ISSN: 0974-5823 Vol. 7 No. 4 April, 2022

International Journal of Mechanical Engineering

A Helping Hand for People Having Loss of Hearing & Speech Using NLP and Speech Recognition with an Interactive UI

Suraj Beera *[1]

[1] School of Computer Science & Engineering , Vellore Institute of Technology, Vellore, India.

Naren Hrithik *[2]

[2] School of Computer Science & Engineering , Vellore Institute of Technology, Vellore, India.

Amaan Ali Jaffri *[3]

[3] School of Computer Science & Engineering , Vellore Institute of Technology, Vellore, India.

Gopalakrishnan.T [4]

[4] Assistant Professor (sr.g), School of Computer Science & Engineering , Vellore Institute of Technology, Vellore, India.

Abstract - Daily activities including Communication, Navigation, etc. are certainly particularly challenging tasks for visually impaired, blind, and deaf-mute people. Indoor navigation itself is certainly becoming a harder task for deaf-mute people.

We have used speech recognition which will convert the speech to text and have used natural language processing to recognize and split the sentence into its words then the hand sign videos are extracted if the corresponding word is present in the database and if the words are not present in our database, they are broken down to their hand signs letter by letter then these videos are merged to form a complete video for the corresponding sentence.

This video is displayed to the user. Detection is done by python libraries which use voice and speech recognition to identify what the mute person is conveying. We have used Reactjs to build the frontend GUI.

Index Terms: *Pytorch,Tensor Flow,Speech Recognition,NLP.*

IMPORTANT ABBREVIATIONS

The expression 'You Only Look Once' is abbreviated as YOLO. This is an algorithm for detecting and recognising different items in a photograph (in real-time). YOLO v5 is utilised to detect objects in this case. [1]

Object detection is a computer approach that searches digital pictures and videos for instances of semantic entities of a certain class (such as people, buildings, or vehicles). It has to do with image processing and computer vision. [2]

INTRODUCTION

THIS project is based on the YOLO algorithm through which we can detect the degree to which the student/person is using mobile phone in the monitored environment in a stipulated time. With progresses in recent technologies, cell phones have filled in notoriety to become the most well-known customer device. Phones are vital piece of current life. Large numbers of us need to settle on a decision or communicate something specific at whenever from anyplace.

Our point is to work on the correspondence with individuals who has hearing hardships and utilizing any sign language to articulate their thoughts. At the principal sight, as a thought, how troublesome could make a sign dialects converter. We will catch capacities and specialized provisions to the motion catch of sign to voice Change.

In this paper we have tried to solve the communication gap between the deaf and dumb people and the people who don't know sign language which will make conversations easier. We have taken the speech of the user as input and converted that input sentence to its corresponding sign language. Our project aims to solve the above-mentioned problems.

MOTIVATION

Technical usability:

Daily activities including Communication, Navigation, etc. are certainly particularly challenging tasks for visually impaired, blind, and deaf-mute people. Indoor navigation itself is certainly becoming a harder task for deaf-mute people.

At the principal sight, as a thought, how troublesome could make a sign dialects converter. We will catch capacities and specialized provisions to the motion catch of sign to voice Change.

The communication gap between the deaf and dumb people and the people who don't know sign language which will be bridged via and user interface which make conversations easier. We have taken the speech of the user as input and converted that input sentence to its corresponding sign language.

The way of transferring the information from one person to another is called communication. Most of the time people use signs and words for the communication. Natural language is used by normal people to communicate/interact with each other while tactile sign language is used by deaf and dumb people to interact.

Factual usability in ground situation:

Nowadays people with disabilities experience difficulties to stand in the race because of ferocious competition in every field. The effort is to develop an interface which will help deaf and dumb people interact with a normal person. According to a survey, India consist of nearly 2.4 million deaf, dumb populations which approximately make up 20% of the world's total deaf and dumb population.

