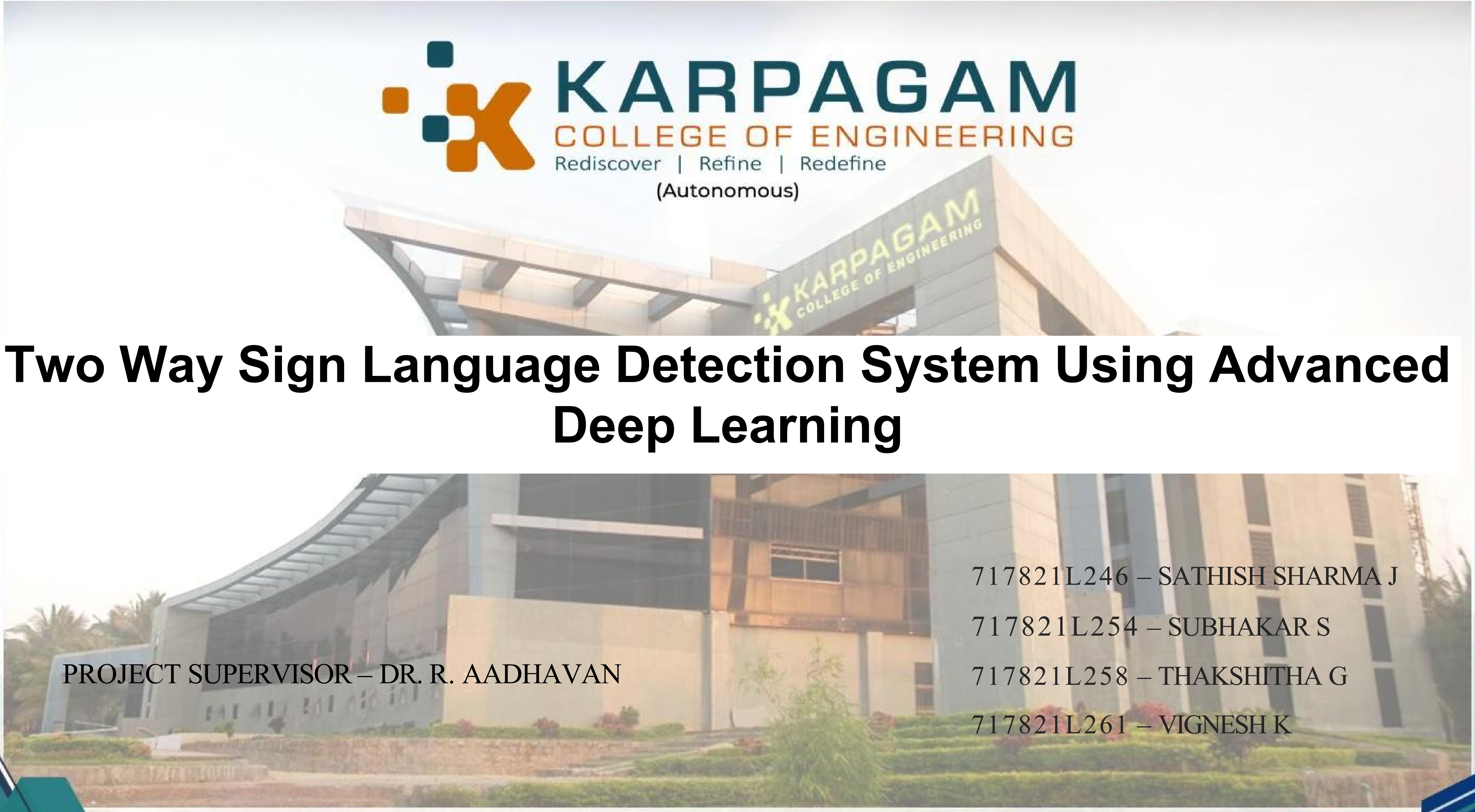


KARPAGAM
COLLEGE OF ENGINEERING
Rediscover | Refine | Redefine
(Autonomous)

Two Way Sign Language Detection System Using Advanced Deep Learning

PROJECT SUPERVISOR – DR. R. AADHAVAN

717821L246 – SATHISH SHARMA J

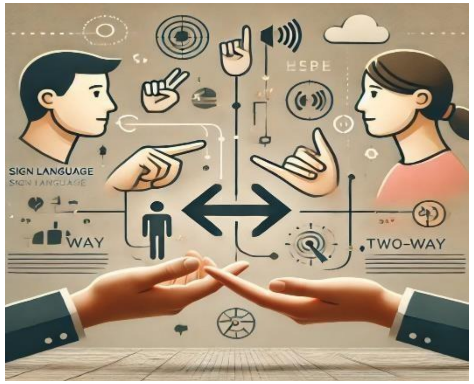
717821L254 – SUBHAKAR S

717821L258 – THAKSHITHA G

717821L261 – VIGNESH K

INTRODUCTION

- Sign language is a crucial mode of communication for the people with hearing disability and hard-of-hearing community, but it is often a barrier to interaction with the hearing population.

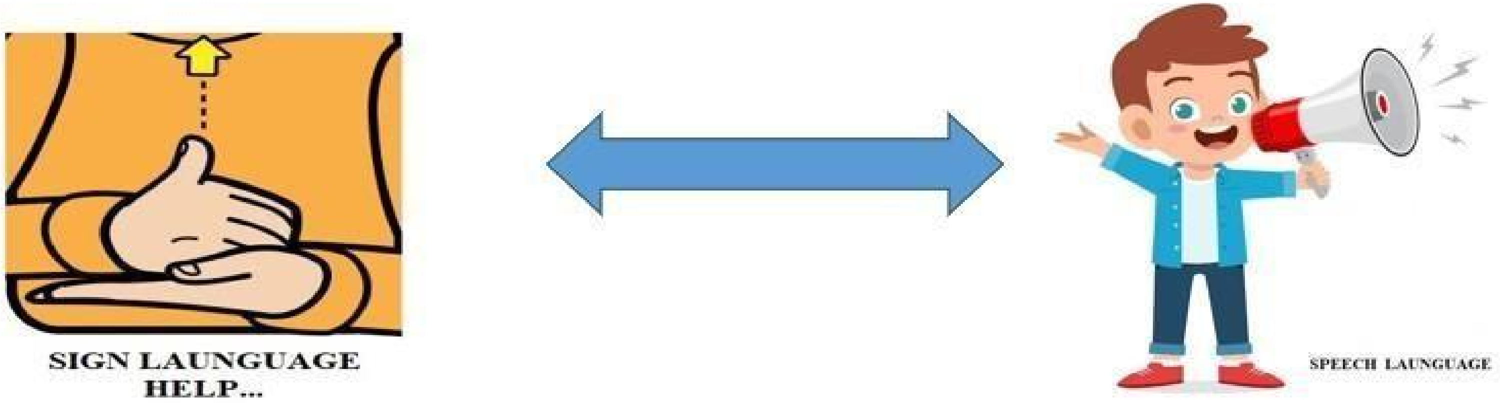


- The proposed functionality aims to bridge communication gaps and promote inclusivity, making interactions more accessible for individuals using sign language in English.

- Our project addresses this challenge by developing a two-way sign language translation system in English. This proposed system enables both conversion from sign language to speech and from speech to sign language.

- For the sign-to-speech conversion, we employ a modified Convolutional Neural Network (CNN) architecture that classifies various signs and translates them into spoken language using Google Text-to-Speech (GTTS).

- Conversely, for speech-to-sign conversion, the system uses PySpeech recognition to transcribe spoken language into text, which is then mapped to the corresponding sign language gestures using open source computer vision(OpenCV).

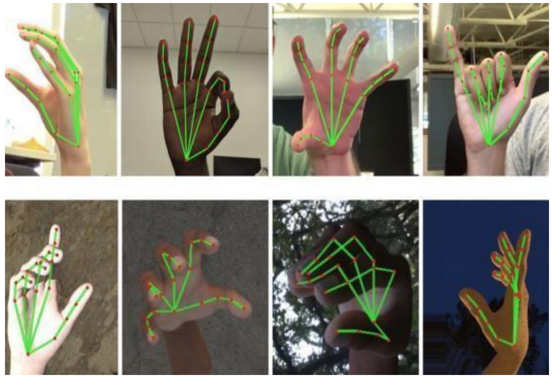


NEED FOR CURRENT WORK

- There's a crucial need for a system that can seamlessly translate both sign language to speech and speech to sign language in English.
- Two-way communication allows professionals to interact with others and build relationships.

OBJECTIVE OF THE WORK:

- Our project aims to bridge the communication gap with a comprehensive system that enables two-way communication between sign language and speech in English using advanced technologies like Modified CNN(Modified Convolution Neural Network) for sign recognition and speech-to-text conversion.



KARPAGAM
COLLEGE OF ENGINEERING
Rediscover | Refine | Redefine
(Autonomous)

LITERATURE REVIEW

S.NO	TITLE OF THE PAPER	AUTHOR	JOURNAL NAME AND YEAR	REMARKS & LIMITATIONS
1.	SignNet II: A Transformer-Based Two- Way Sign Language Translation Model	Lipisha Chaudhary, Tejaswini Ananthanarayana, Enjamamul Hoq, Ifeoma Nwogu	IEEE Transactions on Pattern Analysis and Machine Intelligence 2022	• SignNet II is a sign language processing architecture that uses transformer networks for two-way communication • Outperforms singly-trained networks on German Sign Language benchmark dataset.
2.	Collaborative Multilingual Continuous Sign Language Recognition: A Unified Framework	Hezhen Hu,Junfu Pu,Wengang Zhou,Houqiang Li	IEEE Transactions on Multimedia 2022	• Addresses multilingual sign language recognition with a unified framework. • Includes a shared visual encoder, language-dependent sequential modules, and a universal sequential module. • Outperforms individually trained recognition models on multiple benchmarks.
3.	Recognizing British Sign Language Using Deep Learning: A Contactless and Privacy-Preserving Approach	Hira Hameed,Kashif Ahmad,Amir Hussain,Muhammad Ali Imran,Qammer H. Abbasi	IEEE Transactions on Computational Social Systems 2022	• Proposes a contactless and privacy-preserving system for British Sign Language (BSL) recognition. • Uses radar data and deep learning models to extract and classify spatiotemporal features of BSL signs.
4.	Deep sign: Sign Language Detection and Recognition Using Deep Learning	Kothadiya, Deep, Chintan Bhatt, Krenil Sapariya, Kevin Patel, Ana-Belén Gil-González, and Juan M. Corchado	IEEE Electronics 2022	• Since, it is based on thermal processing, high accuracy of 99.52% can be achieved.

