# SMARTSL: Real-Time Enhanced Gesture Recognition Framework and Tool Utilizing American Sign Language

**Ahmed H Alanazi**

School of Computing and Engineering

UMKC, Kansas City, MO
aha85b@mail.umkc.edu

**Chandni Acharya**

School of Computing and Engineering

UMKC, Kansas City, MO
capr5@mail.umkc.edu

**Yousef Almutairi**

School of Computing and Engineering

UMKC, Kansas City, MO
yousefalmutairi@mail.umkc.edu

**Aswini Priya Ganesh**

School of Computing and Engineering

UMKC, Kansas City, MO

ag8mw@mail.umkc.edu

Abstract ---- This project recognizes and interprets the American Sign Language (ASL) symbols that have hand gestures. The proposed model provides an opportunity for the people with hearing disability to communicate and learn using computers. American Sign Language is a widely used and accepted standard of communication by people with hearing disabilities. The signs are recognized by identifying and tracking the regions of interest using OpenCV segmentation feature. The training and prediction of hand gestures are performed by applying both Convolutional Neural Networks (CNN) and Recurrent Neural Networks (RNN) combined with Long Short-Term Memory (LSTM), capable of recognizing the sequence of gestures. The model is deployed using Flask application. The gestures are captured through the webcam and the hand gestures are predicted in real time. The proposed system is used to recognize the real-time signs. Thus, this system is very much useful for hearing impaired people to communicate with normal people.

**Key Words**: SmartSL, SSL-CNN, SSL-RNN, LSTM, MediaPipe, BigData, Machine/Deep Learning

## I. INTRODUCTION

Sign languages is a mean of communication which convey meaning through visual-manual modality (ability to see) [1]. Sign languages is the combination of hand shapes, movement of the hands, arms or body, and facial expressions. For deaf and muted people sign language is a useful means of communication and plays a vital role in deaf culture. Deaf culture is described as set of social beliefs, behaviors, art, literary traditions, history, and values [2].

American Sign Language (ASL) is a predominant sign language [3] and contains all the fundamental features of a language. The ASL is as rich as another language. Many people use ASL around the world. It is a fully developed and natural language [6]. Some suggests that ASL has been in use for more than 200 years [4]. Even though ASL is gaining increased recognition, it has some failures in teaching it. So, it is not only difficult for a person to learn ASL but also the teachers experience difficulties [5].

The aim of this project is to build a model to recognize American Sign Language (ASL) using computer vision and Google MediaPipe in real time. ASL is the most widely used sign language in the world. We intend only to recognize individual sign. We created two separate models - RNN (Recurrent Neural Network) – LSTM (Long Short-Term Memory) and Convolutional Neural Network model to train the dataset. Few ASL gestures has been created for training and testing. Actions like peace, like, dislike, ok, etc. has been used in the dataset with large number

of frames and sequence for each action. Using Media-pipe which is a hand and finger tracking solution we predict the word. The trained deep learning models have been then deployed to Flask on which a web application is created. The web application allows the user to have experience of creating their own dataset and training the model after which they can predict to see the gestures.

## II. RELATED WORK

There has been lot of research carried out to help ease the communication of deaf and mute individuals [7-12]. There are several technologies developed that helps their communication easier, for example, wearable communication devices, online learning systems. The idea behind these techniques has been making use of image acquisition, skin segmentation, background subtraction and gesture identification. However, these systems require sensors, accelerometer, and external device to interpret the message and the online learning cannot be accessed by the illiterates. Hence, sign language recognition models came into existence.

Hand gesture recognition considers similarities of human hands shape, thus provides user friendly interaction. In sign language each gesture has specific meaning, the complex meanings can be expressed by combination of simpler gestures. By using all these techniques deaf and mute people can share their thoughts freely without any restrictions. In general, hand gestures are recognized using two different approaches.

There are various machine learning algorithms which were used for the recognition of hand gestures and few of them are discussed here.