For hassle-free interaction between the normal person and deaf and dumb person, there is a need of an interpreter (Person who has the knowledge of sign language, as well as normal language).

Sign language is divided into two i.e. Visual Sign Language & Tactile Sign Language.

- Visual sign language: It is used by hearing & speech impaired people
- Tactile sign language: It is used by hearing & sight impaired people. We are basically working on the visual sign language used by deaf & dumb.

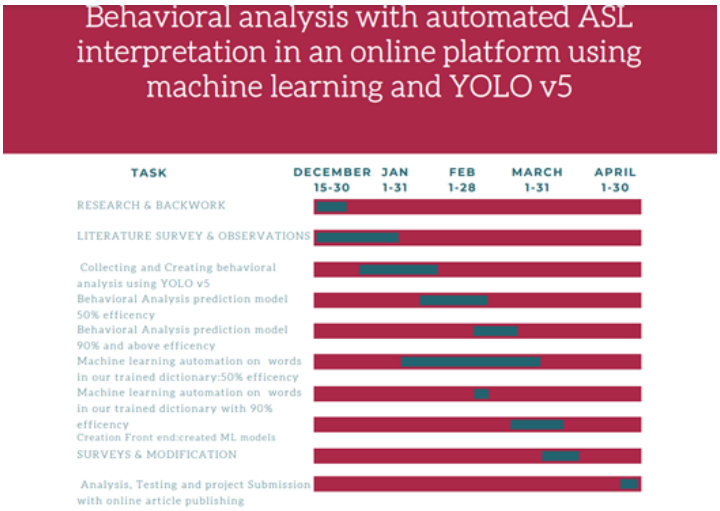
Sign Language varies country to country it depends on its culture as Sign language in India is ISL (Indian Sign Language), America uses ASL (American Sign Language), China uses CSL (Chinese Sign Language).

Sign Language is a method of communication for deaf & dumb which is composed of various gestures formed by hand shapes, body orientation & facial expression. Each gesture has a meaning assigned to it.

Alphabets in sign language are composed of different hand shapes & words are composed of hand shapes with orientation. Complete visual sign language also includes facial expressions. Visual sign language is an effective means of communication for deaf & dumb

Though it is true, the hearing-impaired have to challenge communication obstacles in a mostly hearing capable society.

GANTT CHART



BACKGROUND AND RELATED WORK

1. Research on the Hand Gesture Recognition Based on Deep Learning Study [17]:

With the quick improvement of PC vision, the interest for communication among human and machine is turning out to be increasingly broad. Since hand signals can communicate advanced data, the hand motion acknowledgment is utilized in robot control, shrewd furnishings, and different viewpoints.

The paper understands the division of hand signals by building up the skin shading model and AdaBoost classifier dependent on Haar as indicated by the distinction of skin tone for hand motions, just as the denaturation of hand 5 motions with one edge of video being cut for investigation.

In such manner, the human hand is divided from the convoluted foundation, the continuous hand motion following is additionally acknowledged by Cam Shift calculation. Directly generated by picking lines from the corpus and giving it as questions.

2. SQuAD, 100,000+ Questions for Machine Comprehension of Text [14]:

For each of the four strategies, as opposed to thinking about all O(L2) ranges as competitor replies, where L is the quantity of words in the sentence, we just use traverses which are constituents in the voting demographic parse created by Stanford CoreNLP.

Disregarding accentuation and articles, we track down that 77.3% of the right replies in the improvement set are constituents. This places a successful roof on the exactness of our strategies.

During preparing, when the right reply of a model is certainly not a constituent, we utilize the briefest constituent containing the right reply as the objective.

3. A Conversational Question Answering Challenge[15]:

Answering dataset for measuring the ability of machines to participate in a question answering style conversation. A machine must understand a text passage and answer a series of questions that appear in a conversation.