S.NO	TITLE OF THE PAPER	AUTHOR	JOURNAL NAME AND YEAR	REMARKS & LIMITATIONS
5.	Automated Sign Language Alphabet Detection	van der Merwe, Ashwin, Elie Ngomseu Mambou, and Theo G. Swart.	IEEE Communications of the Association for Information Systems 2021	Only limited signs can be detected using this method.
6.	Deep Learning-Based Sign Language Digits Recognition From Thermal Images With Edge Computing System	Breland, Daniel S., Simen B.Skriubakken, Aveen Dayal Ajit Jha, Phaneendra K. Yalavarthy, and Linga Reddy Cenkeramaddi	IEEE Sensors Journal 2021	Since, it is based on thermal processing, high accuracy of 99.52% can be achieved.
7.	Deep Learning for Sign Language Recognition: Current Techniques, Benchmarks, and Open Issues	Al-Qurishi, Muhammad, Thariq Khalid, and Riad Souissi	IEEE Access 2021	Many SLR tools can be used along with image processing for future applications.
8.	Deep learning-based sign Language recognition system for static signs	Ankita Wadhavan, ParteekKumar	Springer Neural computing and applications 2021	System is robust enough to learn 100 different static manual signs with lower error rates

S.NO	TITLE OF THE PAPER	AUTHOR	JOURNAL NAME AND YEAR	REMARKS & LIMITATIONS
9.	Sign language detection and recognition	I.A. Adeyanju , O.O. Bello b , M.A. Adegboye	Elsevier Intelligent Systems with Applications 2021	• There is a need for more research that fuses images from multiple devices such as a camera
10.	Sign Language Recognition: A Deep Survey	Rastgoo, Razieh, Kourosh Kiani, and Sergio Escalera	Expert Systems with Applications 2021	• It covers various aspects of the field, including data collection, preprocessing, feature extraction, and classification methods. • The survey summarizes the advancements, challenges, and future directions in sign language recognition using deep learning.
11.	A Comprehensive Study on Deep Learning-Based Methods for Sign Language Recognition	Adaloglou, Nikolas, Theocharis Chatzis, Ilias Papastratis, Andreas Stergioulas, Georgios Th Papadopoulos	IEEE Transactions on Multimedia 2021	• Evaluates recent deep neural network methods on multiple publicly available datasets. • Introduces new sequence training criteria and discusses various pretraining schemes.
12.	Hear Sign Language: A Real-Time End-to-End Sign Language Recognition System	Wang, Zhibo, Tengda Zhao, Jinxin Ma, Hongkai Chen, Kaixin Liu, Huajie Shao, Qian Wang, and Ju Ren	IEEE Transactions on Mobile Computing 2020	• DeepSLR, a real-time end-to-end sign language recognition (SLR) System is introduced and it is implemented on a smartphone. • Achieving an average word error rate of 10.8% and less than 1.1s recognition time for 4 sign words.

PROBLEM STATEMENT

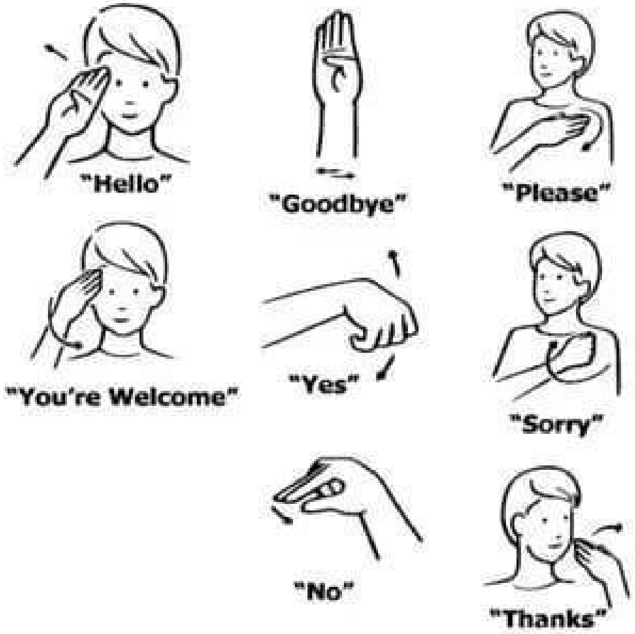
KARPAGAM
COLLEGE OF ENGINEERING
Rediscover | Refine | Redefine
(Autonomous)

- The lack of real-time, accurate, and user-friendly translation tools creates barriers in various contexts, including education, professional settings, and everyday interactions to impaired people.
- Effective communication between sign language users and non-users is challenging, as current solutions often only handle both-way translation rather than providing a comprehensive bidirectional approach.
- This project seeks to address these issues by developing a two-way sign language translation system that seamlessly translates between sign language and speech.

DUAL COMMUNICATION

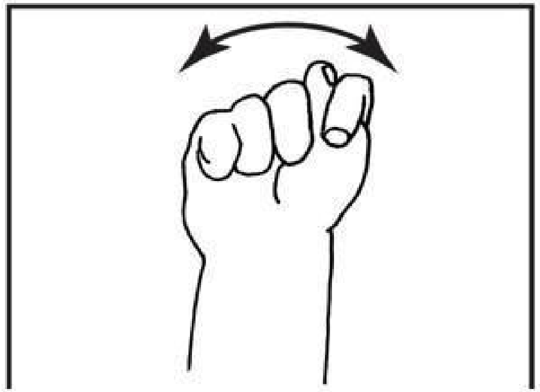
EXISTING METHOD

- ✓ Current sign language translation systems are often limited in scope, focusing primarily on either converting sign language to speech or speech to sign language in English.
- ✓ Current sign-to-speech systems often use pre defined databases or simple rules, lacking flexibility and accuracy, while speech-to-sign systems depend on basic voice recognition and limited gesture databases, missing the full range of sign language.



✓ While some systems integrate machine learning techniques, they often fall short in real-time performance and bidirectional translation capabilities. Current technologies also face challenges in ensuring high accuracy and user-friendliness, particularly in dynamic, conversational settings.

I WANT TO GO REST ROOM

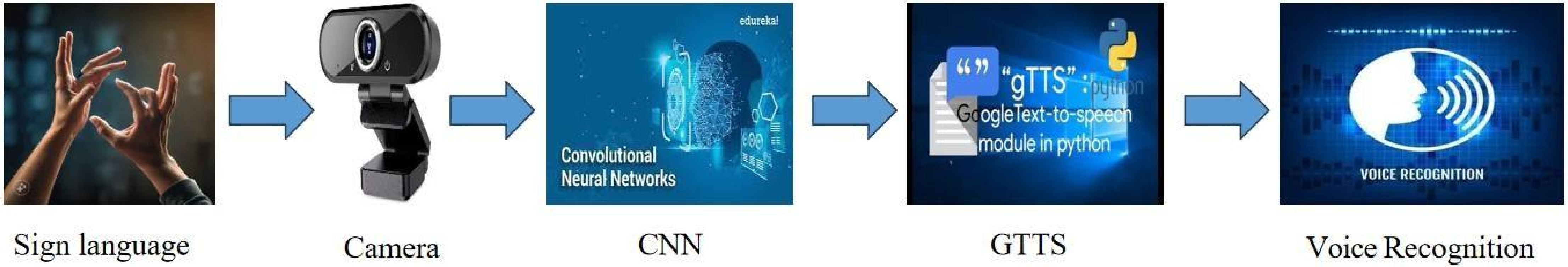


I WILL CALL YOU

PROPOSED METHOD

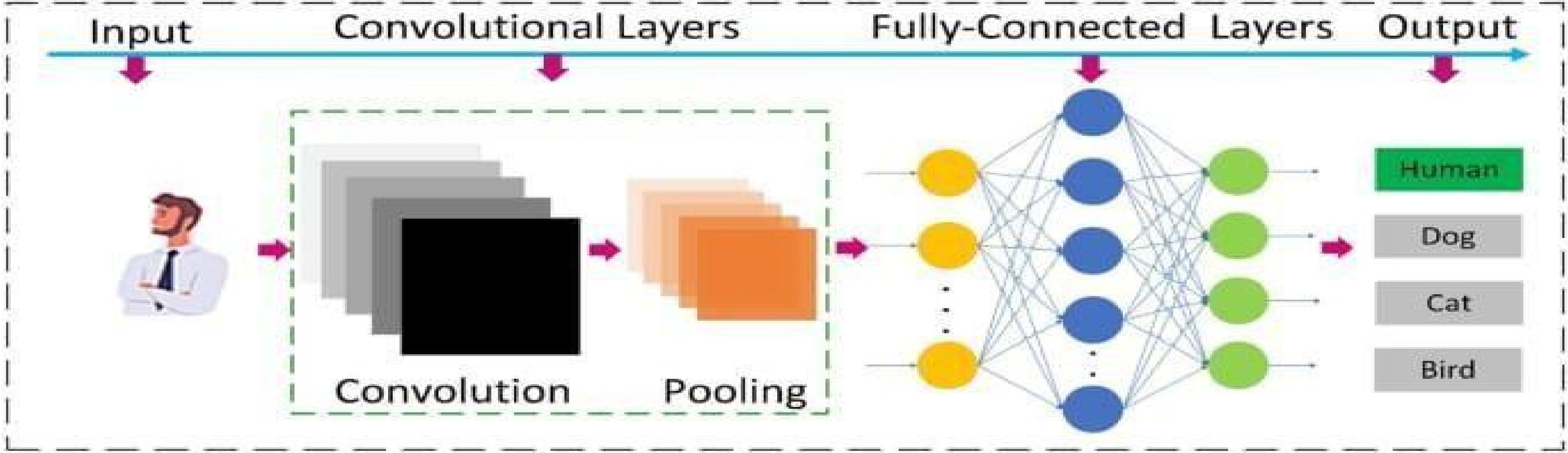
- **Sign-to-Speech Conversion:** The system employs a modified Convolutional Neural Network (CNN) to classify hand signs captured through a webcam. Once the signs are recognized, they are converted into text and then translated into spoken language using Google Text-to-Speech (GTTS).

WORK PROCESS:



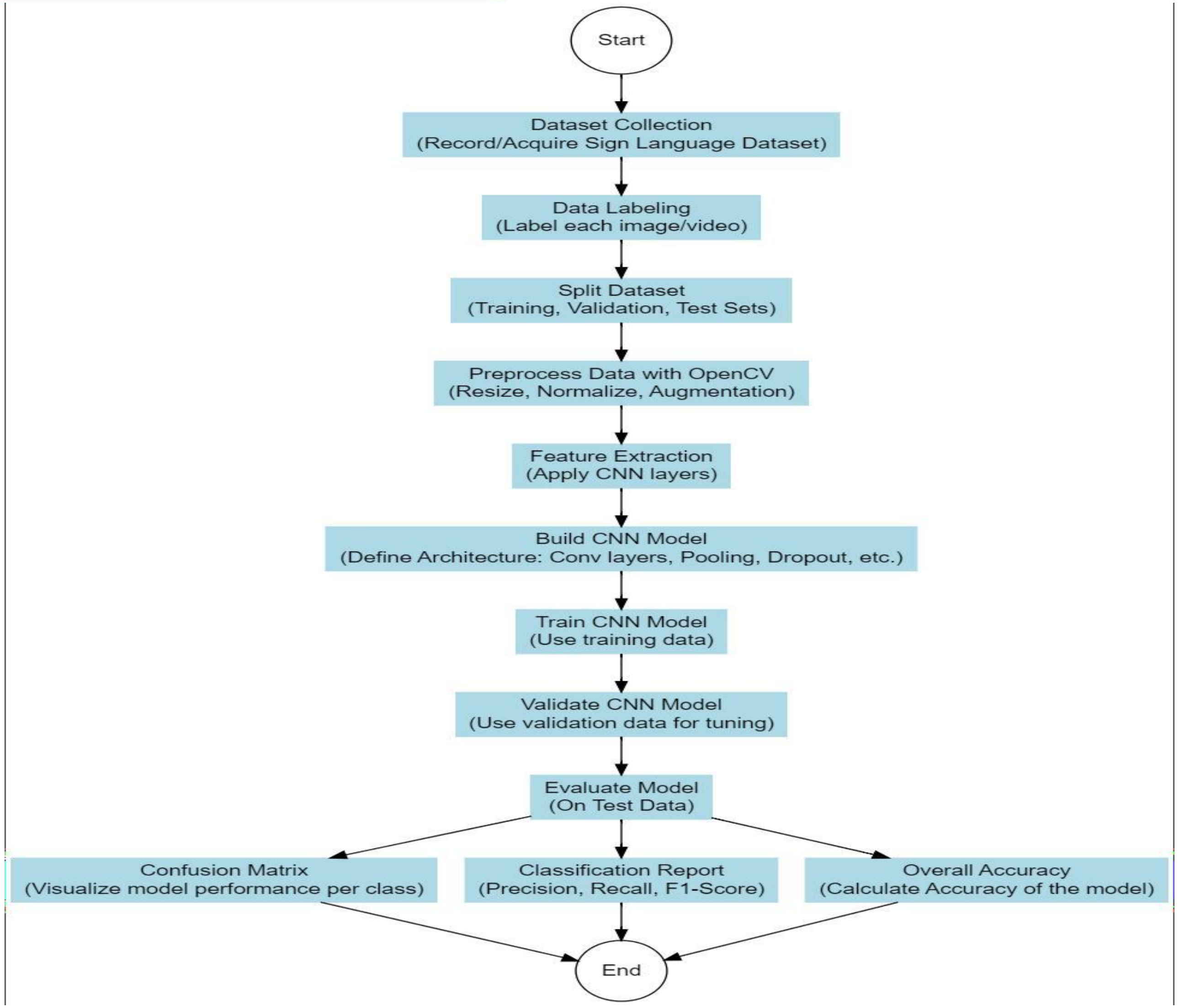
WORKING PRINCIPLE OF CNN

- A convolutional neural network consists of an input layer, hidden layers and an output layer.
- In a convolutional neural network, the hidden layers include one or more layers that perform convolutions.
- Typically this includes a layer that performs a dot product of the convolution kernel with the layer's input matrix.



BLOCK DIAGRAM OF CNN

FLOW CHART OF SIGN TO SPEECH



Start

Dataset Collection
(Record/Acquire Sign Language Dataset)

Data Labeling
(Label each image/video)

Split Dataset
(Training, Validation, Test Sets)

Preprocess Data with OpenCV
(Resize, Normalize, Augmentation)

Feature Extraction
(Apply CNN layers)

Build CNN Model
(Define Architecture: Conv layers, Pooling, Dropout, etc.)

Train CNN Model
(Use training data)

Validate CNN Model
(Use validation data for tuning)

Evaluate Model
(On Test Data)

Confusion Matrix
(Visualize model performance per class)

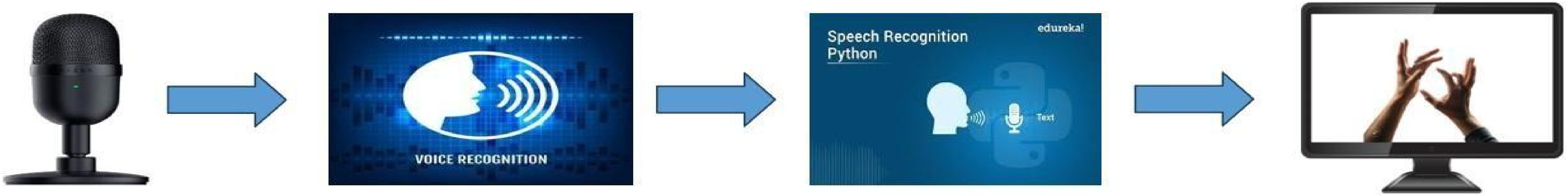
Classification Report
(Precision, Recall, F1-Score)

Overall Accuracy
(Calculate Accuracy of the model)

End

- **Speech-to-Sign Conversion:** For the reverse process, the system utilizes PySpeech recognition to capture spoken language, transcribe it into text, and map the text to corresponding sign language gestures using OpenCV.

WORK PROCESS:



Microphone

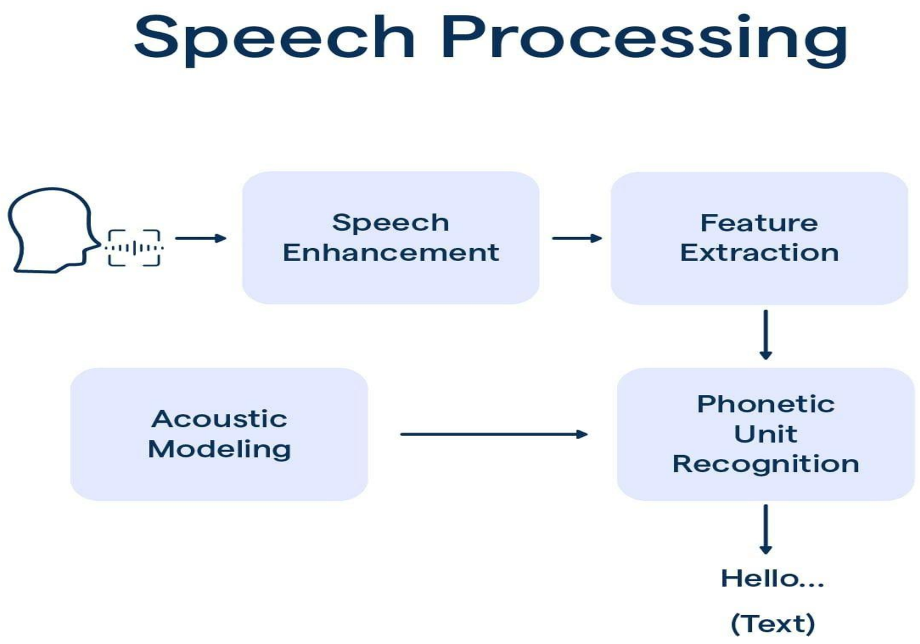
Voice Recognition

Speech Recognition python

Sign language in System

SPEECH RECOGNITION

A speech recognition system, also known as automatic speech recognition (ASR) or speech-to-text, is a technology that converts spoken words into text or commands.


Speech Processing

Speech Enhancement

Feature Extraction

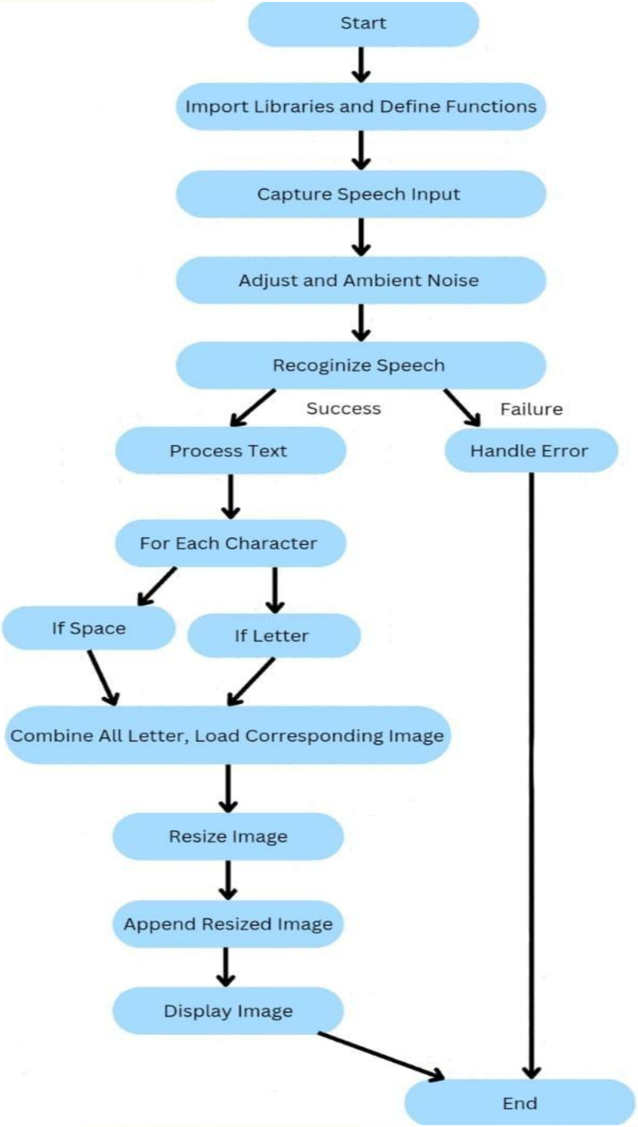
Acoustic Modeling

Phonetic Unit Recognition

Hello...

(Text)

FLOW CHART TO SPEECH TO SIGN



Start

Import Libraries and Define Functions

Capture Speech Input

Adjust and Ambient Noise

Recoginize Speech

Success

Failure

Process Text

Handle Error

For Each Character

If Space

If Letter

Combine All Letter, Load Corresponding Image

Resize Image

Append Resized Image

Display Image

End

IDENTIFICATION OF HARDWARE AND SOFTWARE

HARDWARE:

- **Webcam:** Captures real-time video input of hand signs for processing by the CNN model in the sign-to-speech module.

SPECIFICATION:

✓ Resolution : 1080p (Full HD)
✓ Pixel Count : Approximately 2 megapixel
✓ Lens Type : Fixed focus or Autofocus
✓ Features : Improved image clarity

- **Microphone:** Records spoken language to be transcribed into text in the speech-to-sign module.

SPECIFICATION:

✓ Type : Dual-Array or multiple microphones

✓ Frequency response : 50HZ to 20kHZ

✓ Range:

  ▪ Distance = up to 1 meter(3.3 feet)

  ▪ Angle = 360° (Omnidirectional) or 60° to 120° (Unidirectional)

✓ Features : Noise cancellation, Echo cancellation

- **Speakers:** Outputs the spoken translation of recognized signs, providing auditory feedback in the sign-to-speech module.

