

Aid Quest

Team Members:

Akansha Parmar	18BCE10020
Anamika Lochab	18BCE10035
Meenakshi Yadav	18BCE10159
Yashika	18BCE10302
Tenzin Gendun	18BCG10098
Ravisam Patel	18BCY10071
Tenzin Tsundue	18BCY10108

VIT Bhopal , B. Tech. (3rd Year)



Supervisor :

Dr L.Satish Kumar
Assistant Professor
VIT Bhopal

Outline

- ❑ Problem Statement
- ❑ Objective
- ❑ Existing Projects
- ❑ ASL 1.0
- ❑ ASL 2.0
- ❑ TTS
- ❑ STT
- ❑ Timeline
- ❑ Conclusion
- ❑ Individual Contribution
- ❑ References



Problem Statement

Communication is the process of exchange of thoughts and messages in various ways such as speech, signals, behavior and visuals. Deaf and dumb (D&M) people make use of their hands to express different gestures to express their ideas with other people. Gestures are the nonverbally exchanged messages and these gestures are understood with vision. This nonverbal communication of deaf and dumb people is called sign language.

Hearing loss can affect a person in various ways and may result in social withdrawal due to reduced access to services and difficulties communicating with others, emotional problems caused by a drop in self-esteem and confidence.

So the incentive behind this project is to provide access and inclusion for disabled people.

Objective

1. To translate American Sign Language (ASL) to English, for easing communication between people who are deaf and hard of hearing and the hearing world.
2. To extend the reach of digital content via TTS which gives access to a greater population, such as those with literacy difficulties, reduced vision and those learning a language.
3. To develop a voice assistant via Speech Recognition.



Existing Projects

Individually designed apps for impaired people exist, such as Be my Eyes, Voice Dream Reader, uSound, and others, but none of them combine all of the features.

Many academic articles have already been published that have aided us in completing this project but they were all written for differently abled people.

Our app is extremely cost-effective, extremely reliable with a high accuracy rate, and extremely easy to access and use. We wanted to come up with a single app that contains features that could somehow help the people with all kind of disabilities. Our software harnesses the power of technology to create a one-stop shop for a variety of issues faced by disabled people, allowing physically challenged people to receive assistance.



ASL 1.0

A functional real time vision based american sign language recognition for D&M people have been developed for ASL alphabets using CNN. The preprocessing required in a CNN is much lower as compared to other classification algorithms.

The basic steps in hand gesture recognition are -

- **Data acquisition** - To achieve a high accuracy for sign recognition in sign language recognition systems, a data set of hand sign gestures was acquired. Training set has 27455 data instances with size of 28×28 pixels with 1 channel. Test set has 7172 data instances with the same specification properties.
- **Data preprocessing** - Image Acquisition, this is the first step of pre-processing. This is the process of sensing an image. It will involve pre-processing such as scaling. In image acquisition the image will be taken from the database.
- **Feature extraction** - The reduction of data dimensionality by encoding related information in a compressed representation and removing less discriminative data is called the Feature extraction Technique. Feature extraction is vital to gesture recognition performance.
- **Gesture classification**

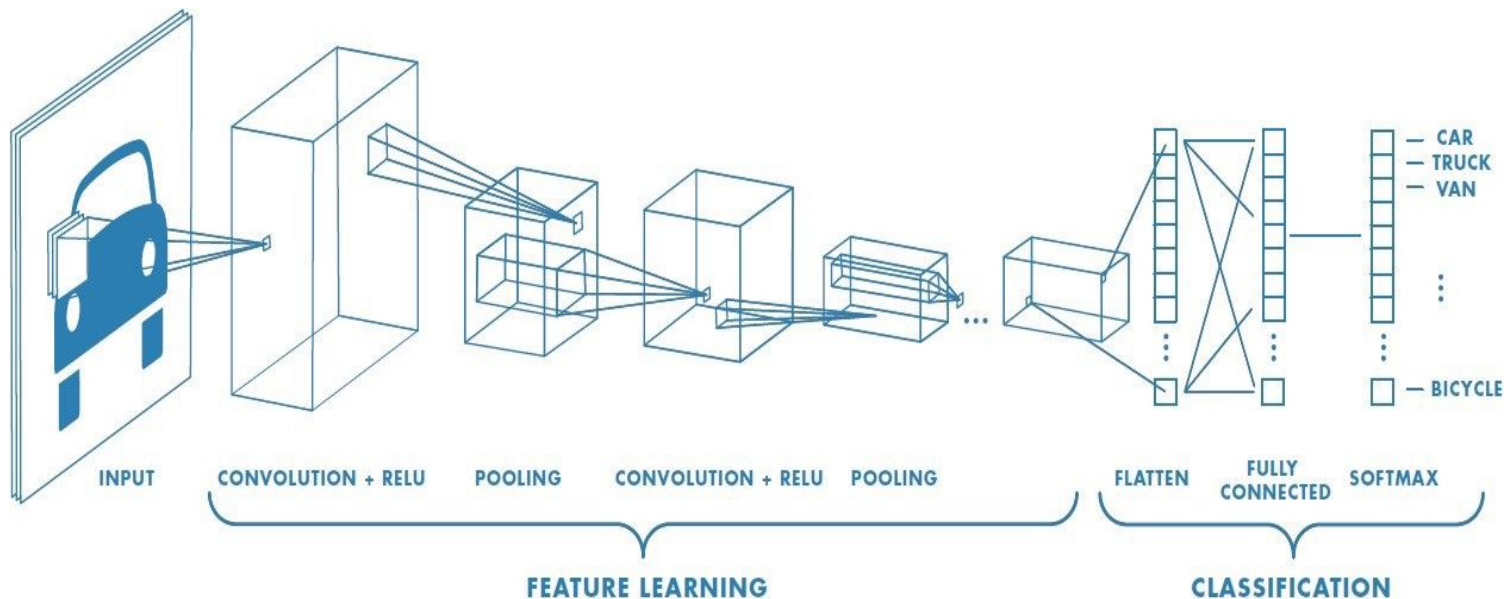
Classification with Convolutional Neural Network



Unlike regular Neural Networks , in the layers of CNN, the neurons are arranged in 3 dimensions : width, height, depth.