4. Reading Wikipedia to Answer Open-Domain Questions :

Paragraph encoding. Dynamic Co- attention Networks. Multi-Perspective Matching. This paper proposes to handle open-area question addressing utilizing Wikipedia as the special information source: the response to any not factual question is a text range in a Wikipedia article.

This undertaking of machine perusing at scale consolidates the difficulties of tracking down the important articles with that of machine understanding of text.

Our methodology joins a hunt part dependent on bigram hashing and TF-IDF coordinating with a multi-facet intermittent neural organization model prepared to distinguish replies in Wikipedia sections.

Our investigations on numerous current QA datasets show that the two modules are exceptionally aggressive regarding existing partners and perform multiple tasks getting the hang of utilizing far off management on their blend is a successful complete framework on this difficult errand.

5. DocChat: An Information Retrieval Approach for Chatbot Engines Using UnstructuredDocuments [21]:

Thinking about a much-worked on task, short text discussion (STC) in which the reaction R is a short text and just relies upon the last client expression Q.

Past strategies for the STC task generally depend on Q-R combines and fall into two classes: Retrieval-based techniques.

This sort of techniques initially recovers the most conceivable pair from a bunch of existing Q-R sets, which best matches current expression Q dependent on semantic coordinating with models, then, at that point, accept Rˆ as the reaction R. One 6 disservice of such a technique is that, for some, particular spaces, gathering such QR sets is obstinate. Age based strategies.

This sort of techniques ordinarily utilizes an encoderdecoder structure which initially encode Q as a vector portrayal, then, at that point, feed this portrayal to decoder to produce reaction R. Like recovery based strategies, such methodologies likewise rely upon existing Q-R sets as preparing information.

Like other language age assignments, like machine interpretation and summarizing, the familiarity and naturality of machine produced text is another downside.

To defeat the issues referenced above, we present an original reaction recovery approach, DocChat, to observe reactions dependent on unstructured records.

For every client expression, rather than searching for the best Q-R pair or creating a word succession dependent on language age procedures, our technique chooses a sentence from given reports straightforwardly, by positioning all sentences dependent on highlights planned at particular degrees of granularity.

On one hand, utilizing reports rather than Q-R sets work on the adjust.

OVERVIEW AND WORKING DIAGRAM

Sign language is a way of communication for deaf & dumb. Different sign recognition techniques are there which are giving output in the form of word for recognized sign.

The proposed method is focusing on interpretation of sign language in proper English sentence.

Different NLP techniques are used in addition to sign recognition. Input is given as video of sign language followed by framing & segmentation on video