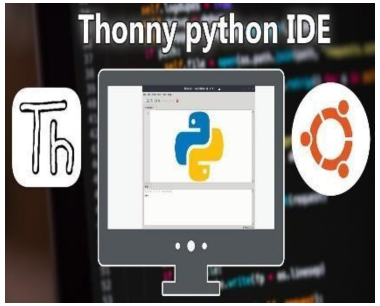
SPECIFICATION:

✓ Type : Stereo or sometimes with small subwoofers
✓ Frequency response : 60HZ to 20kHZ
✓ Range : up to 5 to 8 feet (1.5-2.5 meters)
✓ Features : Better sound Quality and volume

SOFTWARE:

Language: Python 3.7

IDE: Thonny



Libraries:

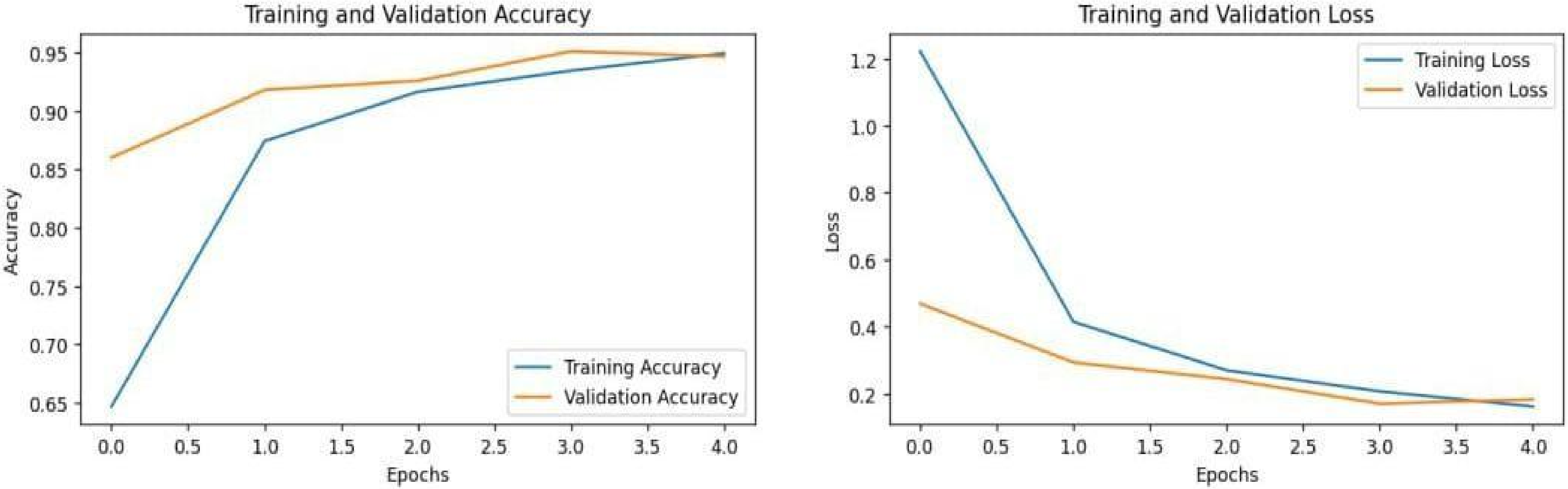
- **TensorFlow/Keras:** Utilized for developing and training the modified Convolutional Neural Network (CNN) model to classify sign language gestures.

- **PySpeech:** A Python library used for speech recognition, transcribing spoken language into text for processing.

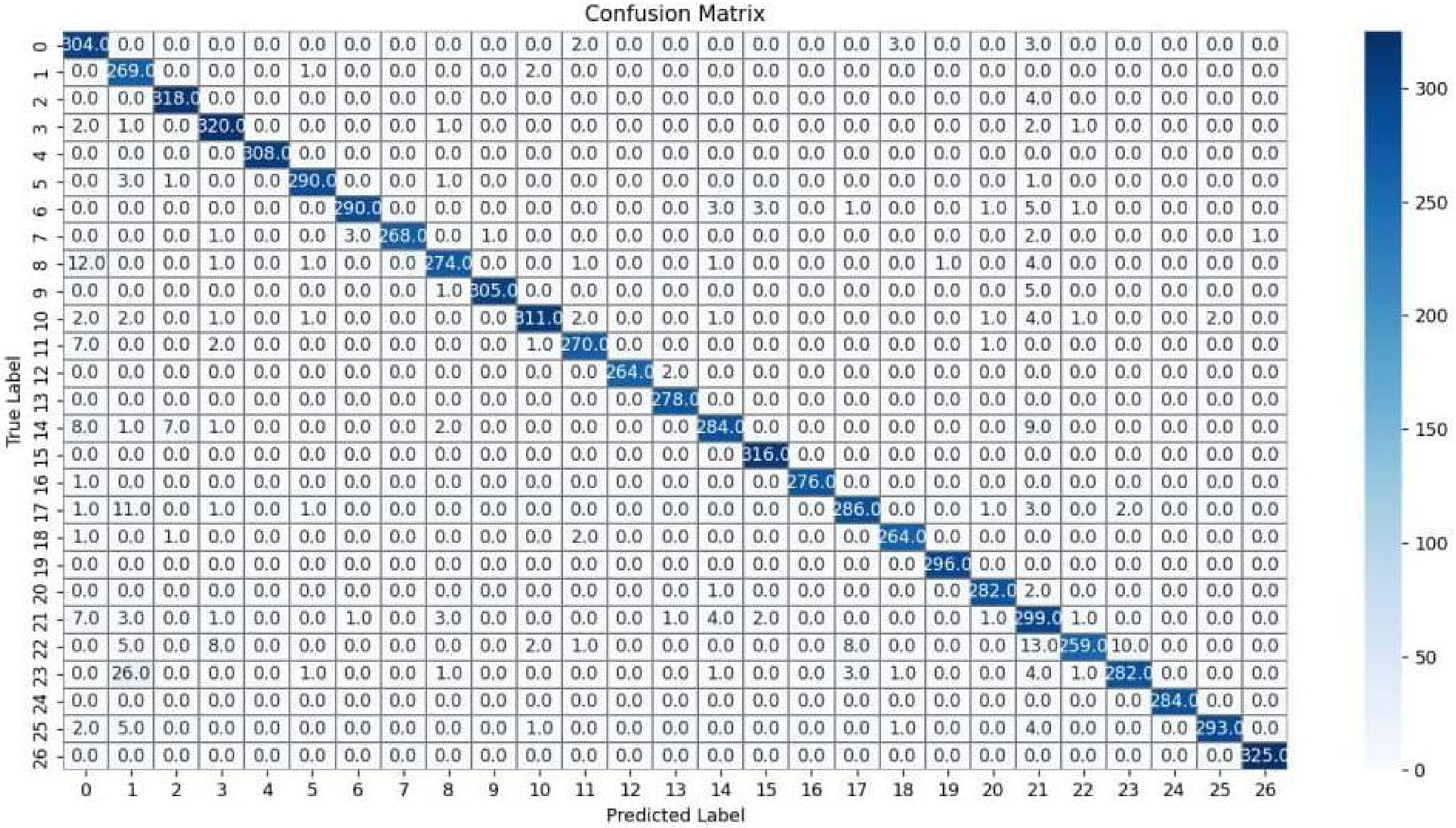
- **OpenCV:** Employed for mapping transcribed text to corresponding sign language gestures, enabling visual representation of signs.

RESULTS

Validation Accuracy and Loss



Confusion Matrix

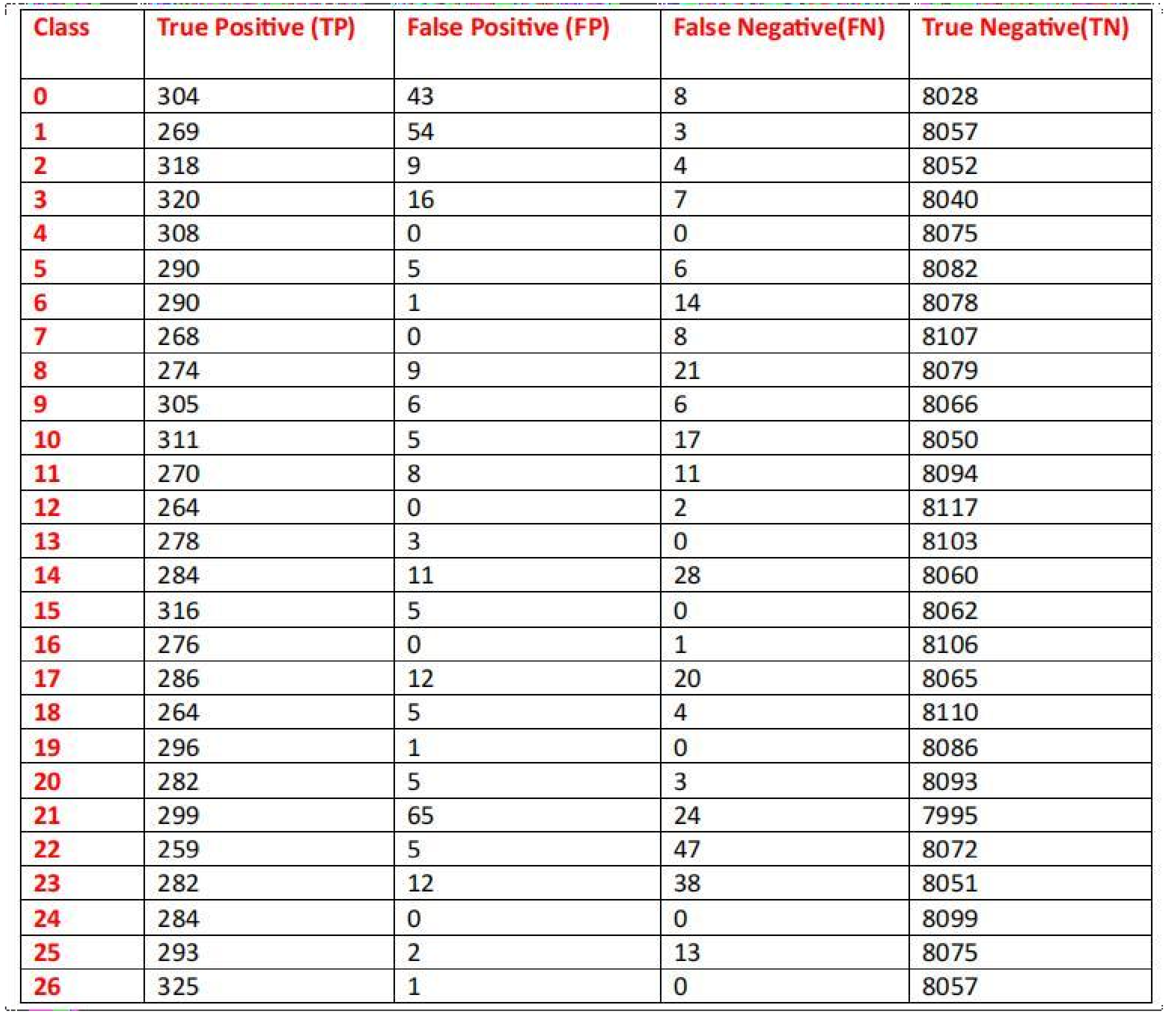


Confusion Matrix

True Label / Predicted Label

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26
0	304.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	2.0	0.0	0.0	0.0	0.0	0.0	0.0	3.0	0.0	0.0	3.0	0.0	0.0	0.0	0.0	0.0
1	0.0	269.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	2.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
2	0.0	0.0	318.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	4.0	0.0	0.0	0.0	0.0	0.0
3	2.0	1.0	0.0	320.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	2.0	1.0	0.0	0.0	0.0	0.0
4	0.0	0.0	0.0	0.0	308.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
5	0.0	3.0	1.0	0.0	0.0	290.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0
6	0.0	0.0	0.0	0.0	0.0	0.0	290.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	3.0	3.0	0.0	1.0	0.0	0.0	1.0	5.0	1.0	0.0	0.0	0.0	0.0
7	0.0	0.0	0.0	1.0	0.0	0.0	3.0	268.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	2.0	0.0	0.0	0.0	0.0	1.0
8	12.0	0.0	0.0	1.0	0.0	1.0	0.0	0.0	274.0	0.0	0.0	1.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	1.0	0.0	4.0	0.0	0.0	0.0	0.0	0.0
9	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0	305.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	5.0	0.0	0.0	0.0	0.0	0.0
10	2.0	2.0	0.0	1.0	0.0	1.0	0.0	0.0	0.0	0.0	311.0	2.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	1.0	4.0	1.0	0.0	0.0	2.0	0.0
11	7.0	0.0	0.0	2.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0	270.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0
12	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	264.0	2.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
13	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	278.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
14	8.0	1.0	7.0	1.0	0.0	0.0	0.0	0.0	2.0	0.0	0.0	0.0	0.0	0.0	284.0	0.0	0.0	0.0	0.0	0.0	0.0	9.0	0.0	0.0	0.0	0.0	0.0
15	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	316.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
16	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	276.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
17	1.0	11.0	0.0	1.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	286.0	0.0	0.0	1.0	3.0	0.0	2.0	0.0	0.0	0.0
18	1.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	2.0	0.0	0.0	0.0	0.0	0.0	0.0	264.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
19	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	296.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
20	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	282.0	2.0	0.0	0.0	0.0	0.0	0.0
21	7.0	3.0	0.0	1.0	0.0	0.0	1.0	0.0	3.0	0.0	0.0	0.0	0.0	1.0	4.0	2.0	0.0	0.0	0.0	0.0	1.0	299.0	1.0	0.0	0.0	0.0	0.0
22	0.0	5.0	0.0	8.0	0.0	0.0	0.0	0.0	0.0	0.0	2.0	1.0	0.0	0.0	0.0	0.0	0.0	8.0	0.0	0.0	0.0	13.0	259.0	10.0	0.0	0.0	0.0
23	0.0	26.0	0.0	0.0	0.0	1.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	3.0	1.0	0.0	0.0	4.0	1.0	282.0	0.0	0.0	0.0
24	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	284.0	0.0	0.0
25	2.0	5.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	4.0	0.0	0.0	0.0	293.0	0.0
26	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	325.0