- **Convolution Layer** - In convolutional layer we take a small window size that extends to the depth of the input matrix. The layer consists of learnable filters of window size. During every iteration we slid the window by stride size, and computed the dot product of filter entries and input values at a given position. As we continue this process we will create a 2-Dimensional activation matrix that gives the response of that matrix at every spatial position.
- **Pooling Layer** - We use pooling layer to decrease the size of activation matrix and ultimately reduce the learnable parameters. There are two type of pooling:
 - Max Pooling - In max pooling we take a window size, and only take the maximum of 4 values.
 - Average Pooling - In average pooling we take the average of all values in a window.
- **Dense layer** - Dense layer is the regular deeply connected neural network layer. It is the most common and frequently used layer. A Dense layer feeds all outputs from the previous layer to all its neurons, each neuron providing one output to the next layer. It's the most basic layer in neural networks.

System Architecture



Working Principle



After completing Data collection and processing we move to classification process. The CNN Model -

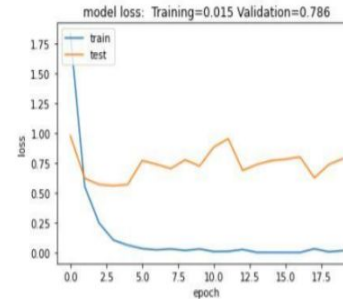
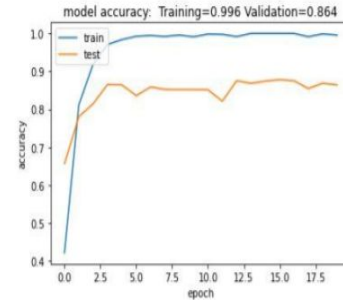
1. **1st Convolution Layer** - The input picture has a resolution of 28x28 pixels. It is first processed in the first convolutional layer using 32 filter weights and feature matrix of 7*7 with relu as activation function.
2. **1st Pooling Layer** - The pictures are downsampled using max pooling of 2x2 i.e we keep the highest value in 2x2 square array.
3. **2nd Convolution Layer** - Now, the output of the first pooling layer is served as an input to the second convolutional layer. It is processed in the second convolutional layer using 32 filter weights.
4. **2nd Pooling Layer** - The pictures are again downsampled using max pooling of 2x2 i.e we keep the highest value in 2x2 square array.
5. **1st Densely Connected Layer** - Now these images are used as an input to a fully connected layer with 128 neurons and the output from the second convolutional layer is reshaped to an array. The input to this layer is an array of values. The output of these layer is fed to the 2nd Densely Connected Layer. We are using a dropout layer of value 0.5 to avoid overfitting
6. **2nd Densely Connected Layer** - Now the output from 1st Densely Connected Layer is used with activation function relu and sigmoid.

Existing Results

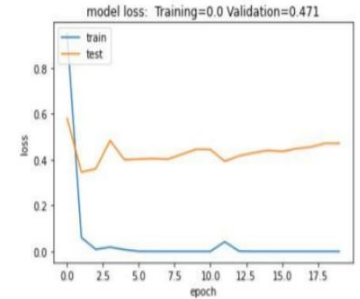
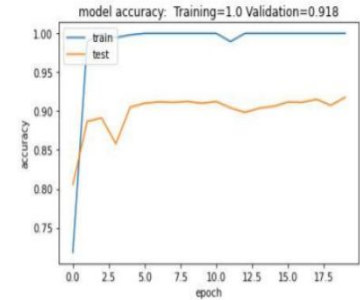
We have achieved an accuracy of 0.996 for Training and 0.864 for Testing using a 3 layer model, the result is quite satisfactory but with the number of datasets we have, we were expecting a lot more accuracy.

While the accuracy with 2 layer model is 1 for Training which clearly seems the model is overfitting but the accuracy for Test is coming as 0.918 which is significantly good.

So with a simpler model with better accuracy we chose to go with a 2-layered model.



Graph for 3 layers



Graph for 2 layers

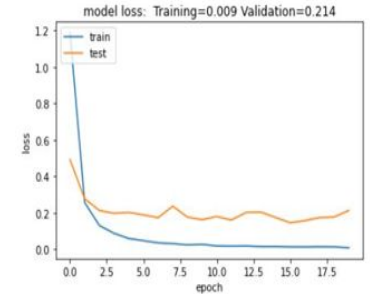
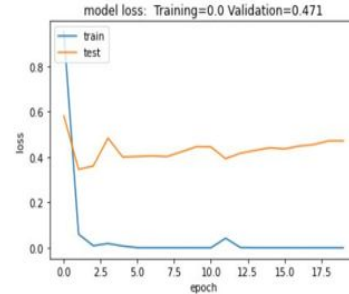
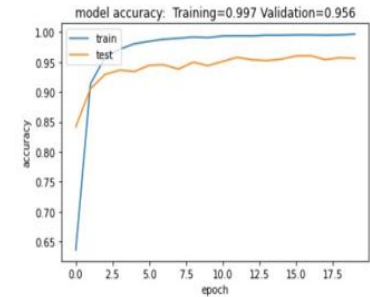
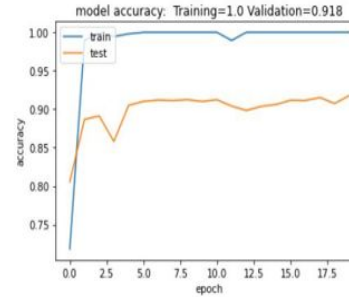
Existing Results