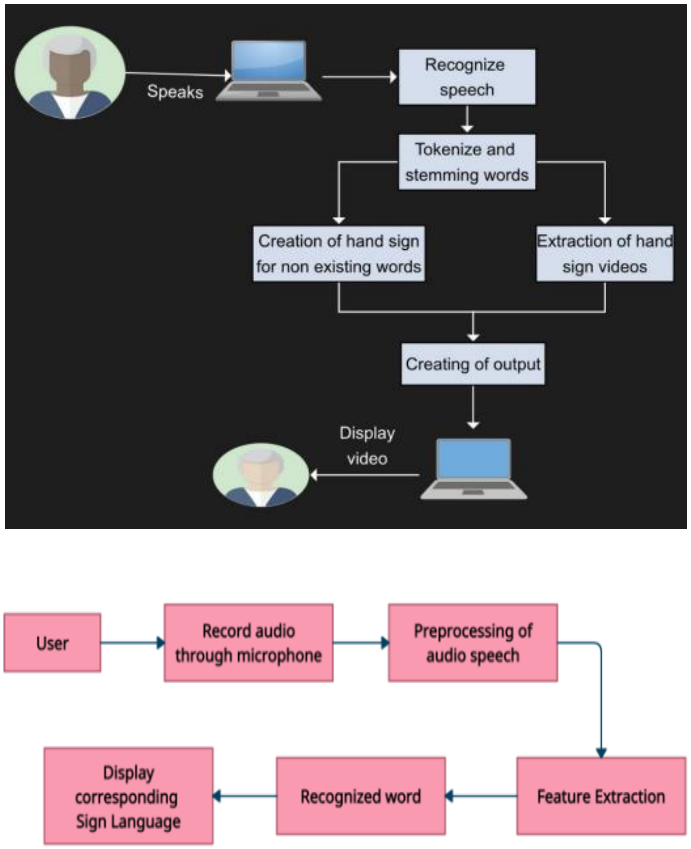


Fig A

PROPOSED ARCHITECTURE

Figure (A) represents the following procedure

1. Get the input voice from the user through the microphone.

1.1. Listen for a particular amount of time.

1.2. Listened voice gets converted into text through speech recognition libraries.

2. After listening the voice is captured and saved in the temporary memory.

3. Through the temporary memory the voice is fetched and converted to text.

3.1. To proceed with further manipulation, convert the entire string of text to lowercase letters

3.2. Then the sentence is tokenised and the words are stemmed to get the root words.

4. So, after getting the text each character of the inputted text is searched throughout the data set.

4.1. If the text is "goodbye" then the application exits, as this is the existing command.

4.2. And if the detected text is not "goodbye" then it first searches the word in the predefined dictionary images and videos.

4.3. If it is not found there, then spells the word using symbols with some delay in the image display actions.

4.4. Loops from step 2 till the speech ends

5. If an error occurs in Step 1, display "Could not listen."

Major Challenges in Sign Recognition Sign Language recognition is not an easy task, looking towards the survey there are so many difficulties. Static sign recognition is comparatively easy than continuous sign recognition. Continuous recognition includes the following problems

a) Identify the start & end of a single sign

b) Sign making speed also varies person to person.

c) The number signs in a phrase vary continuously.

d) Background of the signer & motion of signer also affects the recognition system.

METHODOLOGY

The language used for communication with deaf people is sign language. This is like their native language; in other words, it is like a mother tongue.

As deaf people cannot hear our voices to communicate with them, we use signs or symbols.

But everyone may not know sign language as it is not the necessary one to be learned. so, to overcome this problem we proposed an application that takes live or recorded voice as input and communicates with the deaf through signs or symbols.

In order, to make this application more interactive we have trained the data sets with images and small GIFs. This was implemented using Python.

For our implementation we used libraries like speech recognition, NumPy, NLTK, cv2 etc.

Now, when the application is launched, we get there two option to begin our application like live recording and all done.

All done is to close the application and the live recording is to get the input voice from the user through the microphone which is done using recognizer function which is python inbuilt library.

So store the recorded voice in temporary memory and we do text pre-processing using natural language processing techniques and if the voice is not able to be recorded we display a message called "could not listen" generally this is a user choice of message

For the text detected we perform dictionary based Machine translation that is to search the words in our trained dictionary if they were not found just spell the word using sign language which we have already trained from the alphabets A-Z.

Here to make our job easier and make the application more interactive we have added GIFs and some small video clips which are generally based on daily routines like "Hello"," Good Morning" ,"what is your name" etc.

So that for these common questions and compliments our job gets done easier and here this is the main purpose of this application

For every application there must be a termination phase also so to achieve this we made a terminating command called "goodbye" so when the application receives this command during inputting audio phase it just get terminated.

PROPOSED SYSTEM ANALYSIS AND DESIGN

Introduction

Deaf-mute is a term that was utilized generally to recognize an individual who was either deaf and utilized communication via gestures or both deaf and couldn't talk. The term keeps on being utilized to allude to deaf people who can't communicate in an oral language or have some level of talking capacity, yet decide not to talk due to the pessimistic or undesirable consideration abnormal voices now and then draw in. Such people convey utilizing sign language.

