REAL-TIME COMMUNICATION SYSTEM POWERED BY AI

FOR SPECIALLY ABLED

Bachelor of Engineering

In

Computer Science and Engineering

Submitted by

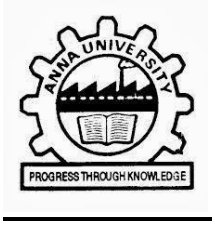
TEAM ID: PNT2022TMID48819

K.PON MUTHU ARAVIND (921019104037)

S.JAWAHAR HARIHARAN (921019104021)

S.KARUPPASAMY (921019104023)

K.ANAND (921019104004)



NADAR SARASWATHI COLLEGE OF ENGINEERING AND TECHNOLOGY, THENI-625531

ANNA UNIVERSITY

CHENNAI

Project Report Index

1.INTRODUCTION:

1.1 Project Overview:

Communication plays a significant role in making the world a better place. Communication creates bonding and relations among the people, whether persona, social, or political views. Most people communicate efficiently without any issues, but many cannot due to disability. They cannot hear or speak, which makes Earth a problematic place to live for them. Even simple basic tasks become difficult for them. Disability is an emotive human condition. It limits the individual to a certain level of performance. Being deaf and dumb pushes the subject to oblivion, highly introverted. In a world of inequality, this society needs empowerment. Harnessing technology to improve their welfare is necessary. In a tech era, no one should be limited due to his or her inability. The application of technology should create a platform or a world of equality despite the natural state of humans. On the other hand, technology is the most innovative thing on Earth for every time the clock ticks, researchers, software engineers, programmers, and information technology specialists are always coming up with bright ideas to provide convenience to everyone. This paper shows how artificial intelligence is being used to help people who are unable to do what most people do in their everyday lives. Aligned with communication, D-talk is a system that allows people who are unable to talk and hear be fully understood and for them to learn their language easier and also for the people that would interact and communicate with them. This system provides detailed hand gestures that show the interpretation at the bottom so that everyone can understand them. This research allows the readers to learn the system and what it can do to people who are struggling with what they are not capable of and will provide the technical terms on how the system works

1.2 Purpose:

In our society, we have people with disabilities. The technology is developing day by day but no significant developments are undertaken for the betterment of these people. Communications between deaf-mute and a normal person has always been a challenging task. It is very difficult for mute people to convey their message to normal people. Since normal people are not trained on hand sign language. In emergency times conveying their message is very difficult. The human hand has remained a popular choice to convey information in situations where other forms like speech cannot be used. Voice Conversion System with Hand Gesture Recognition and translation will be very useful to have a proper conversation between a normal person and an impaired person in any language. The project aims to develop a system that converts the sign language into a human hearing voice in the desired language to convey a message to normal people, as well as convert speech into understandable sign language for the deaf and dumb. We are making use of a convolution neural network to create a model that is trained on different hand gestures. An app is built which uses this model. This app enables deaf and dumb people to convey their information using signs which get converted to human-understandable language and speech is given as output.

2. LITERATURE SURVEY:

2.1 Existing problem :

One of the most precious gifts of nature to the human race is the ability to express itself by responding to the events that occur in its environment. Every normal person sees, hears, and then reacts to the situations by expressing himself. But there are some less lucky ones who are deprived of this precious gift. Such people, especially deaf and mute, rely on some sort of gesture language to communicate their feelings to others. The deaf, dumb and the blind follow similar problems when it comes to the use of computers. In the era of advanced technologies, where computers, laptops and other processor-based devices are an integral part of everyday life, efforts must be made to make the disabilities in life more independent. Our goal is to design a human computer interface system that can accurately identify the language of the deaf and dumb. With the use of image processing and artificial intelligence, many techniques and algorithms have been developed in this area. Each character speech recognition system is trained to recognize the characters and convert them into the required pattern. The proposed system aims to give speech speechless, a real-time character language is captured as a series of images, and it is processed and then converted into speech and text

2.2 References

1) **A Signer Independent Sign Language Recognition with Coarticulation Elimination from Live Videos: an Indian Scenario** P.K. Athira, C.J. Sruthi, A. Lijiya (2019)
   Advantage: Economical can be implemented with a mobile camera which makes it very user-friendly
   Disadvantage: Not efficient under cluttered backgrounds and different illumination conditions
2) **A Deep Learning based Indian Sign Language Recognition System** Sruthi C. J and Lijiya A (2019)
   Advantage: Training accuracy of 99.93% and with testing and validation accuracy of 98.64%.
   Disadvantage: Facial expression and context analysis are the other part not included
3) **Hand Gesture Recognition for Sign Language Using 3DCNN** Muneer AlHammadi, Ghulam Muhammad, Wadood Abdul, Mansour Alsulaiman, Mohamed A. Bencherif, And Mohamed Amine Mekhtiche (2020)
   Advantage: The proposed approaches were compared with six other state-of-the art methods from the literature. They outperformed four of these methods and showed comparable performance to the other two.
   Disadvantage: Does not work for a live video feed.

2.3 Problem Statement Definition

Communication is the only medium by which we can share our thoughts or convey the message but communications between deafmute and a normal person has always been a challenging task. It is very difficult for mute people to convey their message to normal people. Since normal people are not trained on hand sign language. In emergency times conveying their message is very difficult.

Problem:

The boy has difficulty in hearing. He uses sign language to communicate with others. But he can't able to communicate with normal people who don't understand sign language.

Solution:

To develop a system that converts the sign language into a human hearing voice in the desired language to convey a message to normal people, as well as convert speech into understandable sign language for the deaf, the system enhances the user friendly experience.

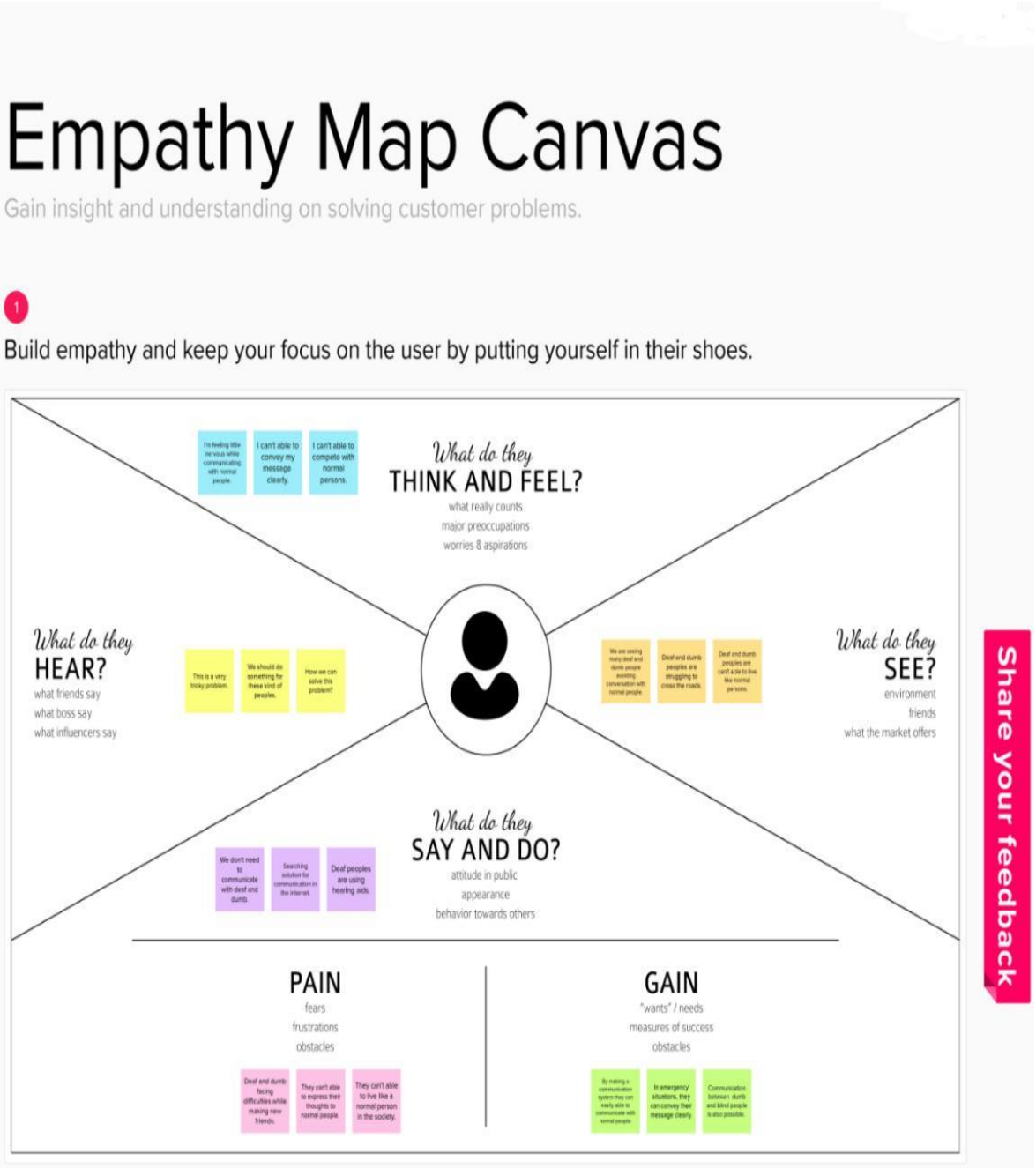
Problem:

Karupan is a dumb by birth. He uses sign language to communicate with others. But he can't able to communicate with normal people who don't understand sign language.

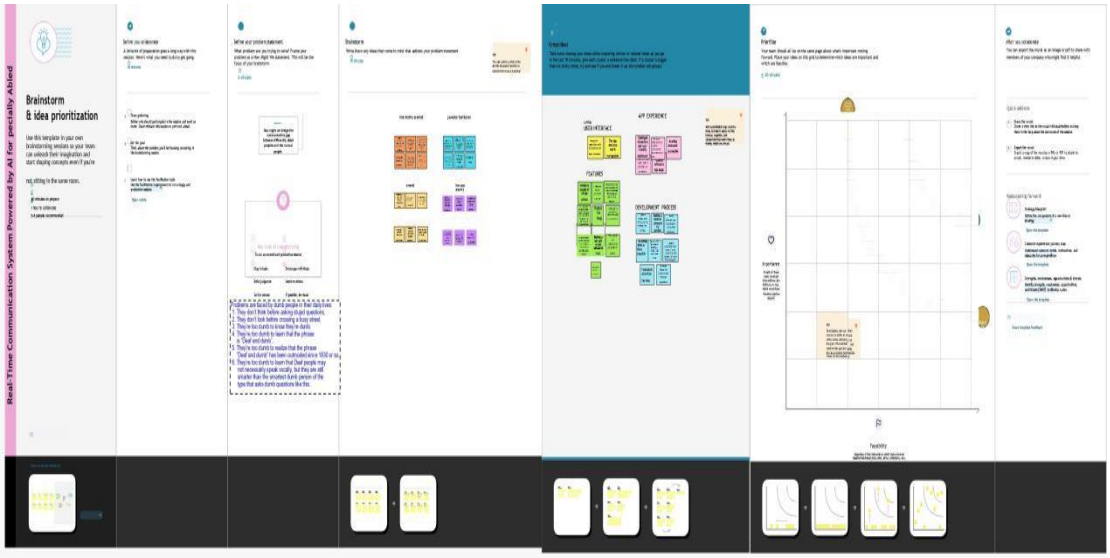
Solution:
To create a app for understanding sign language and convert into
Speech signal as output for normal people

3. IDEATION & PROPOSED SOLUTION

3.1 Empathy Map Canvas:



3.2 Ideation & Brainstorming:



3.3 Proposed Solution

Project team shall fill the following information in proposed solution template.

S.No.	Parameter	Description
1.	Problem Statement (Problem to besolved)	• Everyone is not convenient with language used in the application • Some people cannot understand English we can convert into their convenient language • They are facing difficulties in understanding the language used inthe system
2.	Idea / Solution description	• Even sign language can also be translated to text message in ourapplication using CNN. • Text to sign language convertor usesStanford Parser text processing and JA Signing for the signing avatar • Can change the language using google language translator tool so that people can use the applicationbased on their specialized language • Producing a model which can recognize Finger-spelling based handgestures in order to form a completeword by combining each gesture • By using this application both specially abled and normal people cantranslate their messages to others easily

3.	Novelty / Uniqueness	• Convenient language can be changedusing the google language translator tool • Normal text can also be translated into sign language
4.	Social Impact / CustomerSatisfaction	• The Main aim of the project is to buildan application that helps the especially abled people to communicate with others easily • The deaf and dumb people can easilytranslate their sign languageinto a human hearing voice • The normal people can also easilytranslate their voice into a sign language using this application
5.	Business Model (Revenue Model)	• We can generate revenue by offering subscription-based applications to thepeople • Users who have got subscription canchange the language accordingly
6.	Scalability of the Solution	• Even if the number of users increasethe system will perform well • Need to pay attention to the application and to be responsive tothe changes as fast as possible

3.4 Problem Solution fit:

4. REQUIREMENT ANALYSIS :

4.1 Functional Requirements:

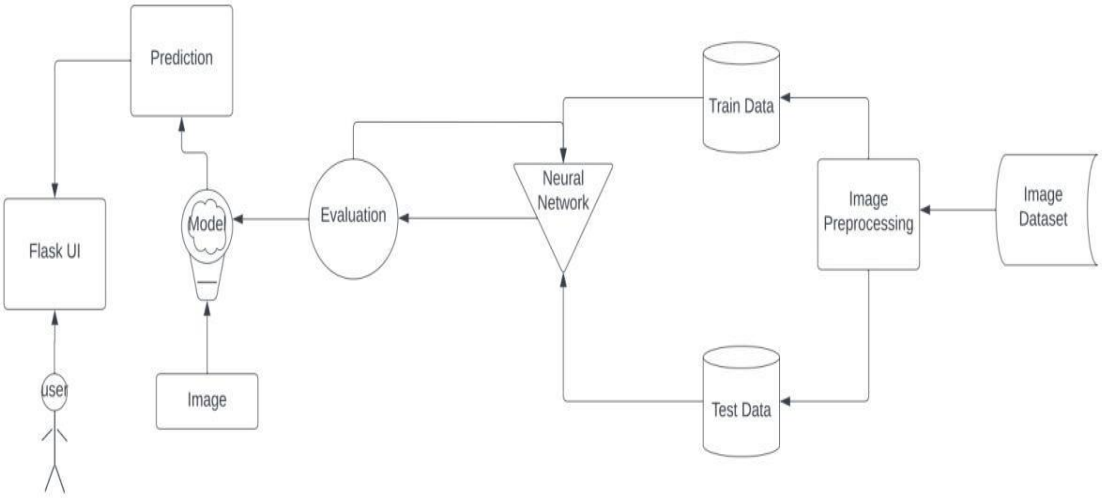
FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration	• Registration through Web UI/ E-Mail ID. • Authentication via OTP.
FR-2	User Confirmation	• Confirmation via mail.
FR-3	System	• Desktop/ Mobile with good resolutioncamera. • Provides system access to capture images/ video and other relevant data.
FR-4	Text conversion	Converts the Sign language into a text using Convolutional Neural Network (CNN) Model.
FR-5	Sentence Translation	To create sentence(s) by recognizing the signs and pauses in the input video stream.