Since for 2 layers the model the accuracy is high, the model is clearly overfitting so we chose to use dropout to reduce the model overfitting . From the graph, we can see that with just using a dropout(default=0.5) the accuracy for the validation increases significantly and loss for testing also decreases.

Further we changed filter size and dropout rates.

Accuracy and Loss for Training and Test set for 2 Layers CNN Model with .5 dropout				
Feature size	Training Accuracy	Test Accuracy	Training Loss	Test Loss
3	0.997	0.956	0.009	0.214
5	0.985	0.961	0.045	0.188
6	0.942	0.938	0.167	0.22
7	0.974	0.968	0.077	0.089
8	0.796	0.69	0.609	0.993
9	0.856	0.767	0.42	0.682

Accuracy and Loss for Training and Test set for 2 Layers CNN Model with filter size 7				
Dropout	Training Accuracy	Test Accuracy	Training Loss	Test Loss
0.4	0.981	0.959	0.057	0.202
0.5	0.974	0.968	0.077	0.089
0.6	0.916	0.954	0.245	0.154
0.8	0.784	0.883	0.629	0.301



2 Layers without dropout

2 layers with dropout

Existing Results

The final needed model for hand signature classification and accuracy are as follows-

Model: "sequential"

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 22, 22, 32)	1600
max_pooling2d (MaxPooling2D)	(None, 11, 11, 32)	0
conv2d_1 (Conv2D)	(None, 5, 5, 32)	50208
max_pooling2d_1 (MaxPooling2D)	(None, 2, 2, 32)	0
flatten (Flatten)	(None, 128)	0
dropout (Dropout)	(None, 128)	0
dense (Dense)	(None, 128)	16512
dense_1 (Dense)	(None, 25)	3225
Total params: 71,545		
Trainable params: 71,545		
Non-trainable params: 0		

Fig: Model summary

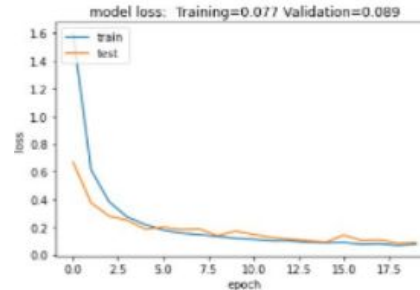
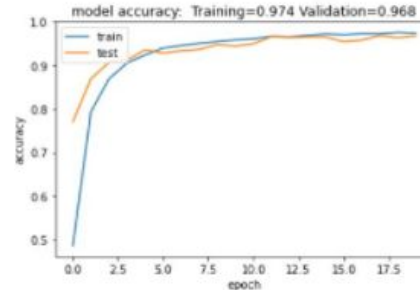


Fig: Accuracy and loss Graph

	precision	recall	f1-score	support
0	0.82	1.00	0.90	331
1	0.98	1.00	0.99	432
2	0.67	1.00	0.80	310
3	0.99	0.93	0.96	245
4	0.85	0.97	0.91	498
5	0.94	1.00	0.97	247
6	0.82	0.91	0.86	348
7	0.92	0.83	0.87	436
8	0.95	0.98	0.97	288
10	1.00	1.00	1.00	331
11	0.91	1.00	0.95	209
12	0.91	0.87	0.89	394
13	1.00	0.76	0.86	291
14	1.00	0.42	0.59	246
15	0.99	0.95	0.97	347
16	1.00	0.62	0.76	164
17	1.00	0.85	0.92	144
18	0.89	0.91	0.90	246
19	0.87	0.89	0.88	248
20	0.99	0.98	0.99	266
21	0.92	1.00	0.96	346
22	1.00	0.86	0.92	206
23	0.92	0.91	0.92	267
24	1.00	0.94	0.97	332
accuracy			0.91	7172
macro avg	0.93	0.90	0.90	7172
weighted avg	0.92	0.91	0.91	7172

Fig: Result Per classes

(Total 25 category excluding J & Z)