### 2.1 Contour Based Approach

Kaur et.al., (2018) [7] proposes a real-time hand gesture recognition system with four modules such as Data acquisition, Pre-processing, Feature extraction, Gesture recognition, and the Real-time hand detection is done, and k-Nearest Neighbor (KNN) algorithm is used to classify the input images. A technique using six static and eight dynamic hand gestures is proposed in [8] with three main steps hand shape recognition, tracing of detected hand (if dynamic), and converting the data into the required command, the system used a CNN architecture to pre-train model to recognize the gestures. The application classified the eight gestures with 93% accuracy. [9] Introduces an effective Hand Gesture Recognition system where the feature extraction is done using Deep Convolutional Neural Network (DCNN) and the classification is done using a Multi-class Support Vector Machine (MCSVM). Distinct person hand gestures were used for validation and the model shown a satisfactory performance with 94.6% classification accuracy.

### 2.2 Hybrid AI Approach

In Masood et.al., (2018) [11], used a similar approach to the one proposed in this paper. The authors have used an inception model which used deep CNN and RNN to train on both the spatial and temporal features contained in the video sequences. They trained their model on the 46 gestures from Argentinian Sign Language and achieved an accuracy of about 95%. The results obtained were really good but this model has limitations in classifying the real time recognition as the authors removed the background in order to detect the hand gestures.

Li et.al., [12] studied the real time recognition with LSTM in deep learning models. To adapt to the large variation of signs and make reliable handcrafted features which is generally a limitation in Hidden Markov Models (HMM), has been overcome with the help of LSTM

layers. After comparing LSTMs with other methods on the trained large Chinese Sign Language (CSL) dataset, the authors claim that LSTM based model works better than HMM.

### 2.3 AI Model using MediaPipe

Similar to our proposed model in this paper, Bagby et.al (2021) [13] proposed a prototype application for client-server separation of the more resource intensive parts of the application and the camera input and command output technology. The authors built a model that recognizes hand gestures for smart home devices using Google MediaPipe tool. They used the video captured from the integrated video camera as individual frames using OpenCV tool and integrated it with the MediaPipe and Python framework containing the gesture recognition model. MediaPipe comes with a hand tracking project that detects a hand and tracks its movement using 21 defined landmarks on the detected hand. Their model used a CNN architecture to predict 4 different gestures from the ASL. After running multiple experiments, the authors claim that MediaPipe could be easily used as a tool to accurately determine hand gestures with a higher accuracy of about 99%. One of the major drawbacks of this approach is further manipulation of hand landmark data is required for more complex and dynamic gestures, such as tracking individual velocity of landmarks for signs such as J or Z, or any other moving hand gestures [13].

### 2.4 Depth Based Approach

There is another approach called Depth based approach which is robust in gesture recognition. In this approach, the 3D geometric information depth camera is used. The latest depth-based gesture recognition uses a Kinect device, which is used to find the depth of the object [10]. The limitation of this method is that the Kinect camera is more expensive.

## III. PROPOSED WORK

The primary idea of the application is to help the people with hearing disability along with their friends and family can use in daily life on any web browser. The application runs on flask and allows the end users to open the camera and make few gestures which would be predicted by the application and displayed on the browser. Once the users are done, they can close the camera and close the application. The prediction is done using mediapipe library and machine learning algorithm called RNN i.e., Recurrent Neural Network and CNN i.e., Convolution Neural Networks along with OpenCV and TensorFlow.

## IV. APPROACH

This section focuses on building our framework and developing our SmartSL tool.

### 4.1 FRAMEWORK

This section will cover the framework of SmartSL. Label method is a combination of OpenCV and MediaPipe to collect byte order of BGR and store them in Numpy file. Our label method opens a webcam through openCV then mediapipe draws landmarks of right and left hand by object detection technique. Drawing function returns the byte orders of left- and right-hand landmarks as shown in Fig[1 &2]. This process is detecting the movement of human's hand, also, mediapipe specify whether it is a right or left hand. We named it a Label Feature because the logic behind it is sequences of each gesture. For example, 'Like' is one of the four words that has been included in this project, if 'like' has sequence of 30 that means there are 30 subfolders and, in each subfolder, consist of 30-byte orders saved as .npy file. All mentioned data are saved in a directory for easy retrieval.