Some believe it to be a slanderous term whenever utilized external it's authentic setting; the favored term today is basically "deaf". We are replacing

Requirement Analysis

In this project we are using many open-source libraries to connect with various process inside the project. Nltk, numpy, Pillow, Jinja, pyparsing. These kinds of libraries are important in their own ways starting from speech recognition to creating output.

Product features

Speech recognition, or speech-to-text, is the capacity of a machine or program to distinguish words expressed resoundingly and convert them into clear text. Simple speech recognition programming has a restricted jargon and may possibly distinguish words and expressions when verbally expressed obviously.

More modern programming can deal with regular speech, various accents and different dialects.

Speech recognition involves an expansive exhibit of exploration in software engineering, etymology and PC designing.

Numerous advanced gadgets and text-centered programs have speech recognition capacities in them to consider simpler or without hands utilization of a gadget for Sign language we use Indiansignlanguage.org datasets which consists of more than 2000 words. These words are extremely helpful in terms of creating conversational video output.

Indiansignlanguage.org offers an immense assortment of Indian Sign Language (ISL) signs. Each sign has a picture, running video, and strung conversations. It is an optimal asset to use while you learn/show Indian Sign Language. Each sign has a picture, running video, and strung conversations.

User characteristics

As we use speech recognition for the input, it is essential for the user to know how to speak in English in terms of communication. So, store the recorded voice in temporary memory and we do text pre-processing using natural language processing techniques and if the voice is not able to be recorded, we display a message called "could not listen" generally this is a user choice of message

Assumption & Dependencies

Software like react.js and Flask are some dependencies required. User should require a computer which could be used for

Non-Functional Requirements

Using of all open-source software shows that the product is independent from any depending software. Reliability of the product is high.

Portability

Using of Java script and python as the language in this project can be used in easy deployment of any other platforms. Portabilty is comfortable.

Implementation Requirements (in terms of deployment)

1) backports.entry-points-selectable==1.1.0
2) click==8.0.1
3) cycler==0.10.0
4) distlib==0.3.3
5) filelock==3.2.0
6) Flask==2.0.1
7) Flask-Cors==3.0.10
8) itsdangerous==2.0.1
9) Jinja2==3.0.1
10) joblib==1.1.0
11) kiwisolver==1.3.2
12) MarkupSafe==2.0.1
13) matplotlib==3.4.3
14) nltk==3.6.5
15) numpy==1.21.2
16) Pillow==8.4.0
17) platformdirs==2.4.0
18) pyparsing==2.4.7
19) python-dateutil==2.8.2
20) regex==2021.10.8
21) six==1.16.0

System Requirements

H/W Requirements(details about Application Specific Hardware)

S/W Requirements(details about Application Specific Software)

- React JS
- Flask

DATASET

This whole application uses the Indian Sign Language (ISL) standard in the database. The ISL hand sign videos are downloaded from YouTube [23].

The videos used are of 360p resolution with a target output resolutionof 360p. The letter hand signs are downloaded from an open-source website for ISL hand signs [24].

This dataset is further indexed according to the unique words with the relative path. The letters are preprocessed with mean height and width also removing additional filters over the images.

1. Preprocessing:

This module helps with handling of images by resizing and applying filters to create a video

```
mean_height = 0
mean_width = 0
num_of_images = len(os.listdir('.'))
for file in os.listdir('.'):
    if file.endswith(".jpg") or file.endswith(".jpeg") or file.endswith("png"):
        im = Image.open(os.path.join(path, file))
        width, height = im.size
        mean_width += width
        mean_height += height
mean_width = int(mean_width / num_of_images)
mean_height = int(mean_height / num_of_images)
for file in os.listdir('.'):
    if file.endswith(".jpg") or file.endswith(".jpeg") or file.endswith("png"):
        im = Image.open(os.path.join(path, file))
        width, height = im.size
        print(width, height)
        imResize = im.resize((mean_width, mean_height), Image.ANTIALIAS)
        imResize.save( file, 'JPEG', quality = 95)
        print(im.filename.split('\\')[-1], " is resized")
```

```
df=pd.DataFrame()
os.chdir(path)
for file in os.listdir():
    if file.endswith("mp4"):
        n=file.split(".mp4")[0].split("-")[0]
        if n not in l1:
            l1.append(n.lower())
            l2.append(path+"/"+file)
df["word"]=l1
df["path"]=l2
os.chdir("/Users/ramkrithik/Downloads/Sign/")
df.to_csv("path.csv",index=False)
```