4.2 Non-Functional Requirements:

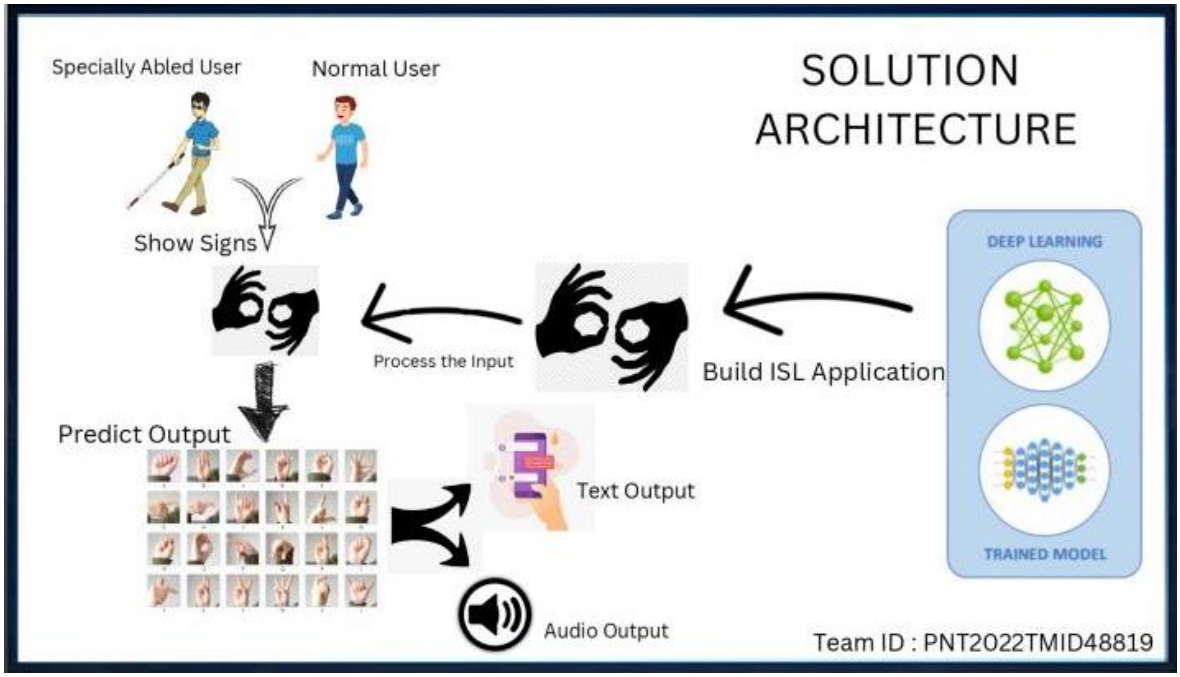
NFR No.	Non-Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
NFR-1	Usability	Deaf-mute people should be able to use the system with ease. The same applies for normal people who get the system's output. The system should have good UI.
NFR-2	Security	Even though the use-case of the system doesn't need any security feature, it must be ensured that the privacy of user data be maintained and handled appropriately.
NFR-3	Reliability	The translation of sign languages should be reliable.The accuracy of the system should be tested extensively to make sure that it is up to the mark.
NFR-4	Performance	The processing should be done in considerable timeso that the conversation can go on without waiting for the system's output.
NFR-5	Availability	The system should be universally accessible. Sincesign language is almost same everywhere, the system can be used across the globe.
NFR-6	Scalability	The system should be scalable to accommodate new features and functionalities and to cater wider range of people in future.

5. PROJECT DESIGN:

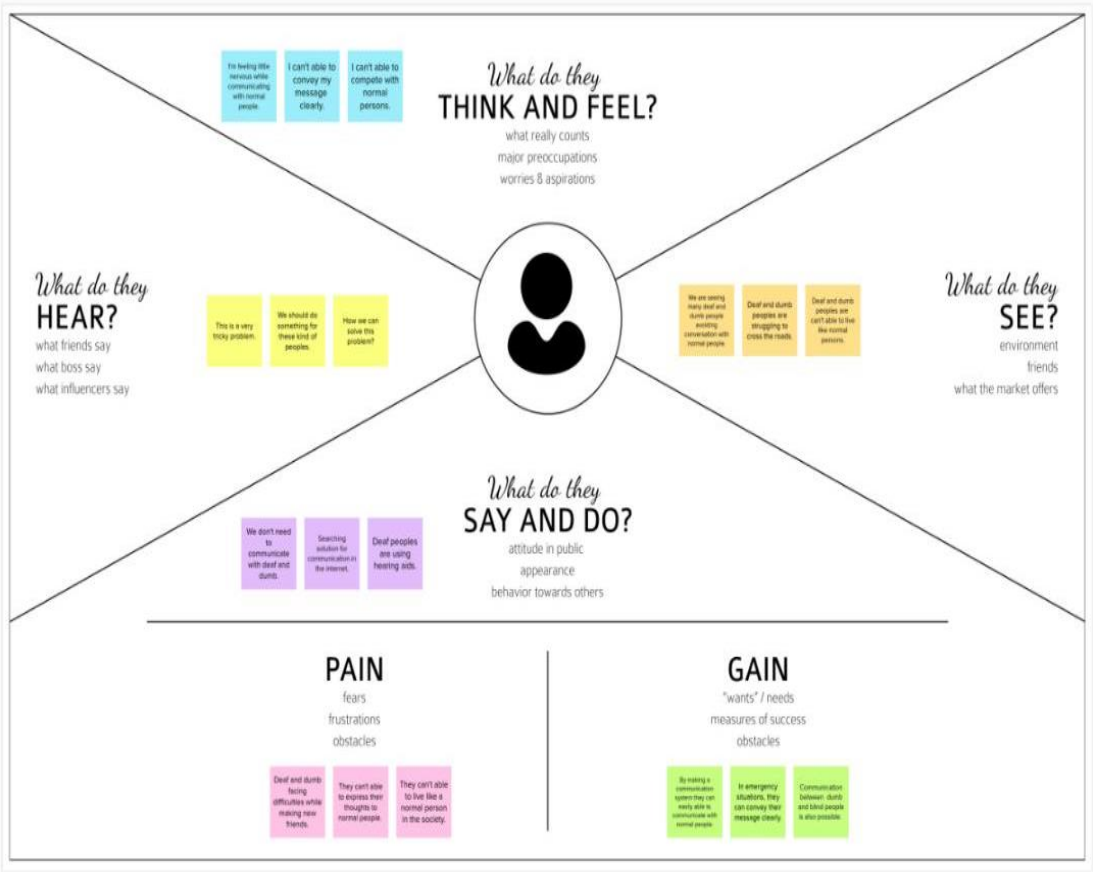
5.1 Data Flow Diagram:



5.2Solution & Technical Architecture:



5.3 User Stories:



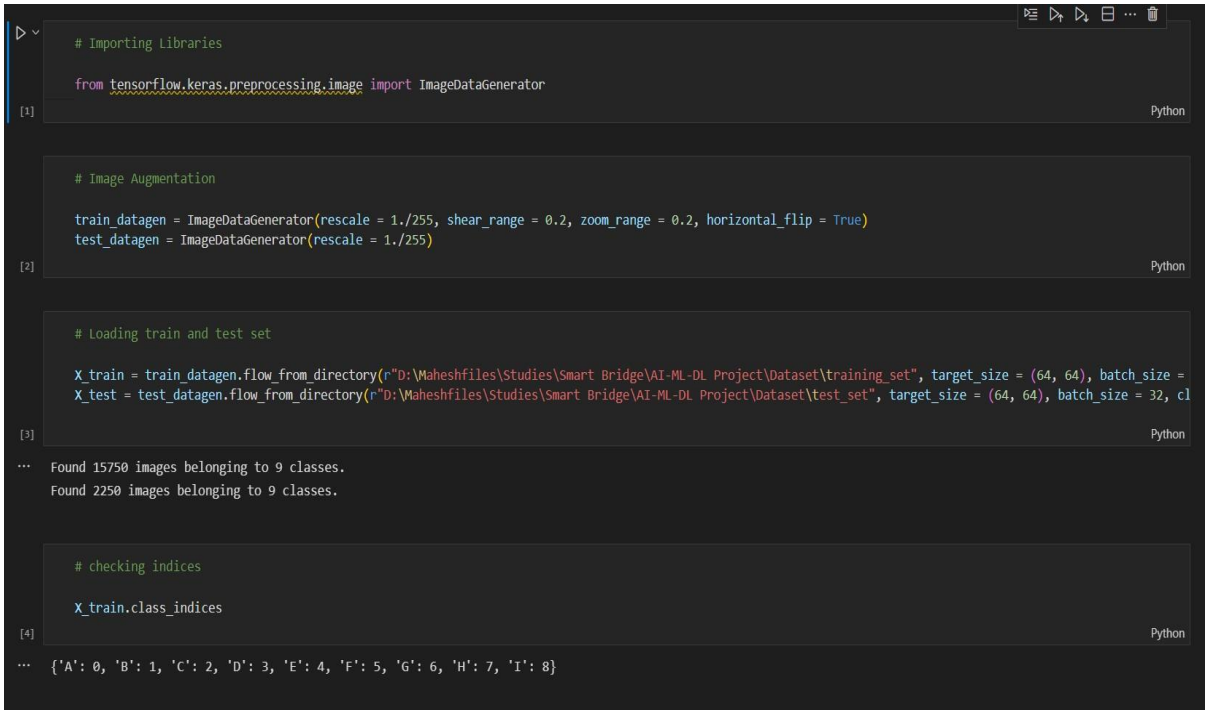
6. PROJECT PLANNING & SCHEDULING

6.1 Sprint Planning & Scheduling:

Sprint	Functional Requirement (Epic)	User Story Number	Story Points	Priority	Team Members
Sprint-1	Data Collection	USN-1	5	High	Pon muthu aravind
Sprint-1	Model Building	USN-2	5	High	Pon muthu aravind
Sprint-2	training the Model Building	USN-3	5	High	Anand
Sprint-2	Testing the Model Building	USN-4	5	Medium	Anand
Sprint-3	Flask, html page	USN-5	5	High	Jawahar Hariharan
Sprint-4	Speech feature implementation	USN-6	5	Medium	karupaasamy

7. CODING & SOLUTIONING

Training the train dataset:



```
# Importing Libraries

from tensorflow.keras.preprocessing.image import ImageDataGenerator
```

[1] Python

```
# Image Augmentation

train_datagen = ImageDataGenerator(rescale = 1./255, shear_range = 0.2, zoom_range = 0.2, horizontal_flip = True)
test_datagen = ImageDataGenerator(rescale = 1./255)
```

[2] Python

```
# Loading train and test set

X_train = train_datagen.flow_from_directory(r"D:\Maheshfiles\Studies\Smart Bridge\AI-ML-DL Project\Dataset\training_set", target_size = (64, 64), batch_size =
X_test = test_datagen.flow_from_directory(r"D:\Maheshfiles\Studies\Smart Bridge\AI-ML-DL Project\Dataset\test_set", target_size = (64, 64), batch_size = 32, cl
```

[3] Python

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

```
# checking indices

X_train.class_indices
```

[4] Python

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

Model Building

```
# Importing Libraries

from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense
from tensorflow.keras.layers import Convolution2D, MaxPooling2D, Flatten
```

[5] Python

```
# Initializing the Model

model = Sequential()
```

[6] Python

```
# Adding Convolution Layer

model.add(Convolution2D((32), (3,3), input_shape = (64, 64, 3), activation = 'relu'))
```

[7] Python

```
# Adding Pooling Layer

model.add(MaxPooling2D(pool_size = (2, 2)))
```

[8] Python

```python
# Adding Flatten Layer

model.add(Flatten())
```

[9] Python

```python
# Adding Hidden Layer

model.add(Dense(units = 512, kernel_initializer = 'random_uniform', activation = 'relu'))
```

[10] Python

```python
# Adding Output Layer

model.add(Dense(units = 9, kernel_initializer = 'random_uniform', activation = 'softmax'))
```

[11] Python

```python
# Compile the model

model.compile(loss = 'categorical_crossentropy', optimizer = 'adam', metrics = ['accuracy'])
```

[12] Python

```python
# Fiiting the model

model.fit_generator(X_train, steps_per_epoch = 24, epochs = 10, validation_data = X_test, validation_steps = 40)
```

] Python

```
C:\Users\mahes\AppData\Local\Temp/ipykernel_10216/1270027362.py:3: 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 = 24, epochs = 10, validation_data = X_test, validation_steps = 40)

Epoch 1/10
24/24 [==============================] - 26s 1s/step - loss: 1.4863 - accuracy: 0.5625 - val_loss: 0.6678 - val_accuracy: 0.7930
Epoch 2/10
24/24 [==============================] - 21s 878ms/step - loss: 0.5226 - accuracy: 0.8385 - val_loss: 0.3198 - val_accuracy: 0.9336
Epoch 3/10
24/24 [==============================] - 18s 759ms/step - loss: 0.3561 - accuracy: 0.8854 - val_loss: 0.3711 - val_accuracy: 0.9328
Epoch 4/10
24/24 [==============================] - 17s 711ms/step - loss: 0.2102 - accuracy: 0.9362 - val_loss: 0.2478 - val_accuracy: 0.9492
Epoch 5/10
24/24 [==============================] - 15s 638ms/step - loss: 0.1726 - accuracy: 0.9570 - val_loss: 0.2474 - val_accuracy: 0.9469
Epoch 6/10
24/24 [==============================] - 16s 648ms/step - loss: 0.1651 - accuracy: 0.9505 - val_loss: 0.2897 - val_accuracy: 0.9617
Epoch 7/10
24/24 [==============================] - 13s 560ms/step - loss: 0.1277 - accuracy: 0.9609 - val_loss: 0.2441 - val_accuracy: 0.9586
Epoch 8/10
24/24 [==============================] - 13s 543ms/step - loss: 0.0985 - accuracy: 0.9714 - val_loss: 0.2331 - val_accuracy: 0.9539
Epoch 9/10
24/24 [==============================] - 13s 528ms/step - loss: 0.0995 - accuracy: 0.9701 - val_loss: 0.2301 - val_accuracy: 0.9609
Epoch 10/10
24/24 [==============================] - 12s 503ms/step - loss: 0.0913 - accuracy: 0.9779 - val_loss: 0.2053 - val_accuracy: 0.9742

<keras.callbacks.History at 0x1d9801fe9d0>
```

```python
# Saving the model

model.save('aslpng1.h5')
```

[14] Python

8. TESTING

Testing the train dataset

```
# Importing Libarries

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

```
# loading model

model = load_model('aslpng1.h5')
```
[2] Python

```
from skimage.transform import resize
def detect(frame):
    img = resize(frame, (64, 64, 3))
    img = np.expand_dims(img, axis = 0)
    if np.max(img) > 1:
        img = img/255.0
    prediction = model.predict(img)
    print(prediction)
    return prediction
```
[3] Python

```
frame = cv2.imread(r"D:\Maheshfiles\Studies\Smart Bridge\AI-ML-DL Project\Dataset\training_set\D\16.png")
data = detect(frame)
```
[4] Python

```
1/1 [==============================] - 0s 266ms/step
[[3.9748478e-08 1.2755189e-05 1.0463478e-08 9.9853325e-01 2.6569789e-06
  1.3680419e-05 4.5120544e-08 1.8048374e-07 1.4373119e-03]]
```