Table Of True Positive (TP) ,True Negative (TN) ,False Positive (FP) , False Negative (FN)



Class	True Positive (TP)	False Positive (FP)	False Negative(FN)	True Negative(TN)
0	304	43	8	8028
1	269	54	3	8057
2	318	9	4	8052
3	320	16	7	8040
4	308	0	0	8075
5	290	5	6	8082
6	290	1	14	8078
7	268	0	8	8107
8	274	9	21	8079
9	305	6	6	8066
10	311	5	17	8050
11	270	8	11	8094
12	264	0	2	8117
13	278	3	0	8103
14	284	11	28	8060
15	316	5	0	8062
16	276	0	1	8106
17	286	12	20	8065
18	264	5	4	8110
19	296	1	0	8086
20	282	5	3	8093
21	299	65	24	7995
22	259	5	47	8072
23	282	12	38	8051
24	284	0	0	8099
25	293	2	13	8075
26	325	1	0	8057

Calculation of [TP,FP,FN,TN] :

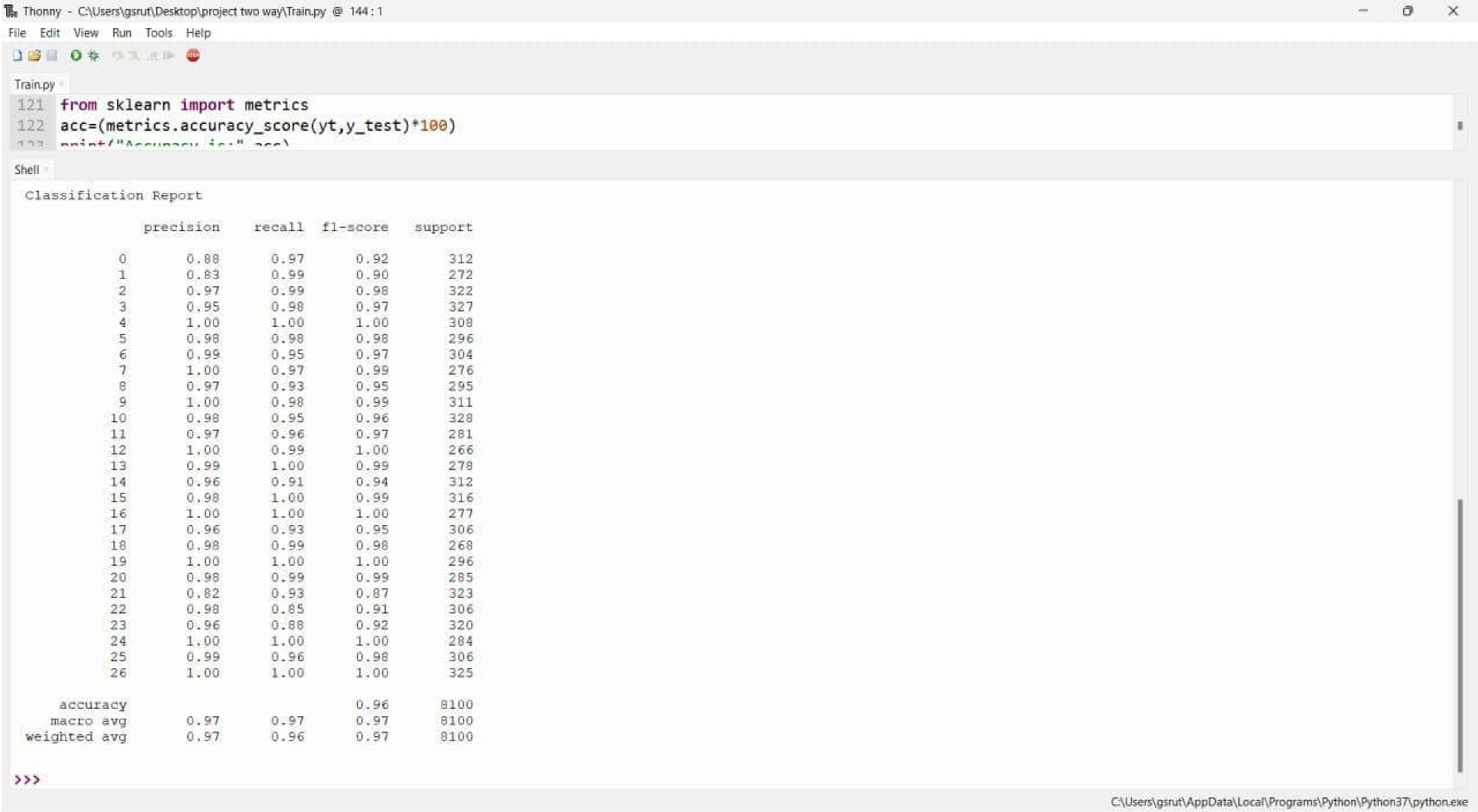
TP = Diagonal Value From the Matrix

FP = Sum of Column 0 (excluding TP)

FN = Sum of row 0 (excluding TP)

TN Total Sum of all Elements - (TP+FP+FN)

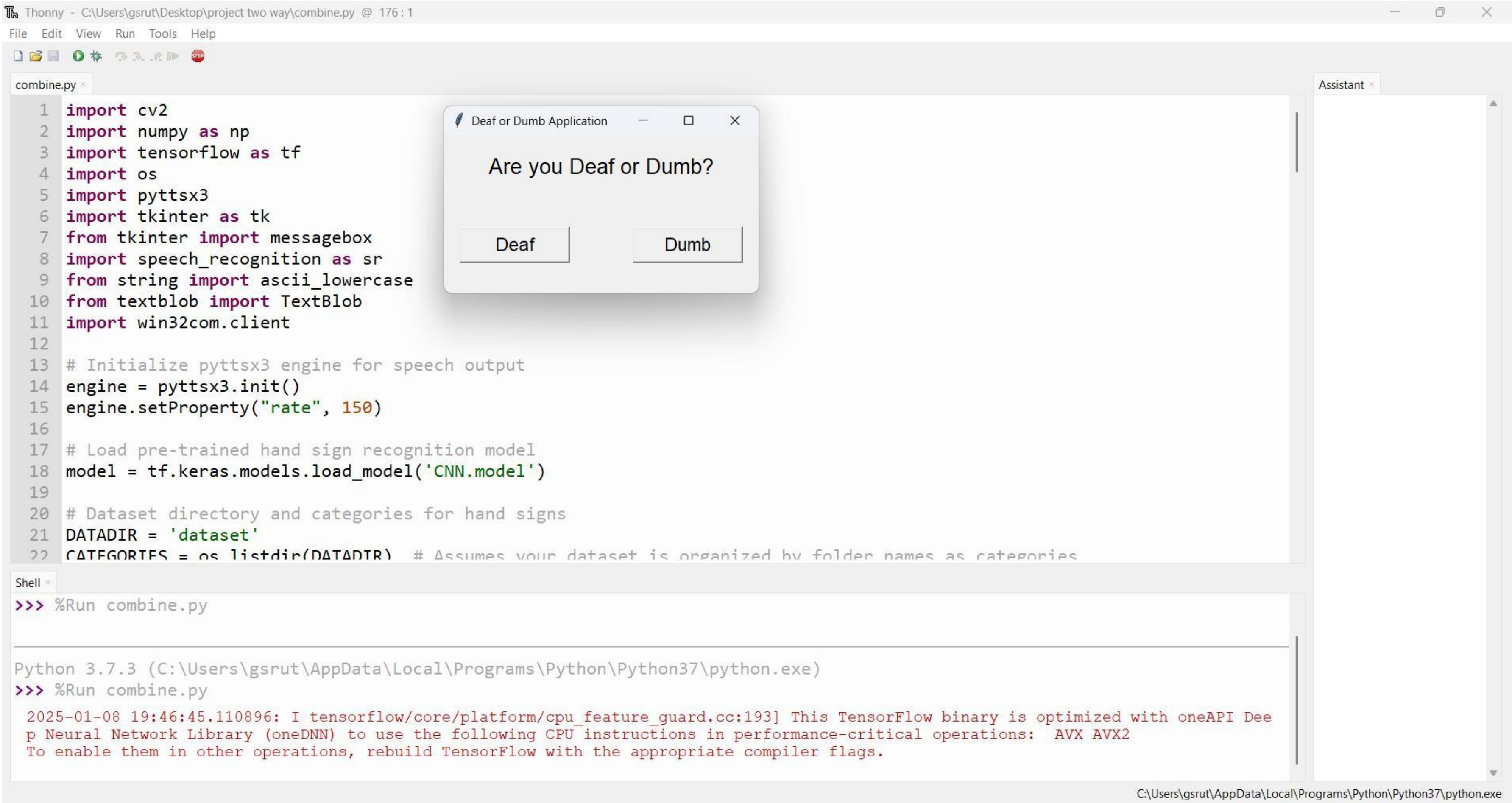
Classification Report



```python
from sklearn import metrics
acc=(metrics.accuracy_score(yt,y_test)*100)
```

```
Classification Report

              precision    recall  f1-score   support

           0       0.88      0.97      0.92       312
           1       0.83      0.99      0.90       272
           2       0.97      0.99      0.98       322
           3       0.95      0.98      0.97       327
           4       1.00      1.00      1.00       308
           5       0.98      0.98      0.98       296
           6       0.99      0.95      0.97       304
           7       1.00      0.97      0.99       276
           8       0.97      0.93      0.95       295
           9       1.00      0.98      0.99       311
          10       0.98      0.95      0.96       328
          11       0.97      0.96      0.97       281
          12       1.00      0.99      1.00       266
          13       0.99      1.00      0.99       278
          14       0.96      0.91      0.94       312
          15       0.98      1.00      0.99       316
          16       1.00      1.00      1.00       277
          17       0.96      0.93      0.95       306
          18       0.98      0.99      0.98       268
          19       1.00      1.00      1.00       296
          20       0.98      0.99      0.99       285
          21       0.82      0.93      0.87       323
          22       0.98      0.85      0.91       306
          23       0.96      0.88      0.92       320
          24       1.00      1.00      1.00       284
          25       0.99      0.96      0.98       306
          26       1.00      1.00      1.00       325

    accuracy                           0.96      8100
   macro avg       0.97      0.97      0.97      8100
weighted avg       0.97      0.96      0.97      8100

>>>
```