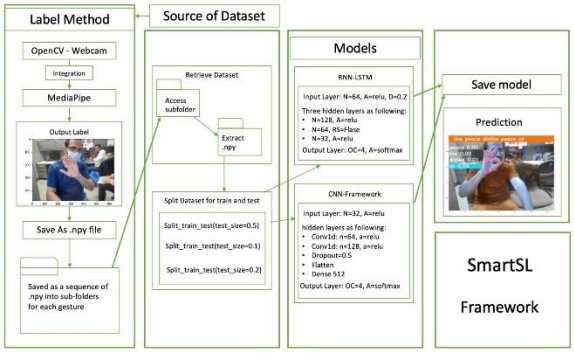



*Figure 1: SmartSL Framework*

## Label Feature

Using our Label method is saving steps of preprocessing data because. npy is ready to be passed to split_train_test function. In the first stage of our project, byte orders of BGR were saved in .csv file which requires a preprocessing procedure such as cleaning extra commas. During this project, out framework went through different hyperparameters, for instance, train and test size changed several times between 80 percent for training and 20 percent for testing. As a result, train size has been chosen to be 90 percent and test size 10 percent. Our label feature is useful in many aspects such as saving time and avoid preprocessing.

**RNN – LSTM**

RNN has been used due to the fact our dataset is a sequence. Therefore, LSTM takes the sequence of byte orders of BGR and pass it to the model. In this model, there are several layers as following; input layer of 64 neurons, activation is relu and dropout of 20 percent. Three hidden layers are being used at 128, 64 and 32. Lastly, the output layer with 4 classes as an output and activation is softmax.

**CNN**

CNN has been used to be compared with RNN, also, adding higher performance in training and testing process. Our CNN layers consist of different parameters and number of layers as well. In this input layer using Conv1D, it starts with 32 neurons, 5 kernel size, input shape of (5, 126). Hidden layers as following; 1) conv1d: 64 neurons, activation is relu 2) conv1d: 128 neurons, activation is relu 3) dropout is 0.5 4) Flatten 5) Dense: 512 neurons. Lastly, output layer with a 4 classes and activation is softmax.

Finally, the model will be saved to be passed into prediction phase. The output is four classes because the target is four different words ‘pease’, ‘like’,’dislike’ and ‘ok’.

## 4.2 SMARTSL TOOL

SmartSL tool consists of web development, Machine/Deep learning frameworks and backend operations. Mainly, Python-Flask has been chosen to build the front/backend. For the first phase, OpenCV was chosen to collect a dataset through capturing images, then a MediaPipe framework integrated into our startup development. “Open-Source Computer Vision Library is an open-source computer vision and machine learning software library” [14]. The role of OpenCV is to open a webcam which attached on a laptop/PC machine. Also, it contains more than 2000 algorithms that can be used in different tasks and functions such as face recognitions, object detection, classify human actions, track camera movement, etc. MediaPipe is a machine learning solution for various issues in particular for live and streaming media [15]. Google created MediaPipe framework to assist in building multimodal such videos and enhanced its adaptability for mobile and web devices. MediaPipe is an open source with an optimized performance.

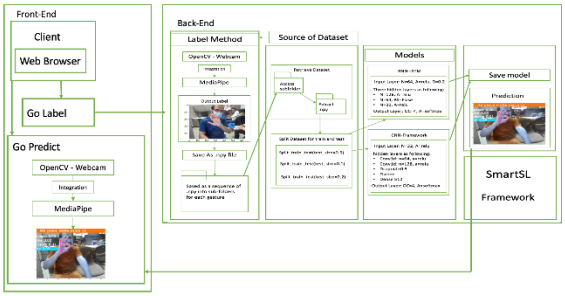



*Figure 2: SmartSL Tool*

In operation, using OpenCV and MediaPipe for collecting the byte orders of BGR patterns by MediaPipe's landmarks algorithms, then byte orders are saved as NPY file [16]. NPY file extension is a NumPy array file which can be created using python software package from NumPy library. The useful feature of NPY file is information can be reconstructed on any computer. After that, our framework extract information from NPY file to be passed to the Neural Network.