```
index = ['A','B','C','D','E','F','G','H','I']
index[np.argmax(data)]
```
[5] Python

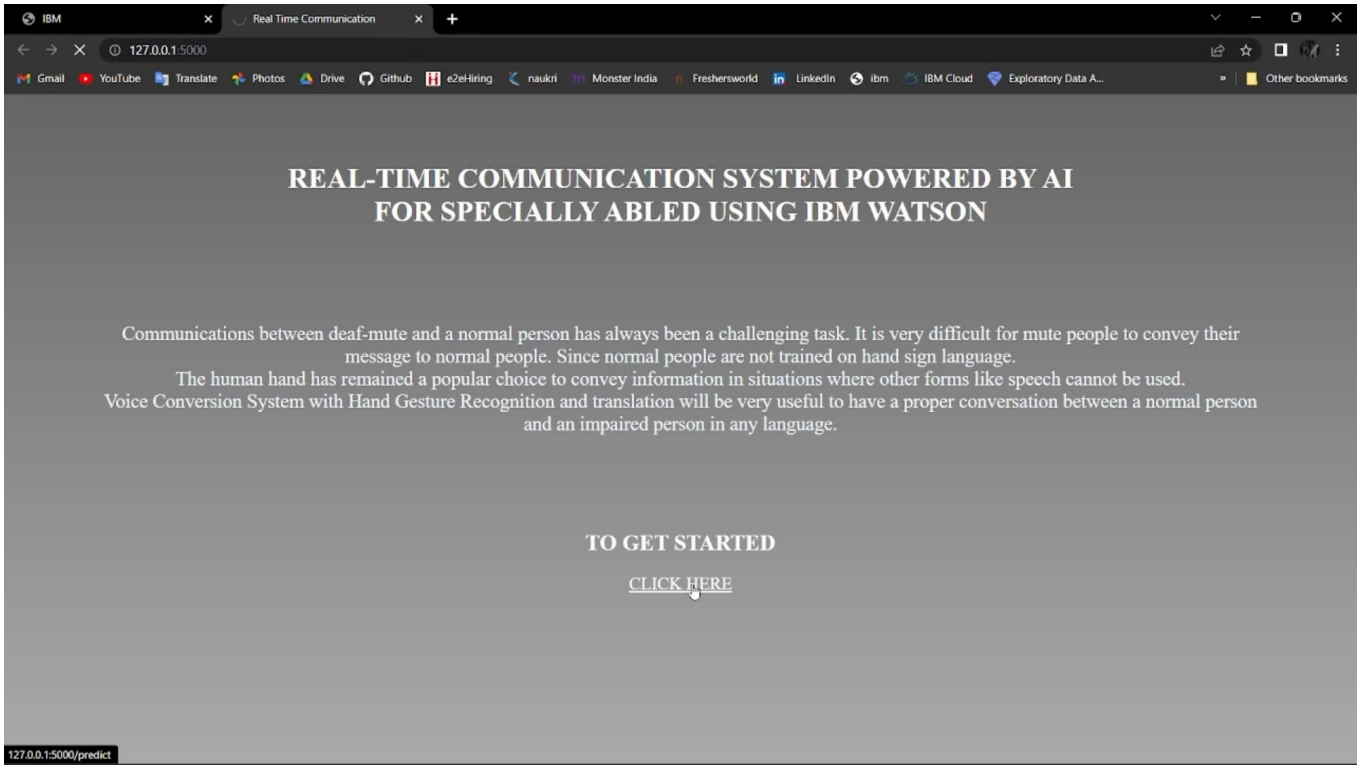
```
'D'
```

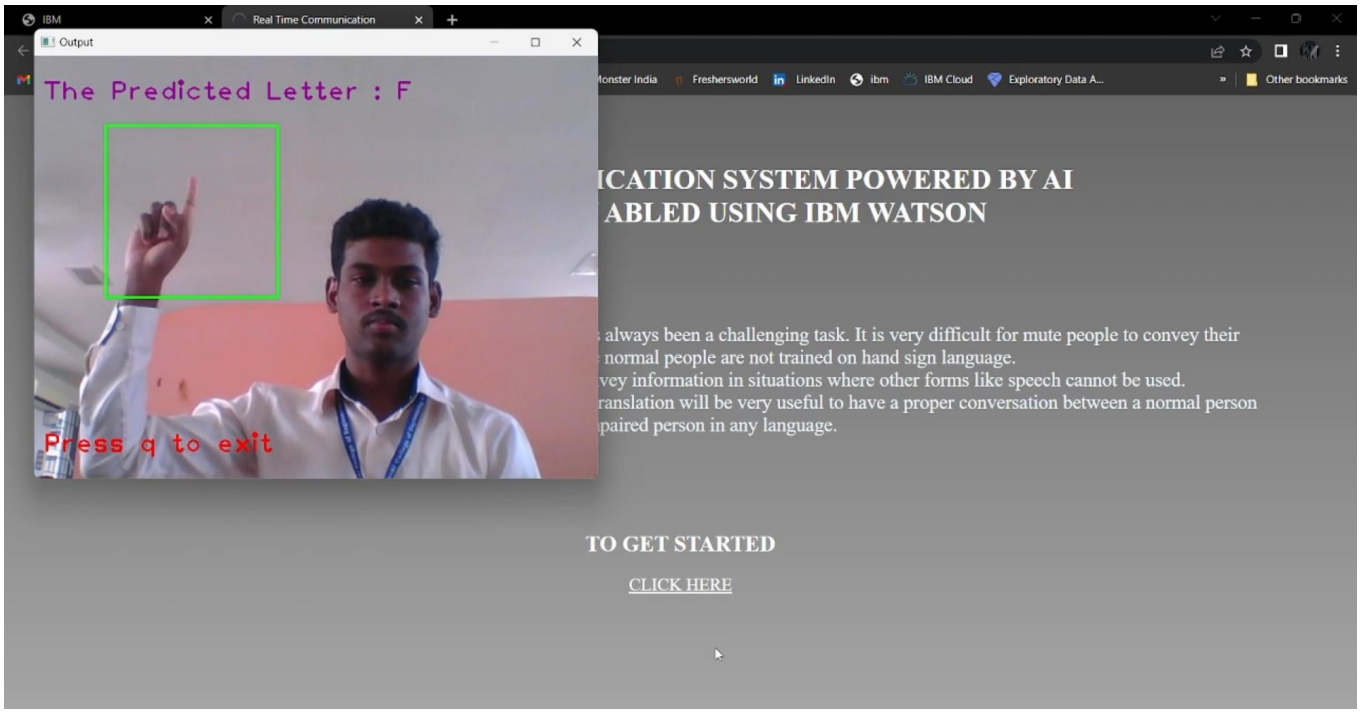
OpenCV

```
# Importing Libraries

import cv2
import numpy as np
from tensorflow.keras.models import load_model
from tensorflow.keras.preprocessing import image
```
[6] Python

9. RESULTS:


IBM
Real Time Communication
127.0.0.1:5000
Gmail
YouTube
Translate
Photos
Drive
Github
e2eHiring
naukri
Monster India
Freshersworld
LinkedIn
ibm
IBM Cloud
Exploratory Data A...
Other bookmarks
REAL-TIME COMMUNICATION SYSTEM POWERED BY AI
FOR SPECIALLY ABLED USING IBM WATSON
Communications between deaf-mute and a normal person has always been a challenging task. It is very difficult for mute people to convey their message to normal people. Since normal people are not trained on hand sign language.
The human hand has remained a popular choice to convey information in situations where other forms like speech cannot be used.
Voice Conversion System with Hand Gesture Recognition and translation will be very useful to have a proper conversation between a normal person and an impaired person in any language.
TO GET STARTED
CLICK HERE
127.0.0.1:5000/predict


IBM
Real Time Communication
Output
The Predicted Letter : F
Press q to exit
ICATION SYSTEM POWERED BY AI
ABLED USING IBM WATSON
always been a challenging task. It is very difficult for mute people to convey their
normal people are not trained on hand sign language.
vey information in situations where other forms like speech cannot be used.
ranslation will be very useful to have a proper conversation between a normal person
paired person in any language.
TO GET STARTED
CLICK HERE

10. ADVANTAGES & DISADVANTAGES

Advantages:

1. It is possible to create a mobile application to bridge the communication gap betweendeaf and dumb persons and the general public.
2. As different sign language standards exist, their dataset can be added, and the user canchoose which sign language to read.

Disadvantage:

1. The current model only works from alphabets A to I.
2. In absence of gesture recognition, alphabets from J cannot be identified as they requiresome kind of gesture input from the user.
3. As the quantity/quality of images in the dataset is low, the accuracy is not great, but thatcan easily be improved by change in dataset

11. CONCLUSION:

Sign language is a useful tool for facilitating communication between deaf and hearing people. Because it allows for two-way communication, the system aims to bridge the communication gap between deaf people and rest of society. The proposed methodology translates language into English alphabets that are understandable to humans. This system sends hand gestures to the model, who recognizes them and displays the equip agent Alphabet on the screen. Deaf-mute people can use their hands to perform signlanguage, which will then be converted into alphabets.

12. FUTURE SCOPE

Having a technology that can translate hand sign language to its corresponding alphabet is agame changer in the field of communication and AI for the specially abled people such as deaf and dumb. With introduction of gesture recognition, the web app can easily be expanded to recognize letters beyond ‘I’, digits and other symbols plus gesture recognition can also allow controlling of software/hardware interfaces.

13.APPENDIX

13.1 Source Code:

```python
# Importing Libraries

from tensorflow.keras.preprocessing.image import ImageDataGenerator
```

[1] Python

```python
# Image Augmentation

train_datagen = ImageDataGenerator(rescale = 1./255, shear_range = 0.2, zoom_range = 0.2, horizontal_flip = True)
test_datagen = ImageDataGenerator(rescale = 1./255)
```

[2] Python

```python
# Loading train and test set

X_train = train_datagen.flow_from_directory(r"D:\Maheshfiles\Studies\Smart Bridge\AI-ML-DL Project\Dataset\training_set", target_size = (64, 64), batch_size =
X_test = test_datagen.flow_from_directory(r"D:\Maheshfiles\Studies\Smart Bridge\AI-ML-DL Project\Dataset\test_set", target_size = (64, 64), batch_size = 32, cl
```

[3] Python

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

```python
# checking indices

X_train.class_indices
```

[4] Python

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

Model Building

```python
# Importing Libraries

from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense
from tensorflow.keras.layers import Convolution2D, MaxPooling2D, Flatten
```

[5] Python

```python
# Initializing the Model

model = Sequential()
```

[6] Python

```python
# Adding Convolution Layer

model.add(Convolution2D((32), (3,3), input_shape = (64, 64, 3), activation = 'relu'))
```

[7] Python

```python
# Adding Pooling Layer

model.add(MaxPooling2D(pool_size = (2, 2)))
```

[8] Python

```python
# Adding Flatten Layer

model.add(Flatten())
```

[9] Python

```python
# Adding Hidden Layer

model.add(Dense(units = 512, kernel_initializer = 'random_uniform', activation = 'relu'))
```

[10] Python

```python
# Adding Output Layer

model.add(Dense(units = 9, kernel_initializer = 'random_uniform', activation = 'softmax'))
```

[11] Python

```python
# Compile the model

model.compile(loss = 'categorical_crossentropy', optimizer = 'adam', metrics = ['accuracy'])
```

[12] Python

```python
# Fiiting the model

model.fit_generator(X_train, steps_per_epoch = 24, epochs = 10, validation_data = X_test, validation_steps = 40)
```

Python

```
C:\Users\mahes\AppData\Local\Temp/ipykernel_10216/1270027362.py:3: 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 = 24, epochs = 10, validation_data = X_test, validation_steps = 40)

Epoch 1/10
24/24 [==============================] - 26s 1s/step - loss: 1.4863 - accuracy: 0.5625 - val_loss: 0.6678 - val_accuracy: 0.7930
Epoch 2/10
24/24 [==============================] - 21s 878ms/step - loss: 0.5226 - accuracy: 0.8385 - val_loss: 0.3198 - val_accuracy: 0.9336
Epoch 3/10
24/24 [==============================] - 18s 759ms/step - loss: 0.3561 - accuracy: 0.8854 - val_loss: 0.3711 - val_accuracy: 0.9328
Epoch 4/10
24/24 [==============================] - 17s 711ms/step - loss: 0.2102 - accuracy: 0.9362 - val_loss: 0.2478 - val_accuracy: 0.9492
Epoch 5/10
24/24 [==============================] - 15s 638ms/step - loss: 0.1726 - accuracy: 0.9570 - val_loss: 0.2474 - val_accuracy: 0.9469
Epoch 6/10
24/24 [==============================] - 16s 648ms/step - loss: 0.1651 - accuracy: 0.9505 - val_loss: 0.2897 - val_accuracy: 0.9617
Epoch 7/10
24/24 [==============================] - 13s 560ms/step - loss: 0.1277 - accuracy: 0.9609 - val_loss: 0.2441 - val_accuracy: 0.9586
Epoch 8/10
24/24 [==============================] - 13s 543ms/step - loss: 0.0985 - accuracy: 0.9714 - val_loss: 0.2331 - val_accuracy: 0.9539
Epoch 9/10
24/24 [==============================] - 13s 528ms/step - loss: 0.0995 - accuracy: 0.9701 - val_loss: 0.2301 - val_accuracy: 0.9609
Epoch 10/10
24/24 [==============================] - 12s 503ms/step - loss: 0.0913 - accuracy: 0.9779 - val_loss: 0.2053 - val_accuracy: 0.9742

<keras.callbacks.History at 0x1d9801fe9d0>
```

```python
# Saving the model

model.save('aslpng1.h5')
```

[14]

Python


```python
# Fiiting the model

model.fit_generator(X_train, steps_per_epoch = 24, epochs = 10, validation_data = X_test, validation_steps = 40)
```

Python

```
C:\Users\mahes\AppData\Local\Temp/ipykernel_10216/1270027362.py:3: 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 = 24, epochs = 10, validation_data = X_test, validation_steps = 40)

Epoch 1/10
24/24 [==============================] - 26s 1s/step - loss: 1.4863 - accuracy: 0.5625 - val_loss: 0.6678 - val_accuracy: 0.7930
Epoch 2/10
24/24 [==============================] - 21s 878ms/step - loss: 0.5226 - accuracy: 0.8385 - val_loss: 0.3198 - val_accuracy: 0.9336
Epoch 3/10
24/24 [==============================] - 18s 759ms/step - loss: 0.3561 - accuracy: 0.8854 - val_loss: 0.3711 - val_accuracy: 0.9328
Epoch 4/10
24/24 [==============================] - 17s 711ms/step - loss: 0.2102 - accuracy: 0.9362 - val_loss: 0.2478 - val_accuracy: 0.9492
Epoch 5/10
24/24 [==============================] - 15s 638ms/step - loss: 0.1726 - accuracy: 0.9570 - val_loss: 0.2474 - val_accuracy: 0.9469
Epoch 6/10
24/24 [==============================] - 16s 648ms/step - loss: 0.1651 - accuracy: 0.9505 - val_loss: 0.2897 - val_accuracy: 0.9617
Epoch 7/10
24/24 [==============================] - 13s 560ms/step - loss: 0.1277 - accuracy: 0.9609 - val_loss: 0.2441 - val_accuracy: 0.9586
Epoch 8/10
24/24 [==============================] - 13s 543ms/step - loss: 0.0985 - accuracy: 0.9714 - val_loss: 0.2331 - val_accuracy: 0.9539
Epoch 9/10
24/24 [==============================] - 13s 528ms/step - loss: 0.0995 - accuracy: 0.9701 - val_loss: 0.2301 - val_accuracy: 0.9609
Epoch 10/10
24/24 [==============================] - 12s 503ms/step - loss: 0.0913 - accuracy: 0.9779 - val_loss: 0.2053 - val_accuracy: 0.9742

<keras.callbacks.History at 0x1d9801fe9d0>
```

```python
# Saving the model

model.save('aslpng1.h5')
```

[14]

Python

```python
# Importing Libarries

from tensorflow.keras.models import load_model
from tensorflow.keras.preprocessing import image
import numpy as np
import cv2
```

[1] Python

```python
# loading model

model = load_model('aslpng1.h5')
```

[2] Python

```python
from skimage.transform import resize
def detect(frame):
    img = resize(frame, (64, 64, 3))
    img = np.expand_dims(img, axis = 0)
    if np.max(img) > 1:
        img = img/255.0
    prediction = model.predict(img)
    print(prediction)
    return prediction
```

[3] Python

```python
while True:
    success, frame = video.read()
    cv2.imwrite('frame.jpg', frame)
    img = image.load_img('frame.jpg', target_size = (64, 64))

    x = image.img_to_array(img)
    x = cv2.cvtColor(x, cv2.COLOR_BGR2HSV)
    a = x.array_to_img(x)
    cv2.imshow("")
    x = np.expand_dims(x, axis = 0)
    pred = np.argmax(model.predict(x), axis = 1)

    y = pred[0]

    copy = frame.copy()


    cv2.rectangle(copy, (320, 100), (620, 400), (255, 0, 0), 5)
    cv2.putText(frame, "The Predicted Alphabet : " + str(index[y]), (100, 100), cv2.FONT_HERSHEY_SIMPLEX, 1, (0, 0, 0), 4)
    cv2.imshow('frame', frame)

    if cv2.waitKey(1) & 0xFF == ord('q'):
        break

video.release()
cv2.destroyAllWindows()
```

[10] Python

```
Output exceeds the size limit. Open the full output data in a text editor
1/1 [==============================] - 0s 44ms/step
1/1 [==============================] - 0s 24ms/step
1/1 [==============================] - 0s 19ms/step
1/1 [==============================] - 0s 16ms/step
```