2. Generating Database:

This module helps to create a database of the videos existing inn the file system in form of CSV file.

```
ps=PorterStemmer()
#text=request.form["speech"]
text=word_tokenize(text)
img_array = []
for i,j in enumerate(text):
    text[i]=ps.stem(j).lower()
print(text)
x=list(df["word"])
p=list(df["path"])
l=[]
for i in text:
    if i in x:
        l.append(p[x.index(i)])
        print(l)
```

Fig 2.4 Generating Database

3. Natural language processing and results:

In this module we get the sentence and tokenize the words. Later we stem the words to get the root word to extract words from database. After the processing we iterate and check the

words in the database.

4. Processing of non existing words in database:

```
for j in i:
    filename=("/Users//Downloads/Sign/image/{}.jpg".format(j))
    img = cv2.imread(filename)
    img_array.append(img)
filename=("/Users//Downloads/Sign/image/space.jpg")
img = cv2.imread(filename)
img_array.append(img)
height,width,layers=img.shape
n="{}.mp4".format(i)
video = cv2.VideoWriter(n, cv2.VideoWriter_fourcc(*'mp4v'), 1, (width, height))
for image in img_array:
    video.write(image)
cv2.destroyAllWindows()
video.release()
```

If a word is not available in the database we use open-cv and write the signs of each letters as a video.

```
for i,j in enumerate(l):
    l[i]=VideoFileClip(l[i],target_resolution=(480, 360))
final=concatenate_videoclips(l,method='compose')
final.write_videofile("final.webm")
```

5. Export of video:

After generating the videos for the words all the video clips are combined and generated in

webm format.

```
from flask import Flask, jsonify, flash, request, redirect, url_for, render_template
from werkzeug.utils import secure_filename
import os
import shutil
from flask_cors import CORS, cross_origin
from nltk import PorterStemmer,word_tokenize
import cv2
import numpy as np
import os
import cv2
from PIL import Image

app = Flask(__name__)
cors = CORS(app)
app.config['CORS_HEADERS'] = 'Content-Type'

ALLOWED_EXTENSIONS = set(['png', 'jpg', 'jpeg'])


@app.route('/', methods=["GET"])
@cross_origin()
def main():
    return jsonify('Speech to Sign Language only API')
```

6. Flask API: Using this API we handle the request from the front-end and generate the video using the above steps.

```
@app.route('/upload', methods=['GET', 'POST'])
@cross_origin()
def upload_voice():
    ps=PorterStemmer()
    text=request.form["speech"]
    text=word_tokenize(text)
    img_array = []
    for i,j in enumerate(text):
        text[i]=ps.stem(j).lower()
    for i in text:
        for j in i:
            filename=("/Users//Downloads/Sign/image/{}.jpg".format(j))
            img = cv2.imread(filename)
            img_array.append(img)
        filename=("/Users//Downloads/Sign/image/space.jpg")
        img = cv2.imread(filename)
        img_array.append(img)
        height,width,layers=img.shape
    video = cv2.VideoWriter("test.webm", cv2.VideoWriter_fourcc(*'vp80'), 1, (width, height))
```

RESULTS

Sentence: Tom Drinks Water

Speech Recognition:

Figure (B) shows speech recognition on the front end (Graphical User Interface) of the application and post the content to the back-end.