Deaf or Dumb Prompt

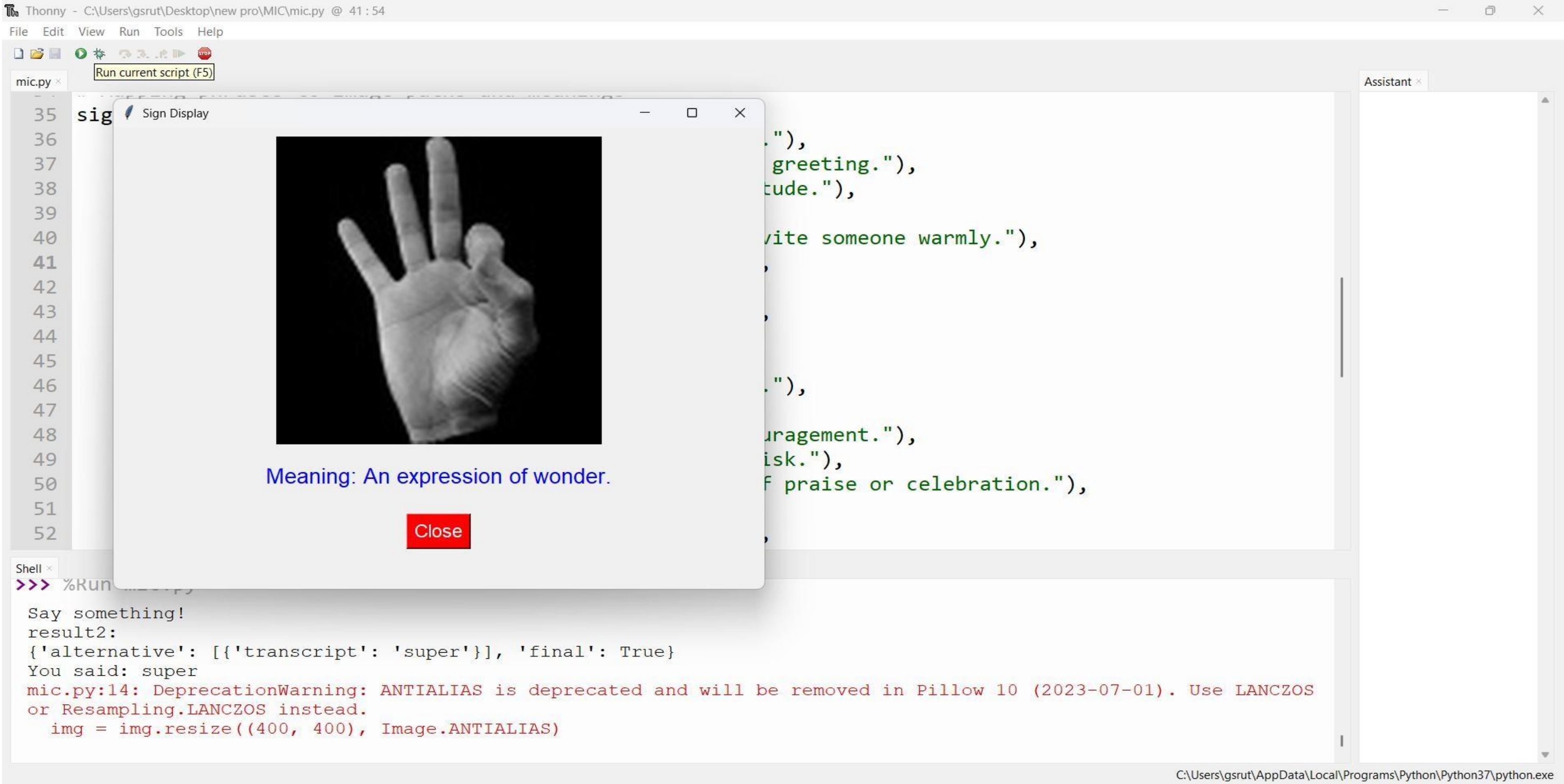


Are you Deaf or Dumb?

```python
import cv2
import numpy as np
import tensorflow as tf
import os
import pyttsx3
import tkinter as tk
from tkinter import messagebox
import speech_recognition as sr
from string import ascii_lowercase
from textblob import TextBlob
import win32com.client

# Initialize pyttsx3 engine for speech output
engine = pyttsx3.init()
engine.setProperty("rate", 150)

# Load pre-trained hand sign recognition model
model = tf.keras.models.load_model('CNN.model')

# Dataset directory and categories for hand signs
DATADIR = 'dataset'
CATEGORIES = os.listdir(DATADIR)  # Assumes your dataset is organized by folder names as categories
```

```
>>> %Run combine.py

Python 3.7.3 (C:\Users\gsrut\AppData\Local\Programs\Python\Python37\python.exe)
>>> %Run combine.py
 2025-01-08 19:46:45.110896: I tensorflow/core/platform/cpu_feature_guard.cc:193] This TensorFlow binary is optimized with oneAPI Dee
 p Neural Network Library (oneDNN) to use the following CPU instructions in performance-critical operations:  AVX AVX2
 To enable them in other operations, rebuild TensorFlow with the appropriate compiler flags.
```

Conversion of Speech to Sign



```
sig
                                                  ."),
                                                  greeting."),
                                                  tude."),

                                                  vite someone warmly."),
                                                  ,

                                                  ,


                                                  ."),

                                                  uragement."),
                                                  isk."),
                                                  f praise or celebration."),

                                                  ,
```

Meaning: An expression of wonder.

```
>>> %Run

Say something!
result2:
{'alternative': [{'transcript': 'super'}], 'final': True}
You said: super
mic.py:14: DeprecationWarning: ANTIALIAS is deprecated and will be removed in Pillow 10 (2023-07-01). Use LANCZOS
or Resampling.LANCZOS instead.
  img = img.resize((400, 400), Image.ANTIALIAS)
```

```
    close_button.pack(pady=20)

    winc

# Speecl
r = sr.I
speech =

# Mappir
signs =
    "was
    "goc                                                             ting."),
    "wai                                                             me a cup of water?"),
    "hel
    "wel                                                             someone warmly."),
    "suj
    "yes
    "App
```

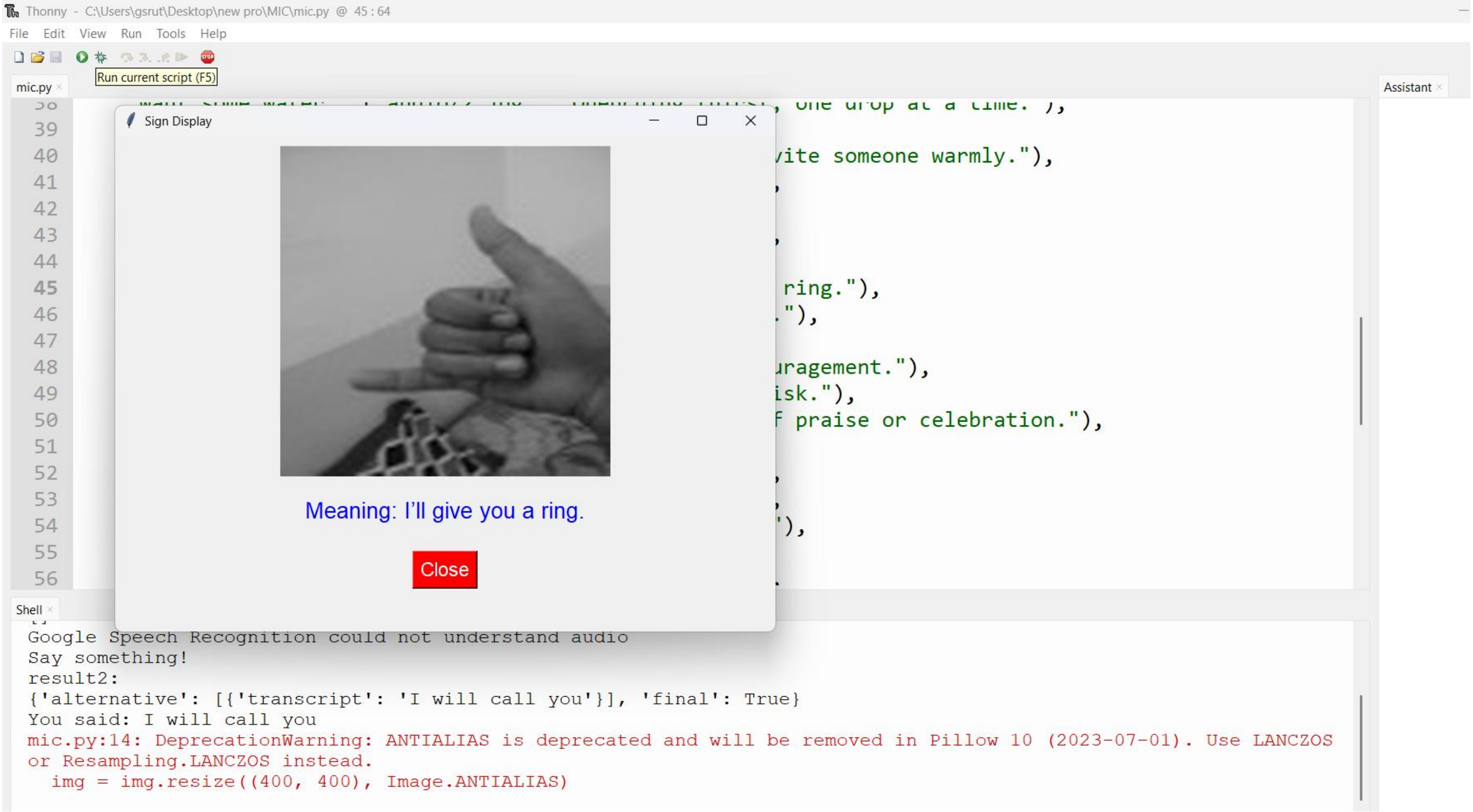
Meaning: Could you hand me a cup of water?

```
>>> %Run mic
 Say somethin
 result2:
 {'alternative': [{'transcript': 'want some cup of water'}], 'final': True}
 You said: want some cup of water
 mic.py:14: DeprecationWarning: ANTIALIAS is deprecated and will be removed in Pillow 10 (2023-07-01). Use LANCZOS
 or Resampling.LANCZOS instead.
   img = img.resize((400, 400), Image.ANTIALIAS)
```

```
    window.mainloop()

# Speech
r = sr.R
speech =

# Mappin
signs =
    "bot
    "goo
    "wan                                                  water?"),
    "i h                                                  ou clarify?"),
    "we                                                   blood, stay calm."),
    "its                                                  ed."),
    "is                                                   ize you."),
    "i w
    "ple
     "I
    "i w
    "smi                                                  ay better!."),
    "i h                                                  stitch. Better get the needle!."),
    "dow                                                  ),
    "i a                                                  drink."),
```

Meaning: I need a bottle.