```
frame = cv2.imread(r"D:\Maheshfiles\Studies\Smart Bridge\AI-ML-DL Project\Dataset\training_set\D\16.png")
data = detect(frame)
```
[4] Python

```
1/1 [==============================] - 0s 266ms/step
[[3.9748478e-08 1.2755189e-05 1.0463478e-08 9.9853325e-01 2.6569789e-06
  1.3680419e-05 4.5120544e-08 1.8048374e-07 1.4373119e-03]]
```

```
index = ['A','B','C','D','E','F','G','H','I']
index[np.argmax(data)]
```
[5] Python

```
'D'
```

OpenCV

```
# Importing Libraries

import cv2
import numpy as np
from tensorflow.keras.models import load_model
from tensorflow.keras.preprocessing import image
```
[6] Python

```
# Loading Model

model = load_model("aslpng1.h5")
```
[7] Python

```
video = cv2.VideoCapture(0)
index = ['A','B','C','D','E','F','G','H','I']
```
[8] Python

Image Preprocessing

```
# Importing Libraries

from tensorflow.keras.preprocessing.image import ImageDataGenerator
```
[23] Python

```
import os, types
import pandas as pd
from botocore.client import Config
import ibm_boto3

def __iter__(self): return 0

# @hidden_cell
# The following code accesses a file in your IBM Cloud Object Storage. It includes your credentials.
# You might want to remove those credentials before you share the notebook.
client_6b6e912f7eac460e813c136094c064e3 = ibm_boto3.client(service_name='s3',
    ibm_api_key_id='5iM4qWKjVM6YNNr3cFCfcTjBx8l8fE45F8bOdJndp8Y7',
    ibm_auth_endpoint="https://iam.cloud.ibm.com/oidc/token",
    config=Config(signature_version='oauth'),
    endpoint_url='https://s3.private.us.cloud-object-storage.appdomain.cloud')

streaming_body_1 = client_6b6e912f7eac460e813c136094c064e3.get_object(Bucket='aibasedrealtimecommunication-donotdelete-pr-uyelwdfsxweesp', Key='Dataset.zip')['

# Your data file was loaded into a botocore.response.StreamingBody object.
# Please read the documentation of ibm_boto3 and pandas to learn more about the possibilities to load the data.
# ibm_boto3 documentation: https://ibm.github.io/ibm-cos-sdk-python/
# pandas documentation: http://pandas.pydata.org/
```

```
[25] pwd
```
Python

```
'/home/wsuser/work'
```

```
[26] from io import BytesIO
     import zipfile
     unzip = zipfile.ZipFile(BytesIO(streaming_body_1.read()), 'r')
     file_paths = unzip.namelist()
     for path in file_paths:
         unzip.extract(path)
```
Python

```
[27] # Image Augmentation

     train_datagen = ImageDataGenerator(rescale = 1./255, shear_range = 0.2, zoom_range = 0.2, horizontal_flip = True)
     test_datagen = ImageDataGenerator(rescale = 1./255)
```
Python

```
[28] # Loading train and test set

     X_train = train_datagen.flow_from_directory(r"/home/wsuser/work/Dataset/training_set", target_size = (64, 64), batch_size = 32, class_mode = 'categorical')
     X_test = test_datagen.flow_from_directory(r"/home/wsuser/work/Dataset/test_set", target_size = (64, 64), batch_size = 32, class_mode = 'categorical')
```
Python

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

```
[29] # checking indices

     X_train.class_indices
```
Python

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

Model Building

```
[30] # Importing Libraries

     from tensorflow.keras.models import Sequential
     from tensorflow.keras.layers import Dense
     from tensorflow.keras.layers import Convolution2D, MaxPooling2D, Flatten
```
Python

```
[31] # Initializing the Model

     model = Sequential()
```
Python

```
[32] # Adding Convolution Layer

     model.add(Convolution2D((32), (3,3), input_shape = (64, 64, 3), activation = 'relu'))
```
Python

```python
# Adding Pooling Layer

model.add(MaxPooling2D(pool_size = (2, 2)))
```

[33] Python

```python
# Adding Flatten Layer

model.add(Flatten())
```

[34] Python

```python
# Adding Hidden Layer

model.add(Dense(units = 512, kernel_initializer = 'random_uniform', activation = 'relu'))
```

[35] Python

```python
# Adding Output Layer

model.add(Dense(units = 9, kernel_initializer = 'random_uniform', activation = 'softmax'))
```

[36] Python

```python
# Compile the model

model.compile(loss = 'categorical_crossentropy', optimizer = 'adam', metrics = ['accuracy'])
```

[37] Python

```python
# Fiiting the model

model.fit_generator(X_train, steps_per_epoch = 24, epochs = 10, validation_data = X_test, validation_steps = 40)
```

[38] Python

```
/tmp/wsuser/ipykernel_236/1270027362.py:3: 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 = 24, epochs = 10, validation_data = X_test, validation_steps = 40)

Epoch 1/10
24/24 [==============================] - 7s 282ms/step - loss: 1.2710 - accuracy: 0.5703 - val_loss: 0.6611 - val_accuracy: 0.7641
Epoch 2/10
24/24 [==============================] - 6s 251ms/step - loss: 0.4401 - accuracy: 0.8438 - val_loss: 0.4736 - val_accuracy: 0.8648
Epoch 3/10
24/24 [==============================] - 6s 272ms/step - loss: 0.2849 - accuracy: 0.9219 - val_loss: 0.3455 - val_accuracy: 0.9195
Epoch 4/10
24/24 [==============================] - 7s 275ms/step - loss: 0.1931 - accuracy: 0.9414 - val_loss: 0.3100 - val_accuracy: 0.9164
Epoch 5/10
24/24 [==============================] - 7s 274ms/step - loss: 0.1410 - accuracy: 0.9570 - val_loss: 0.2939 - val_accuracy: 0.9281
Epoch 6/10
24/24 [==============================] - 7s 277ms/step - loss: 0.1432 - accuracy: 0.9622 - val_loss: 0.2978 - val_accuracy: 0.9438
Epoch 7/10
24/24 [==============================] - 6s 264ms/step - loss: 0.1128 - accuracy: 0.9648 - val_loss: 0.2513 - val_accuracy: 0.9414
Epoch 8/10
24/24 [==============================] - 6s 260ms/step - loss: 0.0925 - accuracy: 0.9674 - val_loss: 0.3209 - val_accuracy: 0.9461
Epoch 9/10
24/24 [==============================] - 7s 273ms/step - loss: 0.1017 - accuracy: 0.9766 - val_loss: 0.3081 - val_accuracy: 0.9555
Epoch 10/10
24/24 [==============================] - 6s 253ms/step - loss: 0.0656 - accuracy: 0.9779 - val_loss: 0.2222 - val_accuracy: 0.9711

<keras.callbacks.History at 0x7f0b498b0f70>
```

```python
# Saving the model

model.save('aslpng1.h5')
```

[39] Python

```python
!tar -zcvf ai-based-real-time-classification-model.tgz aslpng1.h5
```

[52] Python

