trained model
7 import numpy as np
8 import pyttsx3 #to convert text to speech
9 from skimage.transform import resize
10
11
12 app = Flask(__name__,template_folder="templates") # initializing a flask app
13 # Loading the model
14 model=load_model(r"..\ASL_Model.h5")
15 print("Loaded model from disk")
16
17 background = None
18 accumulated_weight = 0.5
19 ROI_top = 100
20 ROI_bottom = 300
21 ROI_right = 150
22 ROI_left = 350
23
24 word_dict = { 0:'A', 1:'B', 2:'C', 3:'D', 4:'E', 5:'F', 6:'G',7:'H', 8:'I' }
25 vals = ['A', 'B','C','D','E','F','G','H','I']
26
27 @app.route('/', methods=['GET'])
28 def index():
29     return render_template('home.html')
30 @app.route('/home', methods=['GET'])
31 def home():
32     return render_template('home.html')
```

```

33 @app.route('/upload', methods=['GET', 'POST'])
34 def predict():
35
36
37     def cal_accum_avg(frame, accumulated_weight):
38
39         global background
40
41         if background is None:
42             background = frame.copy().astype("float")
43             return None
44
45         cv2.accumulateWeighted(frame, background, accumulated_weight)
46
47     def segment_hand(frame, threshold=25):
48         global background
49         diff = cv2.absdiff(background.astype("uint8"), frame)
50         _, thresholded = cv2.threshold(diff, threshold, 255, cv2.THRESH_BINARY)
51         # Fetching contours in the frame (These contours can be of hand or any other object in foreground) ...
52         contours, hierarchy = cv2.findContours(thresholded.copy(), cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)
53         # If length of contours list = 0, means we didn't get any contours...
54         if len(contours) == 0:
55             return None
56         else:
57             # The largest external contour should be the hand
58             hand_segment_max_cont = max(contours, key=cv2.contourArea)
59
60             # Returning the hand segment(max contour) and the thresholded image of hand...
61             return (thresholded, hand_segment_max_cont)
62

```

```

63 cam = cv2.VideoCapture(0)
64 num_frames = 0
65 while True:
66     ret, frame = cam.read()
67     # flipping the frame to prevent inverted image of captured frame...
68     frame = cv2.flip(frame, 1)
69     frame_copy = frame.copy()
70     # ROI from the frame
71     roi = frame[ROI_top:ROI_bottom, ROI_right:ROI_left]
72     gray_frame = cv2.cvtColor(roi, cv2.COLOR_BGR2GRAY)
73     gray_frame = cv2.GaussianBlur(gray_frame, (9, 9), 0)
74     if num_frames < 70:
75         cal_accum_avg(gray_frame, accumulated_weight)
76         cv2.putText(frame_copy, "FETCHING BACKGROUND...PLEASE WAIT", (80, 400),
77                     cv2.FONT_HERSHEY_SIMPLEX, 0.9, (0, 0, 255), 2)
78     else:
79         # segmenting the hand region
80         hand = segment_hand(gray_frame)
81         # Checking if we are able to detect the hand...
82         if hand is not None:
83             thresholded, hand_segment = hand
84             # Drawing contours around hand segment
85             cv2.drawContours(frame_copy, [hand_segment + (ROI_right, ROI_top)],
86                             -1, (255, 0, 0), 1)
87             cv2.imshow("Thresholded Hand Image", thresholded)
88             thresholded = cv2.resize(thresholded, (64, 64))
89             thresholded = cv2.cvtColor(thresholded, cv2.COLOR_GRAY2RGB)
90             thresholded = np.reshape(thresholded, (1, thresholded.shape[0], thresholded.shape[1], 3))
91             pred = model.predict(thresholded)
92             result = word_dict[np.argmax(pred)]
93             cv2.putText(frame_copy, result, (170, 45),
94                         cv2.FONT_HERSHEY_SIMPLEX, 1, (0, 0, 255), 2)
95             text_speech = pyttsx3.init()
96             rate = 100
97             text_speech.setProperty("rate", rate)
98             text_speech.say(result)
99             text_speech.runAndWait()
100

```

```

101     # Draw ROI on frame_copy
102     cv2.rectangle(frame_copy, (ROI_left, ROI_top),
103                   (ROI_right, ROI_bottom), (255, 128, 0), 3)
104     # incrementing the number of frames for tracking
105     num_frames += 1
106     # Display the frame with segmented hand
107     cv2.putText(frame_copy, "DataFlair hand sign recognition_ _ _",
108                 (10, 20), cv2.FONT_ITALIC, 0.5, (51, 255, 51), 1)
109     cv2.imshow("Sign Detection", frame_copy)
110     # Close windows with Esc
111     k = cv2.waitKey(1) & 0xFF
112     if k == 'q':
113         break
114
115     # Release the camera and destroy all the windows
116     cam.release()
117     cv2.destroyAllWindows()
118
119
120     return render_template("upload.html")
121
122
123 if __name__ == '__main__':
124     app.run(host='0.0.0.0', port=8000, debug=False)
125

```

## Github Link

<https://github.com/IBM-EPBL/IBM-Project-27507-1660058769>