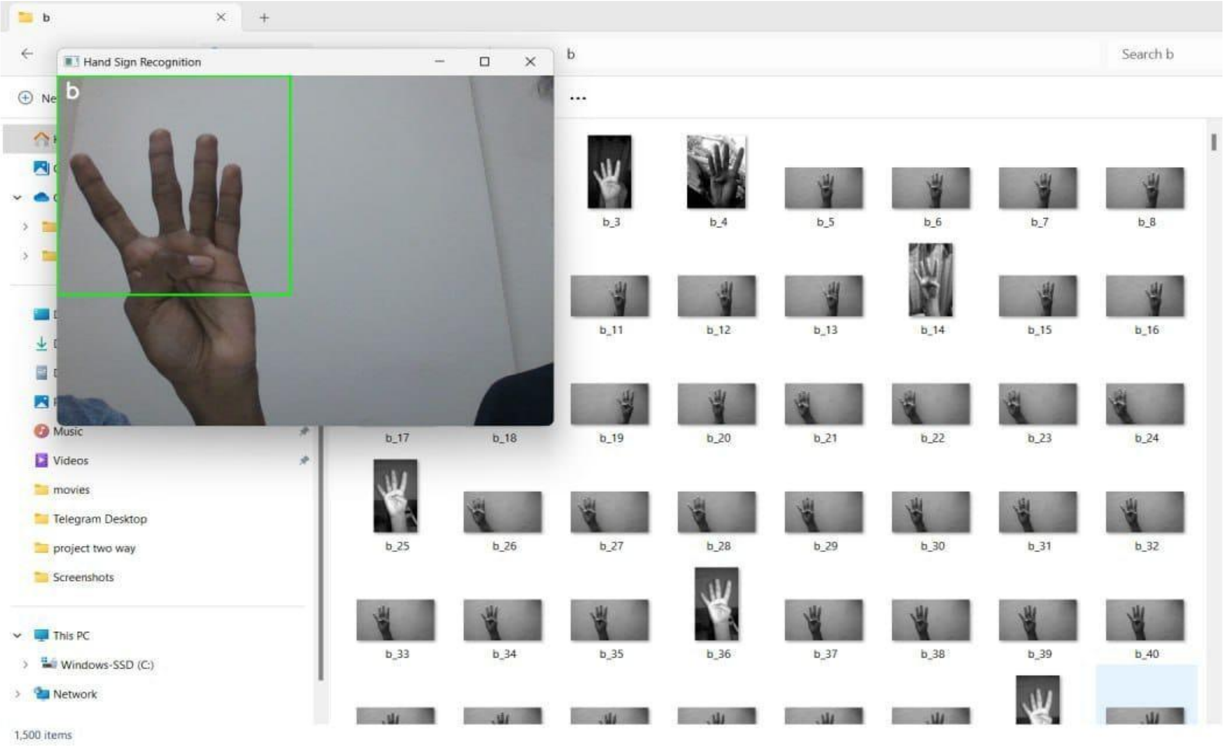
```
Say something!
result2:
{   'alternative': [{'transcript': 'bottle'}, {'transcript': 'bottom'}],
    'final': True}
You said: bottle
mic.py:14: DeprecationWarning: ANTIALIAS is deprecated and will be removed in Pillow 10 (2023-07-01). Use LANCZOS or Resampling.LANCZOS
instead.
  img = img.resize((400, 400), Image.ANTIALIAS)
```

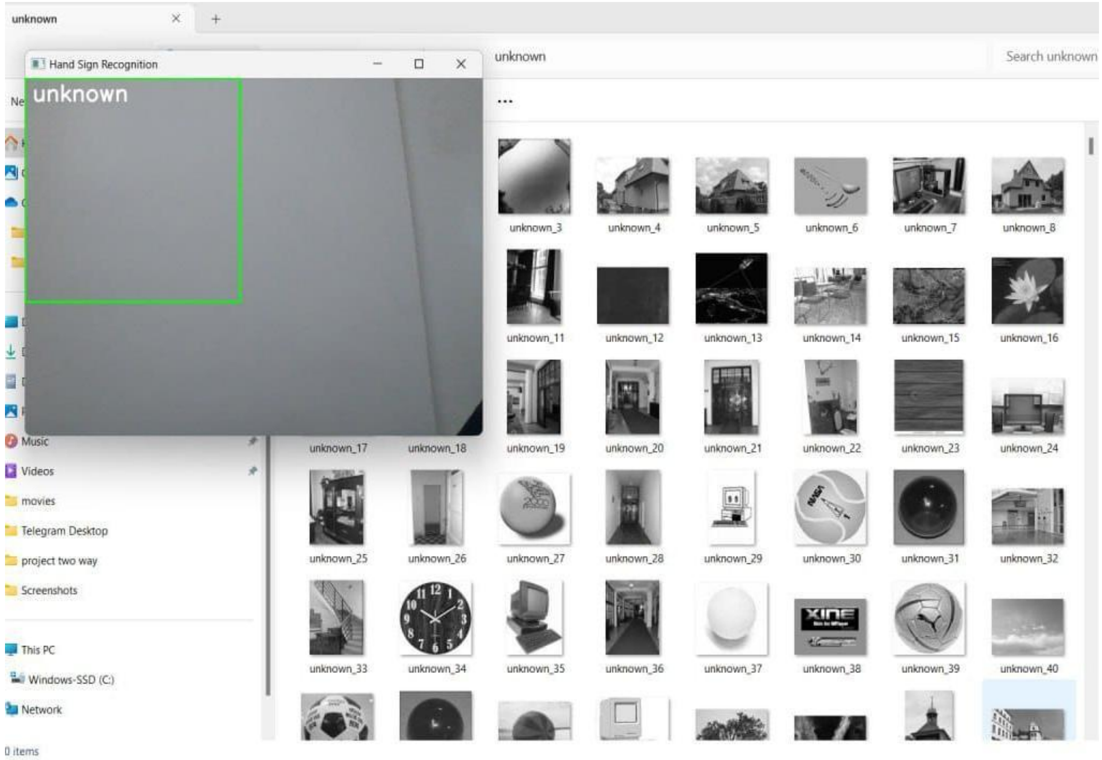


Meaning: I'll give you a ring.

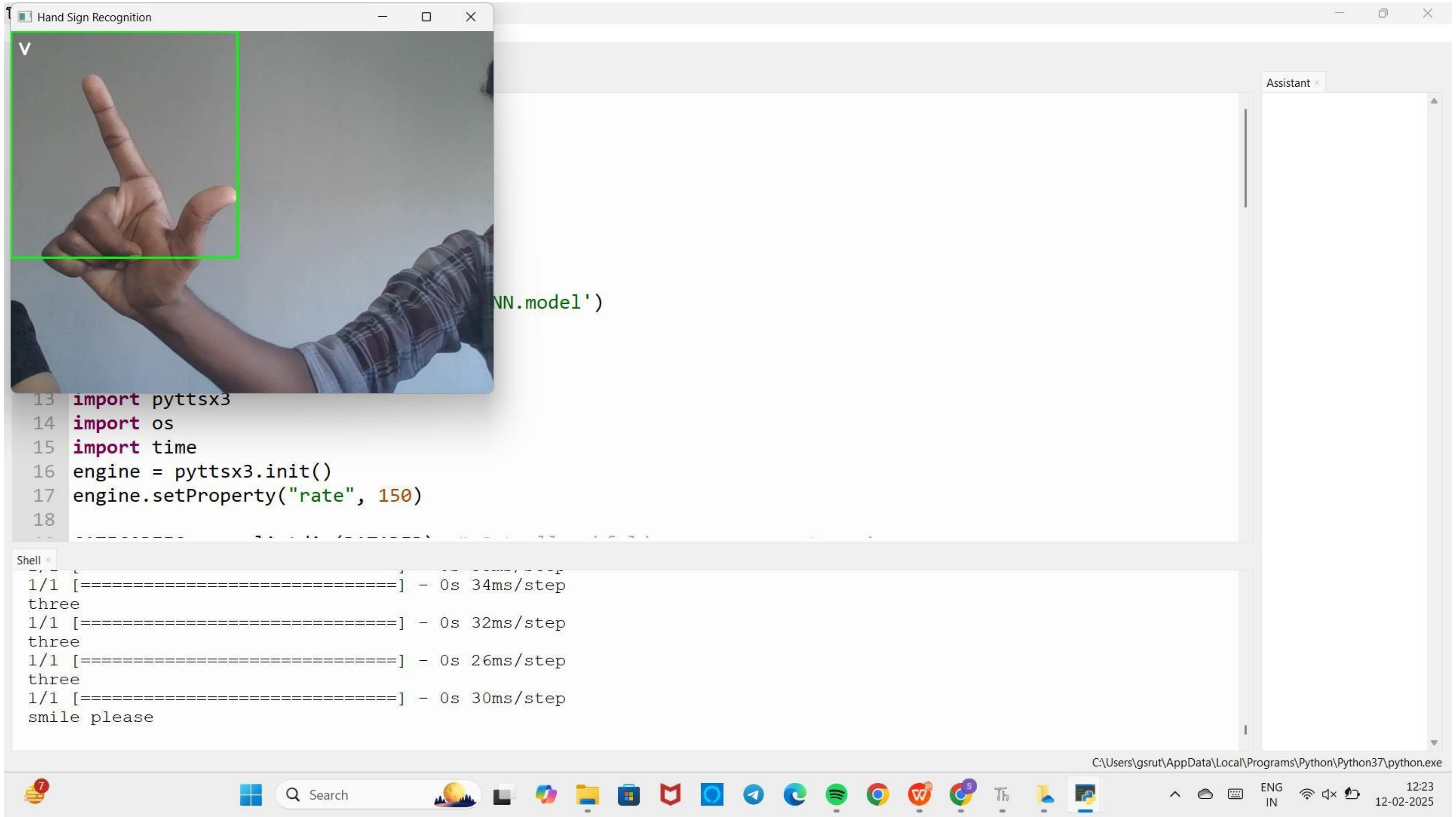
```
Google Speech Recognition could not understand audio
Say something!
result2:
{'alternative': [{'transcript': 'I will call you'}], 'final': True}
You said: I will call you
mic.py:14: DeprecationWarning: ANTIALIAS is deprecated and will be removed in Pillow 10 (2023-07-01). Use LANCZOS
or Resampling.LANCZOS instead.
  img = img.resize((400, 400), Image.ANTIALIAS)
```