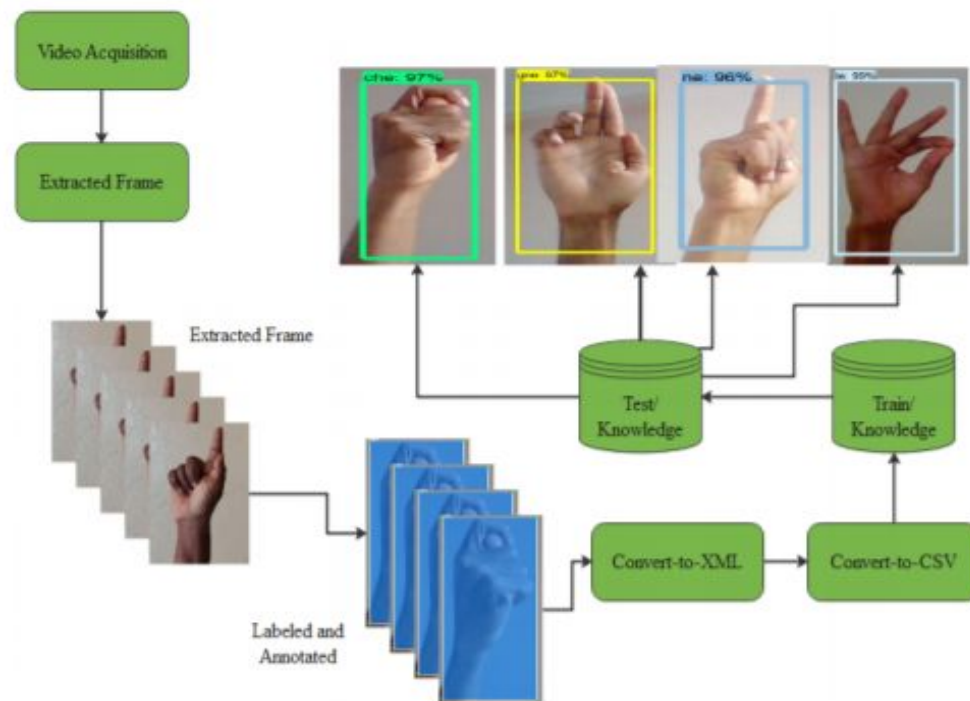
ASL 2.0

Recognizing sign language gestures from continuous gestures is a very challenging research issue. Key frames are helpful in splitting continuous sign language gestures into sequence of signs as well as for removing uninformative frames. After splitting of these gestures each sign is treated as an isolated gesture.

In this module, instead of training individual alphabets, we wanted to train and test the common sentences/words that are generally used while communicating such as yes, no, thank you etc.

Here, we first capture the images for training using webcam. The captured images will work as our dataset. After collecting the images, we labelled each image stating whether it is yes or no or thank you and so on, with the help of labelling package. After labelling, we trained our data to recognize these hand gestures by detecting the hand movements using tensorflow object detection. We transfer learning using 2-layer SSD MobileNet architecture.

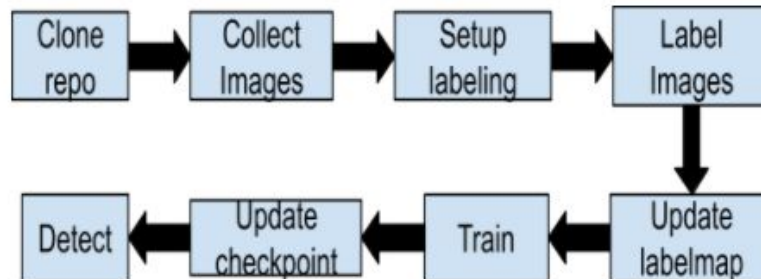
System Architecture



Working Principle

We first capture the images for training using webcam. The captured image will work as our dataset. After collecting the images, we labelled each image with the help of labellmg package. After labelling, we trained our data to recognize these hand gestures by detecting the hand movements using tensorflow object detection. We use SSD MobileNet CNN architecture. Real Time Sign Language Detection architecture includes :

- Labelling images for object detection
- Training Tensorflow for Sign Language
- Detecting Sign Language in Real time