The development of the final Web/MDL learning tool was built by Flask as a web application with the MDL running at the backend as following:

- Graphic User Interface
  - HTML & CSS
    - In this section, webpages were developed and styled
  - JavaScript
    - Add functions to control behavior of HTML tags.
  - Label Feature(GUI) (Figure 5)
    - Users can open their webcam through browser
    - Users follow instructions which provided on the video screen
    - Label operation will stop once the gestures are collected
    - Users can train the gestures using CNN or RNN
    - Available word gestures are "Peace", "Like", "Dislike", "Ok"
  - Predict Feature(GUI) (Figure 9)
    - Open webcam
    - Test gestures
    - Users can see if the right gesture has been predicted or not
- Backend
  - Python
  - Created APIs
  - Functions
  - Collecting Labels
    - Train Collected Datasets
      - CNN saved as .h5
      - RNN saved as .h5
  - Predict Trained Data

This project is running on a localhost using Python-Flask library. Flask is a library that builds web application with enhancing their performance and availability [17]. Our web application is easy for users without programming or Machine/Deep learning knowledge to interact with. In this case, Deaf people will not face difficulty in using out web application.

## V. EXPERIMENTS AND IMPLEMENTATION

This section of the paper emphasizes on the creating of the dataset involving few sign gestures, building a RNN model and CNN model along with usage of mediapipe, training the model as well as working of the training model for prediction and display of those gestures. It also focuses on the implementation of the application using Flask along with html on which the trained model runs for prediction of the various gestures.

### 5.1 Initial Idea

Initially, we started with creating a dataset using our laptop camera by capturing around 300 counts in total for 5 gestures like hello, yes,

IloveYou, thanks and No. After which we started to label the necessary areas in the images using LabelImg application [18] and create XML file for each image which included the various axes of the image being labelled along with its location in the system and few other information. We then tried to create a CNN model to train with the created images, but we faced few issues when going ahead with the same approach as labelling the images manually was a huge task which was tedious and time consuming. Even after increasing the size of the dataset, the accuracy was not up to the mark for predicting the gestures because of few issues with the model creation.

**5.2 Experimentation**

We modified our design approach for creating gestures and chose a method where labelling was automated which helped us to utilize our time to create the dataset of gestures from scratch for training the model and predicting the gestures afterwards. The approach mentioned in the section III (Proposed Work) was using MediaPipe, RNN and CNN for prediction of gestures and using Python-Flask for creating the application/tool.

The benefit of using MediaPipe machine learning framework by Google was because it comes up some pre-defined machine learning solutions like pose recognition, hand recognition, object detection which help in capturing all the gestures using few pre-defined key points. For the real time usage of camera, OpenCV library in python was used.

- **Collecting landmarks by MediaPipe**

The first phase of this implementation is collecting the gestures. The gestures are collected by the system camera making use of OpenCV library and mediapipe framework as BGR images (later converted to RGB images). The 21 right-hand and left-hand landmarks each are captured and stored in individual sequence folders for each gesture. The gestures are converted to numpy arrays based on the key points of the landmarks.

- **Splitting Train-Test and One-hot encoding**

The dataset created is splittted into training and testing dataset (using multiple combinations for splitting the dataset) using the train_test_split library. After which, the labels or gesture names were one-hot encoded, and all the landmarks were converted to numpy array and concatenated.

- **Defining Neural network models(CNN /RNN)**

We have created one Recurrent Neural Network sequential model with 3 LSTM layers (64 and 128 filters) with Relu activation and 2 dense layers with the output of last dense layer(with Softmax activation) being no of actions of gestures.

The Convolution Neural Network sequential model consists of 3 combination of Convolution 1D and Max pooling layers with 32,64 and 128 filters respectively with relu activation function. The model also consists of Flatten layer and Dropout for avoiding any overfitting. The output of last dense layer(with Softmax activation) was no of actions of gestures. The model is compiled using Adam optimizer and categorical entropy loss function and is being trained for 1500 epochs.