Conversion of Sign to Speech





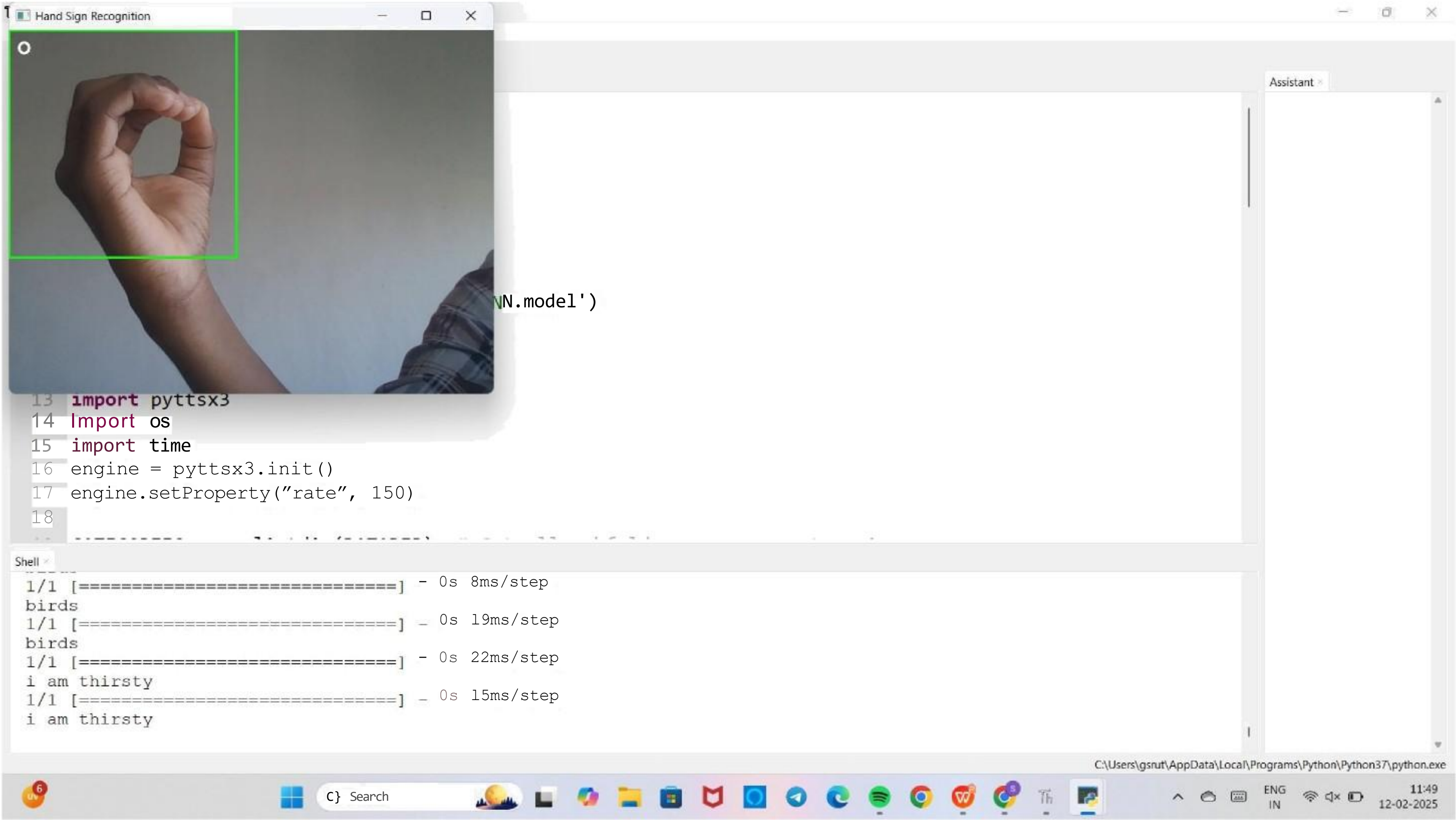
Conversion of Sign to Speech



```
NN.model')
```

```
import pyttsx3
import os
import time
engine = pyttsx3.init()
engine.setProperty("rate", 150)

```

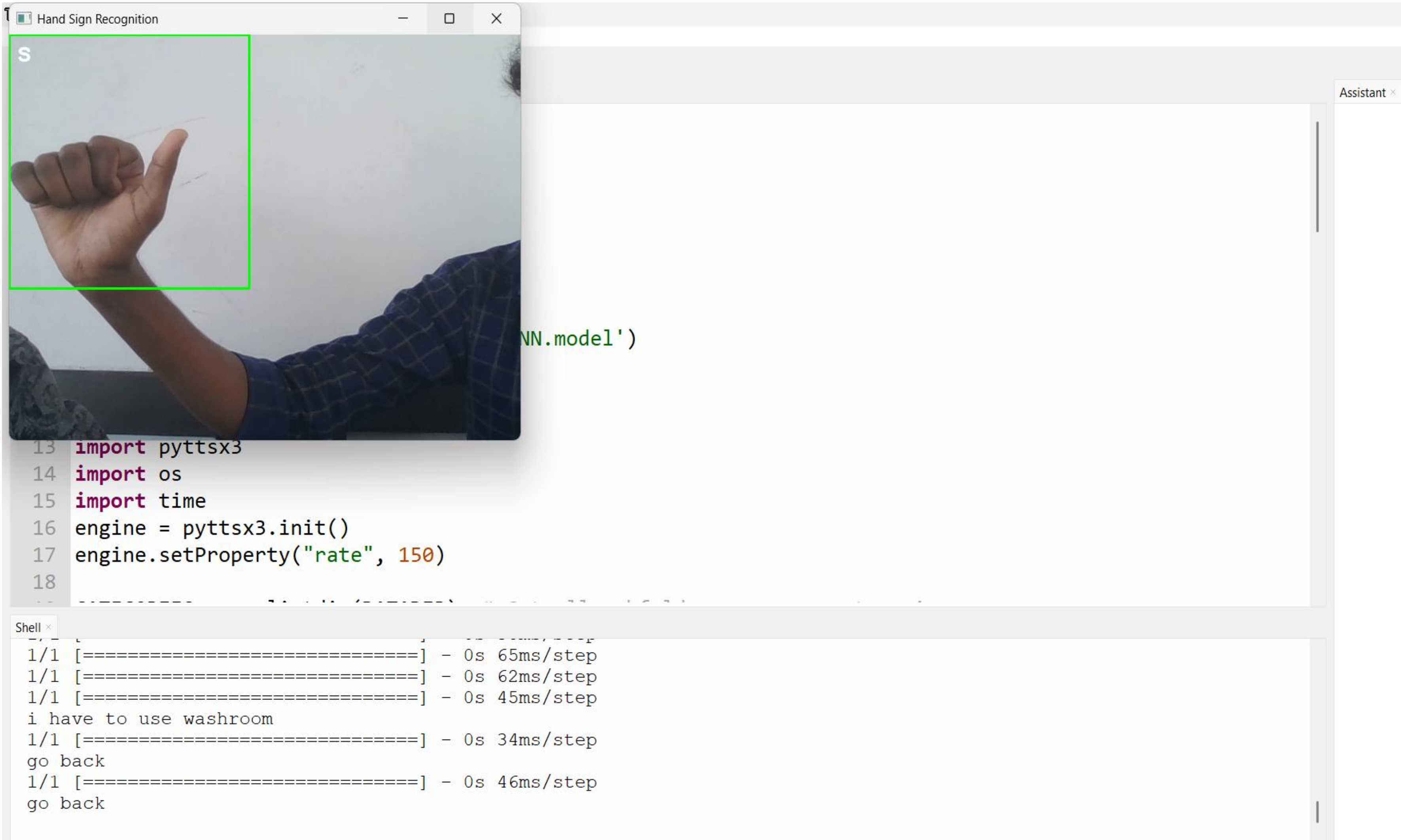
```
1/1 [==============================] - 0s 34ms/step
three
1/1 [==============================] - 0s 32ms/step
three
1/1 [==============================] - 0s 26ms/step
three
1/1 [==============================] - 0s 30ms/step
smile please
```



```
N.model')
```

```
import pyttsx3
Import os
import time
engine = pyttsx3.init()
engine.setProperty(”rate”, 150)
```

```
1/1 [==============================] - 0s 8ms/step
birds
1/1 [==============================] - 0s 19ms/step
birds
1/1 [==============================] - 0s 22ms/step
i am thirsty
1/1 [==============================] - 0s 15ms/step
i am thirsty
```



```
NN.model')
```

```
import pyttsx3
import os
import time
engine = pyttsx3.init()
engine.setProperty("rate", 150)

```

```
1/1 [==============================] - 0s 65ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 45ms/step
i have to use washroom
1/1 [==============================] - 0s 34ms/step
go back
1/1 [==============================] - 0s 46ms/step
go back
```

CONCLUSION

- The system bridges communication gaps between the deaf and hearing communities through a two-way sign language translator.
- **Sign-to-Speech Conversion** uses a modified Convolutional Neural Network (CNN) to classify hand signs, convert them into text, and generate speech using Google Text-to-Speech (GTTS).
- **Speech-to-Sign Conversion** uses PySpeech for speech recognition, transcribes speech into text, and maps it to sign gestures using OpenCV.
- The system achieves high validation accuracy with minimal loss, ensuring reliable sign-to-speech and speech-to-sign translation.
- The confusion matrix analysis confirms strong performance, with high true positives and low false detections. the classification report validates the model's effectiveness in real-time communication.

- **Results** Achieved 96% recognition accuracy and real-time processing.

REFERENCES

1. Chaudhary, L., Ananthanarayana, T., Hoq, E., & Nwogu, I. (2022). Signnet ii: A transformer-based two-way sign language translation model. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.
2. Hu, H., Pu, J., Zhou, W., & Li, H. (2022). Collaborative multilingual continuous sign language recognition: A unified framework. *IEEE Transactions on Multimedia*.
3. Hameed, H., Usman, M., Tahir, A., Ahmad, K., Hussain, A., Imran, M. A., & Abbasi, Q. H. (2022). Recognizing British Sign Language Using Deep Learning: A Contactless and Privacy-Preserving Approach. *IEEE Transactions on Computational Social Systems*.
4. Kothadiya, D., Bhatt, C., Sapariya, K., Patel, K., Gil-González, A. B., & Corchado, J. M. (2022). Deepsign: Sign language detection and recognition using deep learning. *Electronics*, *11*(11), 1780.
5. van der Merwe, A., Mambou, E. N., & Swart, T. G. (2021, September). Automated Sign Language Alphabet Detection. In *2021 IEEE AFRICON* (pp. 1-6). IEEE.
6. Breland, D. S., Skriubakken, S. B., Dayal, A., Jha, A., Yalavarthy, P. K., & Cenkeramaddi, L. R. (2021). Deep learning-based sign language digits recognition from thermal images with edge computing system. *IEEE Sensors Journal*, *21*(9), 10445-10453.

7. Al-Qurishi, M., Khalid, T., & Souissi, R. (2021). Deep learning for sign language recognition: Current techniques, benchmarks, and open issues. *IEEE Access*, *9*, 126917-126951.
8. Wadhawan, A., & Kumar, P. (2020). Deep learning-based sign language recognition system for static signs. *Neural computing and applications*, *32*, 7957-7968.
9. Rastgoo, R., Kiani, K., & Escalera, S. (2021). Sign language recognition: A deep survey. *Expert Systems with Applications*, *164*, 113794.
10. Adaloglou, N., Chatzis, T., Papastratis, I., Stergioulas, A., Papadopoulos, G. T., Zacharopoulou, V., ... & Daras, P. (2021). A comprehensive study on deep learning-based methods for sign language recognition. *IEEE Transactions on Multimedia*, *24*, 1750-1762.
11. Wang, Z., Zhao, T., Ma, J., Chen, H., Liu, K., Shao, H., ... & Ren, J. (2020). Hear sign language: A real-time end-to-end sign language recognition system. *IEEE Transactions on Mobile Computing*, *21*(7), 2398-2410.


Thank
you