```
aslpng1.h5
```

```python
!pip install watson-machine-learning-client
```

[41] Python

```
Collecting watson-machine-learning-client
  Downloading watson_machine_learning_client-1.0.391-py3-none-any.whl (538 kB)
     |████████████████████████████████| 538 kB 23.2 MB/s eta 0:00:01
Requirement already satisfied: certifi in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from watson-machine-learning-client) (2022.6.15)
Requirement already satisfied: ibm-cos-sdk in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from watson-machine-learning-client) (2.11.0)
Requirement already satisfied: requests in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from watson-machine-learning-client) (2.26.0)
Requirement already satisfied: urllib3 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from watson-machine-learning-client) (1.26.7)
Requirement already satisfied: boto3 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from watson-machine-learning-client) (1.18.21)
Requirement already satisfied: tqdm in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from watson-machine-learning-client) (4.62.3)
Requirement already satisfied: pandas in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from watson-machine-learning-client) (1.3.4)
Requirement already satisfied: lomond in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from watson-machine-learning-client) (0.3.3)
Requirement already satisfied: tabulate in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from watson-machine-learning-client) (0.8.9)
Requirement already satisfied: botocore<1.22.0,>=1.21.21 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from boto3->watson-machine-learning-client)
(1.21.41)
Requirement already satisfied: s3transfer<0.6.0,>=0.5.0 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from boto3->watson-machine-learning-client)
```

```python
from ibm_watson_machine_learning import APIClient
wml_credentials = {
    "url" : "https://us-south.ml.cloud.ibm.com",
    "apikey" : "T7rpH1KfTn-s-zfDCmTyeArxYrcvFGHFV21qTDW5pf5x"
}
client = APIClient(wml_credentials)
```

[42] Python

```python
def guid_space_name(client, ai_based_real_time_communication_deploy_space):
    space = client.spaces.get_details()
    return(next(item for item in space['resources'] if item['entity']['name'] == ai_based_real_time_communication_deploy_space)['metadata']['id'])
```

[43] Python

```python
client.spaces.get_details()
```

[45] Python

```
Output exceeds the size limit. Open the full output data in a text editor
{'resources': [{'entity': {'compute': [{'crn': 'crn:v1:bluemix:public:pm-20:us-south:a/5be23fa7fba94c8aa2e3db0b2a4db8d2:04f159f4-9ffb-4e0d-b70b-2a1f3b216970::',
      'guid': '04f159f4-9ffb-4e0d-b70b-2a1f3b216970',
      'name': 'Watson Machine Learning-av',
      'type': 'machine_learning'}],
    'description': '',
    'name': 'ai_based_real_time_communication_deploy_space',
    'scope': {'bss_account_id': '5be23fa7fba94c8aa2e3db0b2a4db8d2'},
    'stage': {'production': False},
    'status': {'state': 'active'},
    'storage': {'properties': {'bucket_name': '0a42d73b-35af-4f9d-92da-4a84147fcb1c',
      'bucket_region': 'us-south',
```

```
space_id = guid_space_name(client, 'ai_based_real_time_communication_deploy_space')
space_id
```

[47] Python

```
'1853d74e-ca3c-4075-81e3-d5cdd0741a52'
```

```
client.set.default_space(space_id)
```

[48] Python

```
'SUCCESS'
```

```
client.software_specifications.list(100)
```

[50] Python

```
Output exceeds the size limit. Open the full output data in a text editor
------------------------------  ------------------------------------  ----
NAME                            ASSET_ID                              TYPE
default_py3.6                   0062b8c9-8b7d-44a0-a9b9-46c416adcbd9  base
kernel-spark3.2-scala2.12       020d69ce-7ac1-5e68-ac1a-31189867356a  base
pytorch-onnx_1.3-py3.7-edt      069ea134-3346-5748-b513-49120e15d288  base
scikit-learn_0.20-py3.6         09c5a1d0-9c1e-4473-a344-eb7b665ff687  base
spark-mllib_3.0-scala_2.12      09f4cff0-90a7-5899-b9ed-1ef348aebdee  base
pytorch-onnx_rt22.1-py3.9       0b848dd4-e681-5599-be41-b5f6fccc6471  base
ai-function_0.1-py3.6           0cdb0f1e-5376-4f4d-92dd-da3b69aa9bda  base
shiny-r3.6                      0e6e79df-875e-4f24-8ae9-62dcc2148306  base
tensorflow_2.4-py3.7-horovod    1092590a-307d-563d-9b62-4eb7d64b3f22  base
pytorch_1.1-py3.6               10ac12d6-6b30-4ccd-8392-3e922c096a92  base
tensorflow_1.15-py3.6-ddl       111e41b3-de2d-5422-a4d6-bf776828c4b7  base
```

```
software_space_id = client.software_specifications.get_uid_by_name('tensorflow_rt22.1-py3.9')
software_space_id
```

[51] Python

```
'acd9c798-6974-5d2f-a657-ce06e986df4d'
```

```
model_details = client.repository.store_model(model = 'ai-based-real-time-classification-model.tgz', meta_props = {
    client.repository.ModelMetaNames.NAME:"CNN Model Buiding",
    client.repository.ModelMetaNames.TYPE:"tensorflow_2.7",
    client.repository.ModelMetaNames.SOFTWARE_SPEC_UID:software_space_id
})
```

[53] Python

```
model_id = client.repository.get_model_id(model_details)
model_id
```

[54] Python

```
'59b18265-3a03-47d3-b2d8-d9a0c5106f05'
```

```
client.repository.download(model_id, 'ai-based-real-time-classification-model.h5')
```

[56] Python

```
Successfully saved model content to file: 'ai-based-real-time-classification-model.h5'

'/home/wsuser/work/ai-based-real-time-classification-model.h5'
```

Web Application in Flask:

webstreaming.py

```python
from flask import Flask,render_template,request
import cv2
from keras.models import load_model
import numpy as np
from gtts import gTTS
import os
from keras.preprocessing import image
from skimage.transform import resize
from playsound import playsound
app = Flask(__name__)

model=load_model("aslpng1.h5")

vals = ['A', 'B','C','D','E','F','G','H','I']

@app.route('/', methods=['GET'])
def index():
    return render_template('index.html')
@app.route('/index', methods=['GET'])
def home():
    return render_template('index.html')
@app.route('/predict', methods=['GET', 'POST'])
def predict():
        print("[INFO] starting video stream...")
        vs = cv2.VideoCapture(0)

        (W, H) = (None, None)

        while True:
            (grabbed, frame) = vs.read()

            if not grabbed:
                break

            if W is None or H is None:
                (H, W) = frame.shape[:2]
            output = frame.copy()
            # r = cv2.selectROI("Slect", output)
            # print(r)
            cv2.rectangle(output, (81, 79), (276,274), (0,255,0), 2)
            frame = frame[81:276, 79:274]
            frame = cv2.cvtColor(frame, cv2.COLOR_RGB2GRAY)
            _, frame = cv2.threshold(frame, 95, 255, cv2.THRESH_BINARY_INV)
            frame = cv2.cvtColor(frame, cv2.COLOR_GRAY2RGB)


            img = resize(frame,(64,64,3))
            img = np.expand_dims(img,axis=0)
            if(np.max(img)>1):
                img = img/255.0


            result = np.argmax(model.predict(img))
            index=['A', 'B','C','D','E','F','G','H','I']
            result=str(index[result])


            cv2.putText(output, "The Predicted Letter : {}".format(result), (10, 50), cv2.FONT_HERSHEY_PLAIN,
                        2, (150,0,150), 2)
            cv2.putText(output, "Press q to exit", (10,450), cv2.FONT_HERSHEY_PLAIN, 2, (0,0,255), 2)


            speech = gTTS(text = result, lang = 'en', slow = False)

            cv2.imshow("Output", output)
            key = cv2.waitKey(1) & 0xFF

            if key == ord("q"):
                break


        print("[INFO] cleaning up...")
        vs.release()
        cv2.destroyAllWindows()
        return render_template("index.html")


if __name__ == '__main__':
    app.run(debug=False)
```

index.html

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>AI Based Real time Communication</title>
</head>
<body>
    <h2>Show some Gestures with your Hand.. By Clicking on the <a href="{{ url_for('predict') }}">Camera</a></h2>
    <div>
        <img src = 'https://www.researchgate.net/publication/274480405/figure/fig4/
        AS:668304138063878@1536347531535/26-sample-hand-gesture-images-for-26-letters-in-American-Sign-Language.jpg'  width = 50%>

    </div>
</body>
</html>
```

13.2 Github & demo link:

Github link

https://github.com/IBM-EPBL/IBM-Project-41261-1660640730

Demo link:

https://drive.google.com/drive/folders/15z5tqUw2T8FsNrR9SQtdFn-6IcXVb-7L?usp=share_link