- **Training , Saving, and testing the model**

The models are compiled using Adam optimizer and categorical entropy loss function. The training is being performed for 1500 epochs. Both the models are saved as .h5 for being used in further prediction. The saved model is used to predict the gestures in real time.

**Implementation Steps**:

- ✓ The first step was to install all dependencies for OpenCV library and mediapipe along with Tensorflow
- ✓ Implementing the opening of camera using OpenCV library and detecting the gestures and converting them from BGR( default in mediapipe) to RGB
- ✓ Create methods for defining the various landmarks for left hand, right hand using a built-in method DrawingSpec.
- ✓ Setting the media-pipe model for capturing the frames using OpenCV and mediapipe detection and also capturing landmarks.
- ✓ Setting up path and creating various folders for different gestures along with individual sequence of folders for number of gestures captured
- ✓ Collecting frames for each gesture using the hand recognition key points

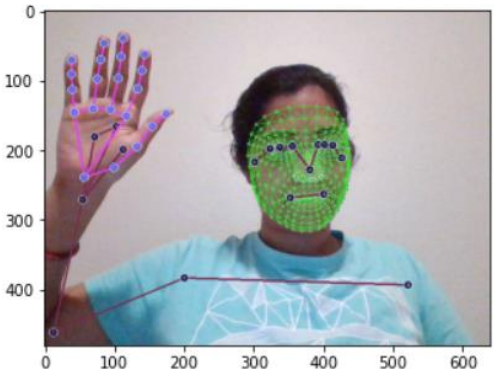



*Figure 3: capturing gestures with media pipe framework*

- ✓ Extraction of key points, converting and combining into numpy arrays
  ( Concatenating the key points from all landmarks together)
- ✓ Splitting the dataset into training and testing and training the model( using combination of LSTM/Convolution layers and Dense layers)
- ✓ Creation of tool for end users and loading the trained model using python-flask.
- ✓ Accessing the webcam of the system using the application to capture gestures, train the model(using the backend logic) and make prediction in real time.
- ✓ The prediction of the words based on the gestures are displayed on the screen as text along with the probability. [Figure 10]