Sign Language Generation:

Natural Language processing and output video processing

Figure (D) explains about the natural language processing for the given sentence. First step is to tokenize the words. Later stemming the words to get the root word. Then checking the existence of the root word in the database.

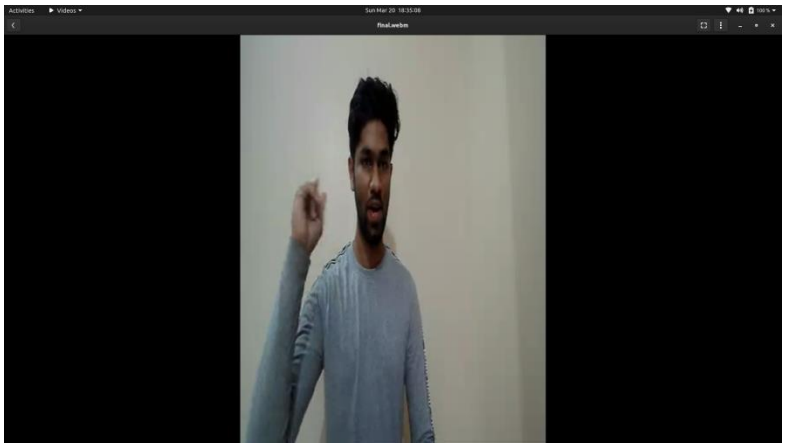
Generation of sign for non-existing words in database:

Figure (E)shows if a word is not available in the database, we use open-cv and write the hand signs of each letter as a video.

Final video:

FIGURE(F)

FINAL VIDEO:



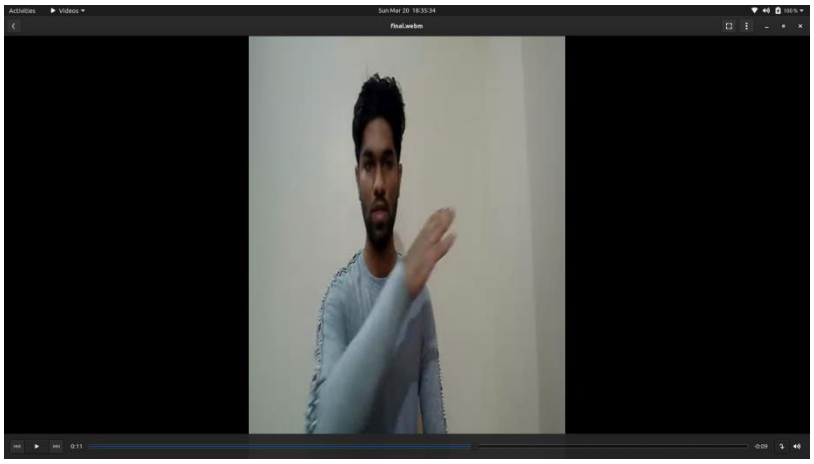


Figure (G)

Figure(H)

Figure (F,G,H) shows the final output generated combining hand signs for different root words and producing a video.

CONCLUSION

With the skills and the motivation, we had to do something for the community, we made this application which will help the deaf and mute people a way to communicate better. Users will be able to use the features of this application according to their needs.

Our helping hand is an application which conducts a conversation by means of voice recognition and hand gestures. Such projects are regularly intended to convincingly reproduce to assist people with various assignments.

This paper presents speech recognition and incorporation of gesture-based communication through python libraries with assistance and gives us hand gestures visuals.

Using React we made the application more user friendly and easy to use and the libraries in python helped us to enhance it. We have built this project from scratch so that the users can easily access our tools and use the system to its fullest capabilities

REFERENCES

[1] Bahdanau, Dzmitry, Jan Chorowski, Dmitriy Serdyuk, Philemon Brakel, and YoshuaBengio. "End-to-end attention-based large vocabulary speech recognition." In 2016 IEEE international conference on acoustics, speech and signal processing (ICASSP), pp. 4945-4949. IEEE, (2016).