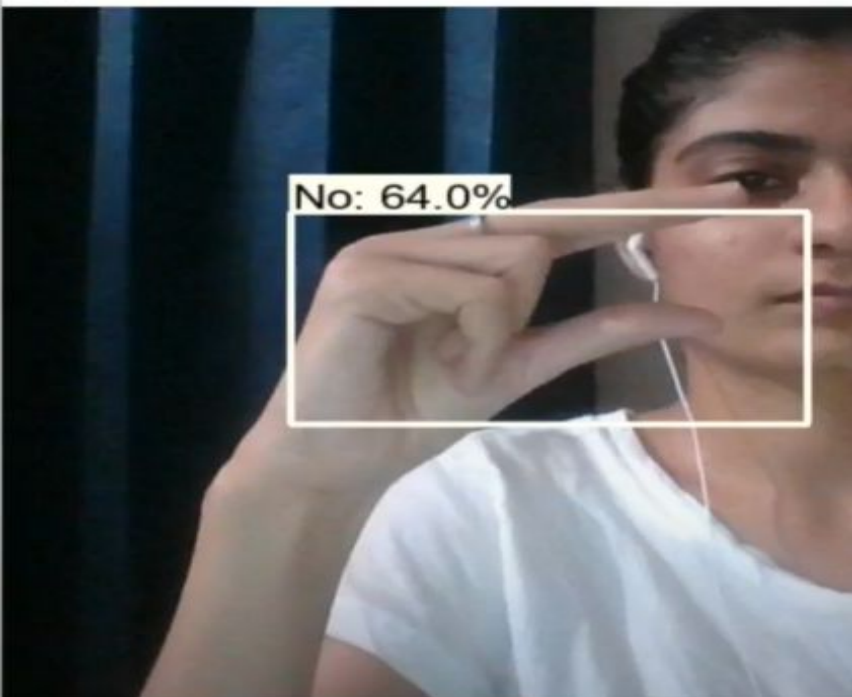


Working Principle



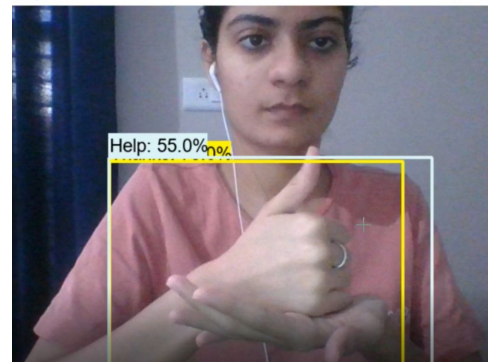
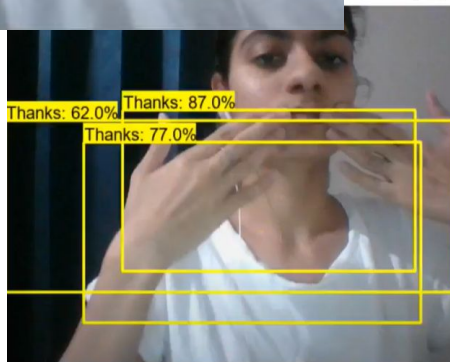
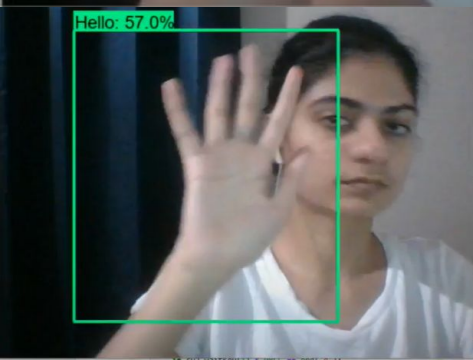
The steps involved are -

1. **Image Collection** - Capturing the images for training via video capture. The captured image will work as our dataset.
2. **Annotating images and serializing the dataset**- Annotation of the dataset is done using labellmg. Each image annotation will be saved to an individual XML file with the name of the original image file and the .xml extension. The XML files generated by labellmg is converted into a single CSV file. This is then split into two for training and testing. A "label map" is created for the classes. Each of the CSV files is changed into two TFRecord files, which is a serialized data format that TensorFlow is most familiar with.
3. **Choosing a neural network and preparing the training pipeline** - A neural network is used for classification, the training pipeline is then described in a .config file.
4. **Training the network** - The model is trained locally with 8000 training steps and training checkpoints and events are written into binary files interleaving both training and testing.
5. **Real time detection** - The module is ready for real time detection which is done using a webcam and OpenCV.



```

Command Prompt
INFO:tensorflow:Step 6700 per-step time 1.441s loss=0.228
[0524 10:17:53.429407 11804 model_lib_v2.py:683] Step 6700 per-step time 1.441s loss=0.228
INFO:tensorflow:Step 6800 per-step time 1.443s loss=0.188
[0524 10:20:17.737463 11804 model_lib_v2.py:683] Step 6800 per-step time 1.443s loss=0.188
INFO:tensorflow:Step 6900 per-step time 1.439s loss=0.181
[0524 10:22:41.644182 11804 model_lib_v2.py:683] Step 6900 per-step time 1.439s loss=0.181
INFO:tensorflow:Step 7000 per-step time 1.456s loss=0.162
[0524 10:25:07.279851 11804 model_lib_v2.py:683] Step 7000 per-step time 1.456s loss=0.162
INFO:tensorflow:Step 7100 per-step time 1.448s loss=0.139
[0524 10:27:32.085698 11804 model_lib_v2.py:683] Step 7100 per-step time 1.448s loss=0.139
INFO:tensorflow:Step 7200 per-step time 1.445s loss=0.181
[0524 10:29:56.619796 11804 model_lib_v2.py:683] Step 7200 per-step time 1.445s loss=0.181
INFO:tensorflow:Step 7300 per-step time 1.448s loss=0.227
[0524 10:32:21.409502 11804 model_lib_v2.py:683] Step 7300 per-step time 1.448s loss=0.227
INFO:tensorflow:Step 7400 per-step time 1.445s loss=0.208
[0524 10:34:45.952367 11804 model_lib_v2.py:683] Step 7400 per-step time 1.445s loss=0.208
INFO:tensorflow:Step 7500 per-step time 1.446s loss=0.174
[0524 10:37:10.587876 11804 model_lib_v2.py:683] Step 7500 per-step time 1.446s loss=0.174
INFO:tensorflow:Step 7600 per-step time 1.476s loss=0.185
[0524 10:39:38.138224 11804 model_lib_v2.py:683] Step 7600 per-step time 1.476s loss=0.185
INFO:tensorflow:Step 7700 per-step time 1.445s loss=0.161
[0524 10:42:02.621287 11804 model_lib_v2.py:683] Step 7700 per-step time 1.445s loss=0.161
INFO:tensorflow:Step 7800 per-step time 1.569s loss=0.136
[0524 10:44:39.504991 11804 model_lib_v2.py:683] Step 7800 per-step time 1.569s loss=0.136
INFO:tensorflow:Step 7900 per-step time 1.455s loss=0.150
[0524 10:47:04.976467 11804 model_lib_v2.py:683] Step 7900 per-step time 1.455s loss=0.150
INFO:tensorflow:Step 8000 per-step time 1.440s loss=0.150
[0524 10:49:28.985733 11804 model_lib_v2.py:683] Step 8000 per-step time 1.440s loss=0.150
E:\SignLanguageDetection>
    
```





Text to speech

Digital speech processing plays a vital role in modern speech communication research and applications. The fundamental purpose of speech is communication; it means transmission of messages between human and machine.

Text to speech system (TTS) converts text into voice using a speech synthesizer. It is the artificial production of human speech.