## VI. WORKING SMARTSL TOOL

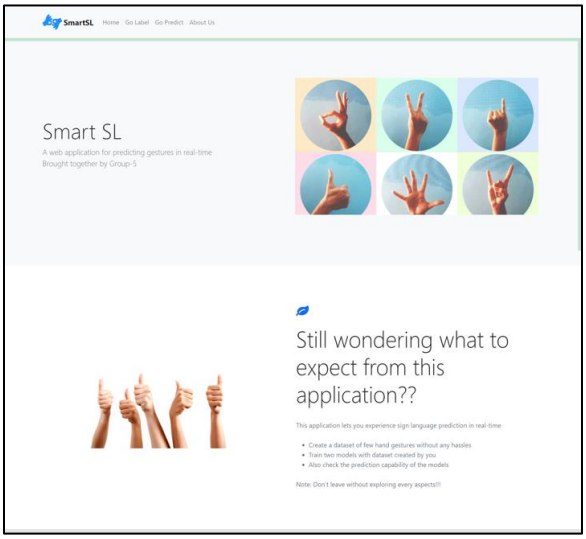



*Figure 4: Home page of the tool*

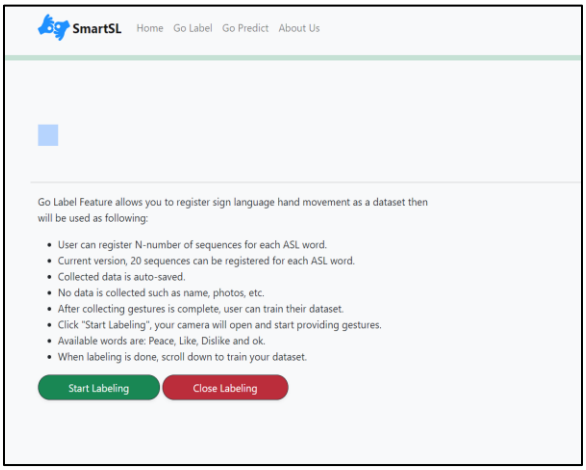



*Figure 5: Go label page*

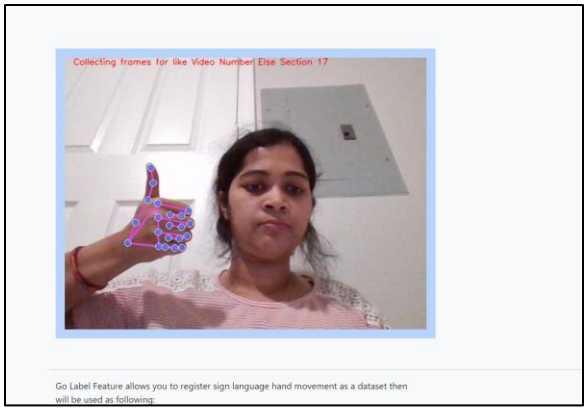



*Figure 6: Open Camera functionality of Go Label*

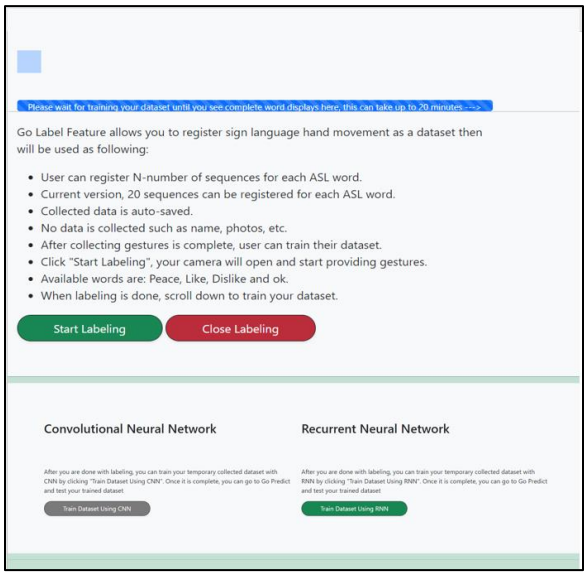



*Figure 7: Train using Model functionality on Go Label Page*

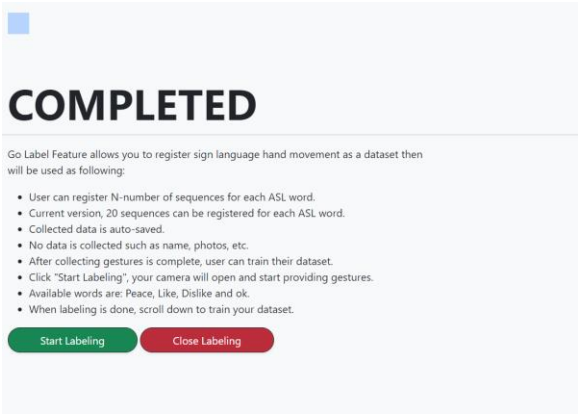



*Figure 8: Completed message displayed after training completion*

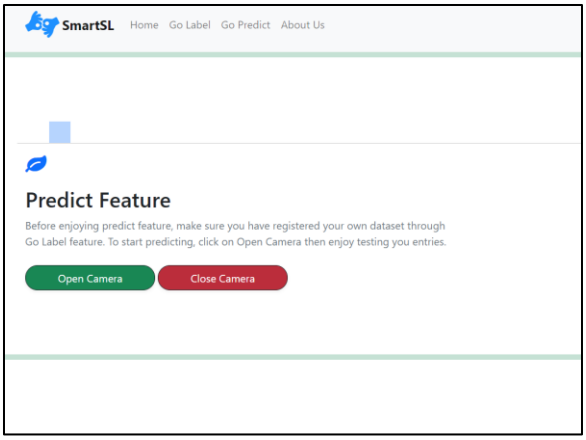



*Figure 9: Go Predict page*

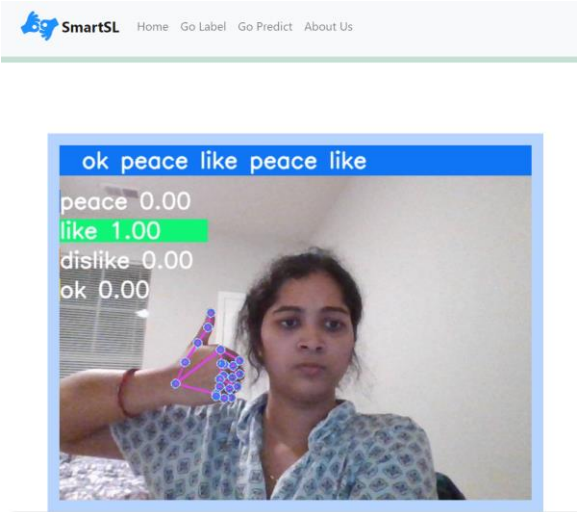



*Figure 10: Open camera functionality showing Prediction*

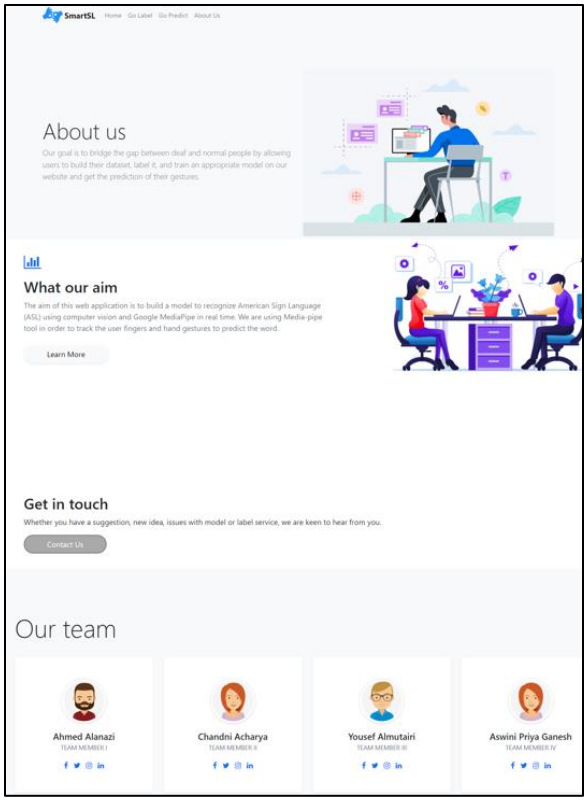



*Figure 11: About Us page*

## VII. EVALUATION AND RESULTS

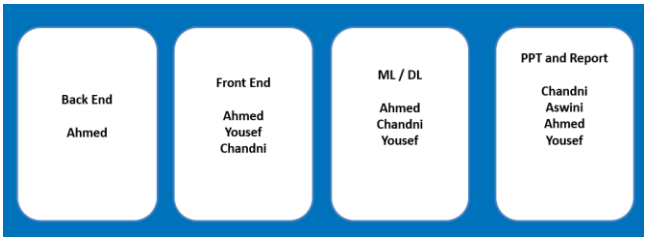
This part of the paper accentuates on working of the Real-Time Enhanced Gesture Recognition Framework and Tool Utilizing American Sign Language (SMARTSL). We have completed the development of a web application that allows users to create, train, and predict their datasets.

There are many things we have addressed at the beginning in order to design an excellent ASL framework, the first thing is the dataset, which we created ourselves and used that to train the model. Second, the model that can provide us with accurate prediction accuracy. The last thing is that how we can provide this service to everyone on the internet to collect data, train it, get the prediction without any knowledge of the ML/DL.

We have tried to develop our application continuously in order to achieve a good framework with high accuracy. Firstly, we have created our own dataset by using python program to take a hand gesture from our laptop camera and label that manually using LabelImg application. It was very long process to get a sufficient image in short period, that can be split to train. Therefore, we decided to not move forward with manual labelling, and use MediaPipe machine learning framework instead which can assist us with a real-time gesture recognition and auto-labelling it. In addition, we developed a web application using Flask framework to allow end users to open the camera and take hand gestures that can be predicted by the application.