[2] Chen, Danqi, Adam Fisch, Jason Weston, and Antoine Bordes. "Reading wikipedia to answer open-domain questions." arXiv preprint arXiv:1704.00051 (2017).

[3] Devlin, Jacob, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. "Bert: Pre-training of deep bidirectional transformers for language understanding." arXiv preprint arXiv:1810.04805 (2018).

[4] Hermann, Karl Moritz, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. "Teaching machines to read and comprehend." Advances in neural information processing systems 28 (2015).

[5] Karpukhin, Vladimir, BarlasOğuz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. "Dense passage retrieval for open-domain question answering." arXiv preprint arXiv:2004.04906 (2020).

[6] Kratzwald, Bernhard, and Stefan Feuerriegel. "Adaptive document retrieval for deep question answering." arXiv preprint arXiv:1808.06528 (2018).

[7] Kupiec, Julian. "MURAX: A robust linguistic approach for question answering using an on-line encyclopedia." In Proceedings of the 16th annual international ACM SIGIR conference on Research and development in information retrieval, (1993).

[8] Lan, Zhenzhong, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. "Albert: A lite bert for self-supervised learning of language representations." arXiv preprint arXiv:1909.11942 (2019).

[9] Lee, Jinhyuk, Seongjun Yun, Hyunjae Kim, Miyoung Ko, and Jaewoo Kang. "Ranking paragraphs for improving answer recall in open-domain question answering." arXiv preprint arXiv:1810.00494 (2018).

[10] Liu, Yinhan, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and VeselinStoyanov. "Roberta: A robustly optimized bert pretraining approach." arXiv preprint arXiv:1907.11692 (2019).

[11] Moldovan, Dan, SandaHarabagiu, Marius Pasca, Rada Mihalcea, Roxana Girju, Richard Goodrum, and Vasile Rus. "The structure and performance of an open-domain question answering system." In Proceedings of the 38th annual meeting of the Association for Computational Linguistics, pp. 563-570. (2000).

[12] Peters, Matthew E., Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. "Deep contextualized word representations." arXiv preprint arXiv:1802.05365 (2018).

[13] Radford, Alec, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. "Improving language understanding by generative pre-training." (2018). 15

[14] Rajpurkar, Pranav, Jian Zhang, Konstantin Lopyrev, and Percy Liang. "Squad: 100,000+ questions for machine comprehension of text." arXiv preprint arXiv:1606.05250 (2016).

[15] Reddy, Siva, Danqi Chen, and Christopher D. Manning. "Coqa: A conversational question answering challenge." Transactions of the Association for Computational Linguistics 7 (2019).

[16] Simmons, Robert F., Sheldon Klein, and Keren McConlogue. "Indexing and dependency logic for answering English questions." American Documentation 15, no. 3 (1964).

[17] Shridhar, A. "A beginner's guide to deep learning." (2017).

[18] Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and IlliaPolosukhin. "Attention is all you need." In Advances in neural information processing systems, (2017).

[19] Wang, Wenhui, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. "Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers." arXiv preprint arXiv:2002.10957 (2020).

[20] Wolf, Thomas, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, PierricCistac et al. "Huggingface's transformers: State-of-the-art natural language processing." arXiv preprint arXiv:1910.03771 (2019).

[21] Yan, Zhao, Nan Duan, Junwei Bao, Peng Chen, Ming Zhou, Zhoujun Li, and Jianshe Zhou. "Docchat: An information retrieval approach for chatbot engines using unstructured documents." In Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), (2016).

[22] Yang, Yi, Wen-tau Yih, and Christopher Meek. "Wikiqa: A challenge dataset for opendomain question answering." In Proceedings of the 2015 conference on empirical methods in natural language processing, (2015).

[23] Open source ISL video library by indiansignlanguage.org

[24] ISL Letters Hand Sign picture by talkinghands.co.i