TTS System's main phases

1. Text processing
2. Speech generation

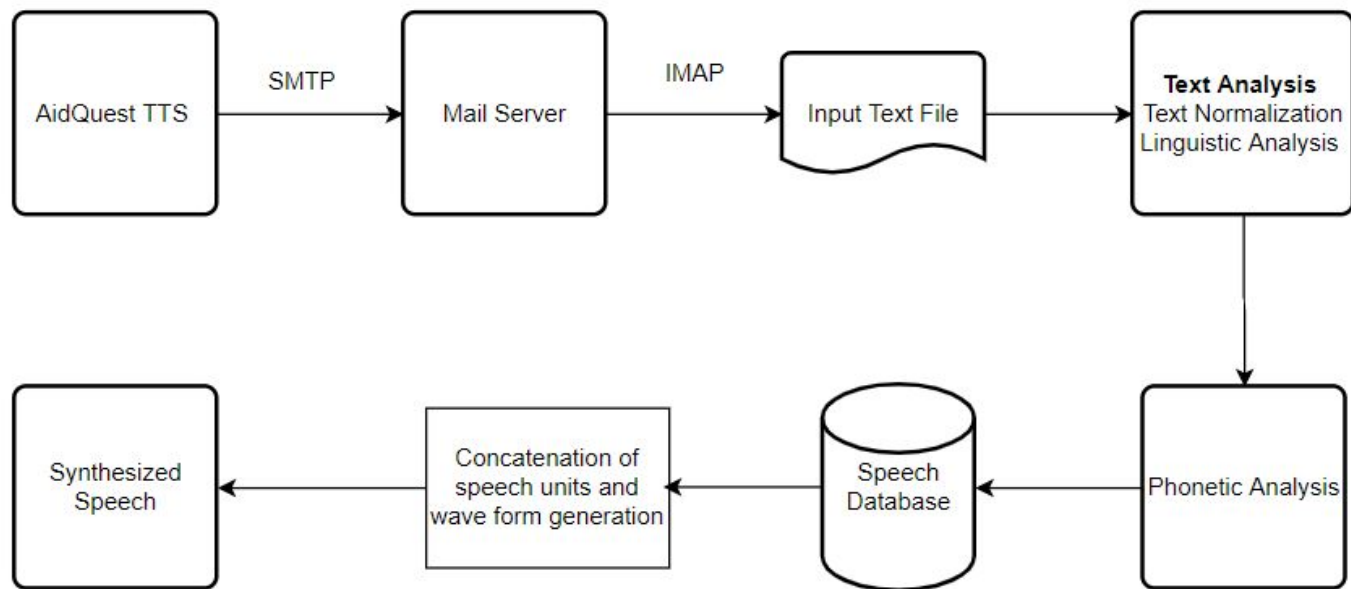
Text processing -

A text-to-speech system the input text is first analyzed, normalized and transcribed into a phonetic or some other linguistic representation. This includes Text normalization, Linguistic analysis.

Speech generation -

The speech generation component processes to generate the speech by using parameters as Phonetic analysis, Prosodic analysis.

System Architecture



Existing Result



We expect the module to access the unseen emails, get the sender, title and body data into a file, which gets converted to audio and finally the audio can be heard by the user.

Easyimap module is used to access user's email account by setting up a server which then reads the data and stores in a text file on the system, this text file data is then normalized and linguistically analysed, then using IBM's Watson speech synthesizer we read the text data and convert it into an audio which is further played using python's playsound module.



Speech to Text

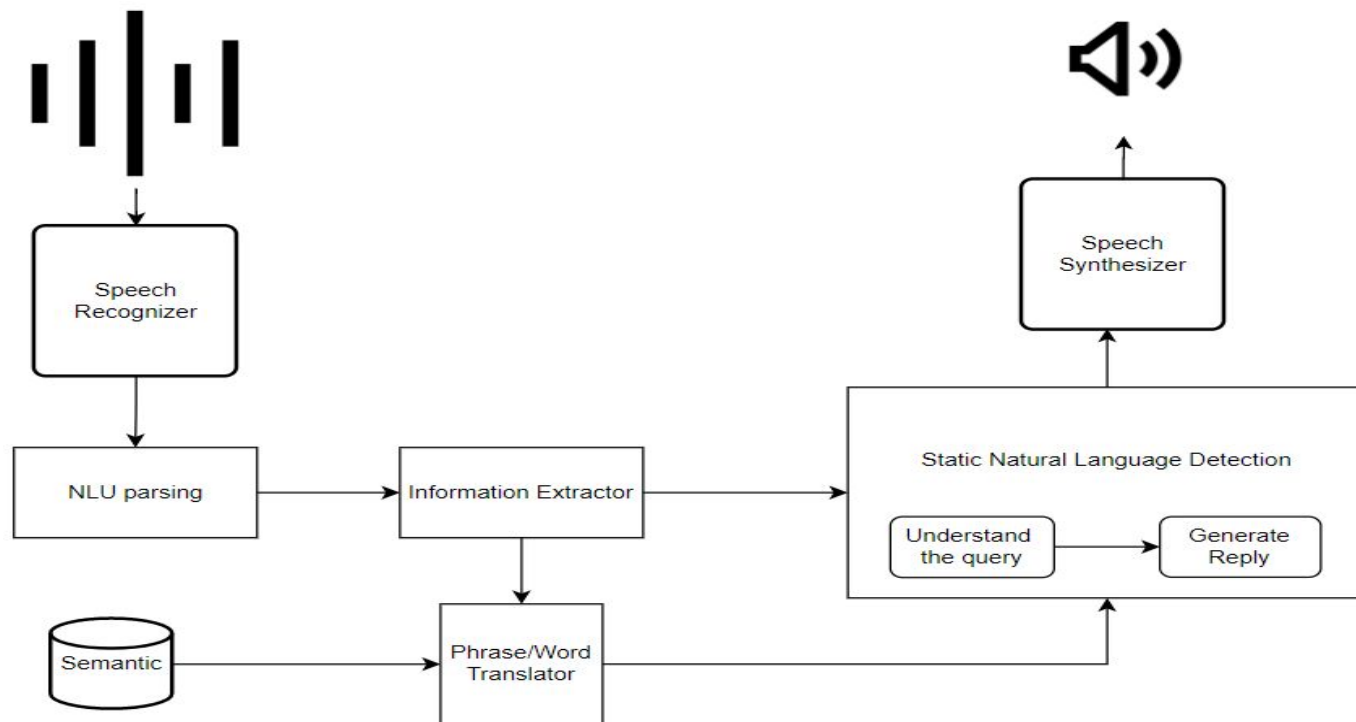
We need to develop a Personal Assistant having brilliant powers of deduction and the ability to interact with the surroundings just by one of the materialistic form of human interaction i.e. Human Voice.