The results have been evaluated using different metrics. First, we used MediaPipe as a finger tracking solution to recognize end-user's hand gestures, and auto-label it. The accuracy of our project is how the model classifies the detected gestures in real time to the right word that have labelled. For instance, once the user show "thank you" gesture on the camera, the probability word will appear on the progress bar and be printed at the top of the screen.

## VIII. CONTRIBUTION




## IX. CONCLUSION

In this project, we presented an idea for creating an easier communication platform between hearing impaired and the normal people. The SMARTSL tool is developed which gives the end users a chance to capture their gestures, train the model and look at how good

or bad the model predicts. We have used two different models for training the gestures (CNN and RNN) along with Media-pipe framework which helps in capturing key points of hand landmarks of the gestures. Finally, the GUI is developed using python-flask along with HTML /CSS after creating, labelling gestures, training the models, and loading the trained models over the local host on the backend.

## X. REFERENECES


[1] ”What is Sign Language?". Archived from the original on 13 February 2018. Retrieved 10 March 2018. Available: https://web.archive.org/web/20180213195125/https://www.linguisticsociety.org/content/what-sign-language

[2] Padden, Carol A.; Humphries, Tom (Tom L.) (2005). Inside Deaf Culture. Cambridge, MA: Harvard University Press. p. 1. ISBN 978-0-674-01506-7.

[3] About American Sign Language, Deaf Research Library, Karen Nakamura. Available: http://www.deaflibrary.org/asl.html

[4] Online: https://www.nidcd.nih.gov/health/american-sign-language

[5] Rachel Locker McKee, David Mckee January 1992Sign Language Studies 75(1):129-157 DOI:10.1353/sls.1992.0000. Available: https://www.researchgate.net/publication/265828936_What%27s_So_Hard_About_Learning_ASL_Students%27_Teachers%27_Perceptions

[6] Online: https://www.nad.org/wp-content/uploads/2018/06/List_States_Recognizing_ASL.pdf

[7] Sanmukh Kaur, Anuranjana (2018), “Electronic Device Control Using Hand Gesture Recognition System for Differently Abled”, 8th International Conference on Cloud Computing, Data Science & Engineering

[8] Soeb Hussain, Rupal Saxena, Xie Han, Jameel Ahmed Khan, Hyunchul Shin (2017), “Hand Gesture Recognition Using Deep Learning”, International SoC Design Conference.

[9] Md Rashedul Islam, Rasel Ahmed Bhuiyan, Ummey Kulsum Mitu, Jungpil Shin (2018), “Hand Gesture Feature Extraction Using Deep Convolutional Neural Network for Recognizing American Sign Language”, 4th International Conference on Frontiers of Signal Processing.

[10] Wang, Lei, Du Q. Huynh, and Piotr Koniusz. "A comparative review of recent kinect-based action recognition algorithms." IEEE Transactions on Image Processing 29 (2019): 15-28. Available Online: https://arxiv.org/pdf/1906.09955.pdf

[11] Masood, Sarfaraz, et al. "Real-time sign language gesture (word) recognition from video sequences using CNN and RNN." Intelligent Engineering Informatics. Springer, Singapore, 2018. 623-632.

[12] T. Liu, W. Zhou and H. Li, "Sign language recognition with long short-term memory," 2016 IEEE International Conference on Image Processing (ICIP), 2016, pp. 2871-2875, doi: 10.1109/ICIP.2016.7532884.

[13] Bagby, B., Gray, D., Hughes, R., Langford, Z. and Stonner, R., 2021. Simplifying sign language detection for smart home devices using google mediapipe.

[14] OpenCV. 2021. Home - OpenCV. [online] Available at: <https://opencv.org/> [Accessed 29 November 2021].

[15] *Live ML anywhere*. MediaPipe. (n.d.). Retrieved November 29, 2021, from https://mediapipe.dev/index.html.

[16](NPY File Extension - What is an .npy file and how do I open it?, 2021) https://fileinfo.com/extension/npy
[17]Flask.palletsprojects.com. 2021. Welcome to Flask — Flask Documentation (2.0.x). [online] Available at: <https://flask.palletsprojects.com/en/2.0.x/> [Accessed 29 November 2021].
[18] https://github.com/tzutalin/labelImg