The Hardware device captures the audio request through the microphone and processes the request so that the device can respond to the individual using the in-built speaker module. For Example, if you ask the device “what’s the weather?” or “how’s traffic?” using its built-in skills, it looks up the weather and traffic status respectively and then returns the response to the customer through a connected speaker.

It is named as Personal Assistant with Voice Recognition Intelligence, which takes the user input in the form of voice and processes it and returns the output in various forms like action to be performed or the search result is dictated to the end user.

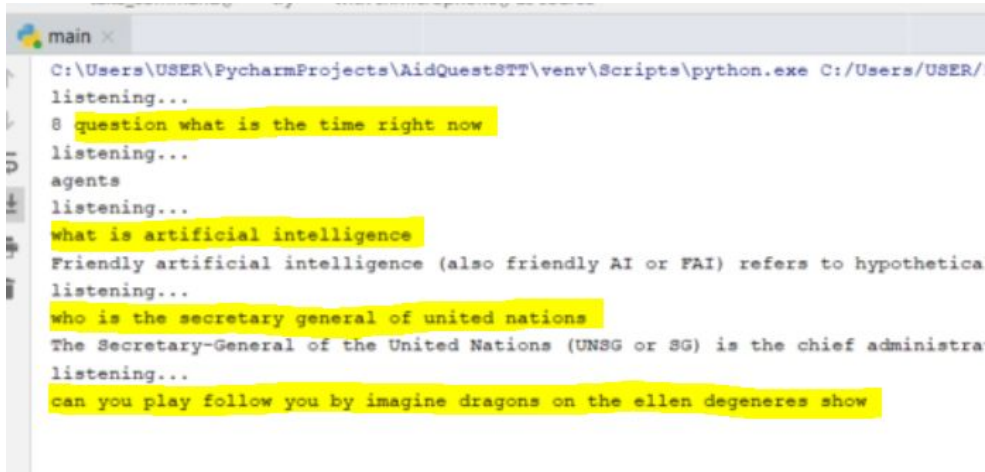
The whole project is put in action through a python script which includes online Speech to Text conversion and Text to Speech conversion

System Architecture



Existing Result

Streaming speech to text in real-time, the API is capable of processing real-time audio signals from the device microphone as input and converting it into text and resulting outputs also. We expect the module to convert speech to text using speech recognition and use that query string to open the url.



```
main x
C:\Users\USER\PycharmProjects\AidQuestSTT\venv\Scripts\python.exe C:/Users/USER/
listening...
8 question what is the time right now
listening...
agents
listening...
what is artificial intelligence
Friendly artificial intelligence (also friendly AI or FAI) refers to hypothetica
listening...
who is the secretary general of united nations
The Secretary-General of the United Nations (UNSG or SG) is the chief administra
listening...
can you play follow you by imagine dragons on the ellen degeneres show
```

Conclusion/Outcome

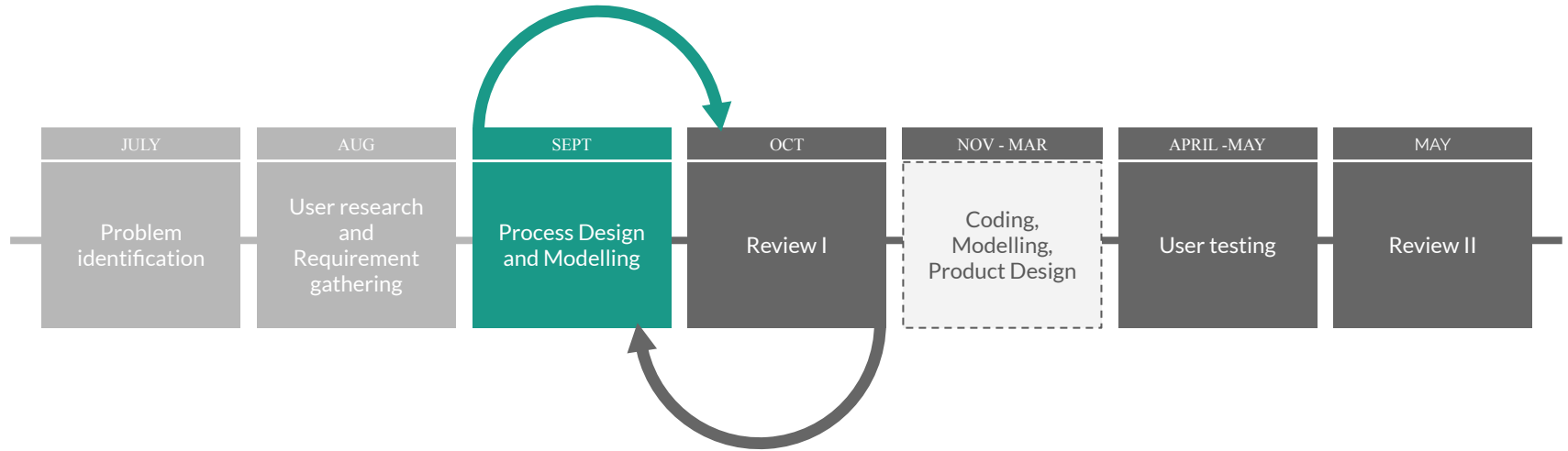
A functional real time vision based american sign language recognition for D&M people have been developed for ASL alphabets. We have achieved an accuracy of 0.974 for Training and 0.968 for testing.

In the ASL 2.0 module, we created a real time Sign Language Detection Tutorial with Tensorflow Object Detection, python and OpenCV. In this module instead of detecting alphabets, we are detecting words using a webcam. This real time system detects the hand gestures in a video frame and shows the detected phrase with it's detection probability.

The text to speech module, we implemented email access via text processing and speech generation. This module converts text into voice using a speech synthesizer thus makes digital content more accessible to a blind person.

The personal voice assistant system presented is a very fundamental system with few features for the people who have trouble navigating through the phone. People just have to command what they want to do and via speech Recognition we capture that voice and turn that voice into a text and thus with the help of the text sentence formed, we perform the given task.

Timeline



Responsibility of each team member

Akansha Parmar (18BCE10020)	Finding dataset, research work, and testing
Anamika Lochab (18BCE10035)	Module 2- ASL 2.0
Meenakshi Yadav (18BCE10159)	Module 4- Speech to text
Yashika (18BCE10302)	Module 3- Text to speech
Tenzin Gendun (18BCG10098)	Machine Learning Algorithm Research, Source Code Development (Python)
Ravisam Patel (18BCY10071)	Feasibility Research and Analysis
Tenzin Tsundue (18BCY10108)	Module 1- ASL 1.0, Database collection

References (Books) -

- T. Yang, Y. Xu, and “A. , Hidden Markov Model for Gesture Recognition”, CMU-RI-TR-94 10, Robotics Institute, Carnegie Mellon Univ.,Pittsburgh,PA, May 1994.
- Pujan Ziaie, Thomas Muller , Mary Ellen Foster , and Alois Knoll“A N  ive Bayes Munich,Dept. of Informatics VI, Robotics and Embedded Systems,Boltzmannstr. 3, DE-85748 Garching, Germany.
- Mohammed Waleed Kalous, Machine recognition of Auslan signs using PowerGloves: Towards large-lexicon recognition of sign language.
- PigouL.,DielemanS.,KindermansPJ.,Schrauwen B.(2015)Sign Language Recognition Using Convolutional Neural Networks. In: Agapito L., Bronstein M., Rother C. (eds) Computer Vision - ECCV 2014 Workshops. ECCV 2014. Lecture Notes in Computer Science, vol 8925. Springer, Cham.
- Zaki, M.M., Shaheen, S.I.: Sign Language Recognition Using a combination of new vision based features.Pattern Recognition Letters 32(4), 572–577 (2011)25
- N. Mukai,N.Harada and Y.Chang,"Japanese Fingerspelling Recognition Based on Classification Tree and Machine Learning," 2017 Nicograph International (NicoInt), Kyoto, Japan, 2017, pp. 19-24.
- Byeong Keun Kang, Subarna Tripathi, Truong Q. Nguyen ”Real-time sign language fingerspelling recognition using convolutional neural networks from depth map” 2015 3rd IAPR Asian Conference on Pattern Recognition (ACPR)

Questions?

References (URL) -

- <http://ijcsit.com/docs/Volume%205/vol5issue01/ijcsit2014050166.pdf>
- <http://www-i6.informatik.rwth-aachen.de/~dreuw/database.php>
- <https://towardsdatascience.com/applied-deep-learning-part-4-convolutional-neural-networks-584bc134c1e2>
- https://docs.opencv.org/2.4/doc/tutorials/imgproc/gaussian_median_blur_bilateral_filter/gaussian_median_blur_bilateral_filter.html
- <https://machinelearningmastery.com/how-to-configure-image-data-augmentation-when-training-deep-learning-neural-networks/>
- https://www.ripublication.com/ijaer18/ijaerv13n9_90.pdf
- <https://www.youtube.com/watch?v=aircAruvnKk&t=597s>
- https://www.researchgate.net/publication/304600612_Review_on_Text-To-Speech_Synthesizer
- <https://link.springer.com/article/10.1007/s40747-021-00324-x>
- <https://datascience.stackexchange.com/questions/22760/number-and-size-of-dense-layers-in-a-cnn>
- <https://heartbeat.fritz.ai/classification-with-tensorflow-and-dense-neural-networks-8299327a818a>
- <https://code.activestate.com/pypm/easyimap/>
- <https://ai.googleblog.com/2020/06/leveraging-temporal-context-for-object.html>
- <https://www.irjet.net/archives/V7/i9/IRJET-V7I9678.pdf>
- https://www.researchgate.net/publication/304600612_Review_on_Text-To-Speech_Synthesizer
- <https://towardsdatascience.com/image-data-labelling-and-annotation-everything-you-need-to-know-86ede6c684b1>
- <https://blog.roboflow.com/labelimg/>

THANK
YOU

