Date: 09/05/2022

Sign Language Interpreter

By:

Pranav Unkule

ABSTRACT

Sign Language is used in the deaf / speech impaired community all over the world. In this report I have proposed a system that can interpret the Sign language in the domain of numerals, alphabets, and basic phrases in day-to-day communication, so that the less fortunate people will be able to communicate with the outside world in public places like **banks**, **malls**, **shops, restaurants,** etc. Sign Language interpreter devices are an urgent requirement at institutes and places where communication is important. So, the concept of Sign Language Interpreter Device came to mind.

The Interpreter device will be placed in key locations where communications is most necessary for example in banks, etc. With the help of Machine Learning, Computer Vision and Text to Speech, the interpreter will be able to convert the sign language gestures into a language like English in text as well as audio format.

Introduction:

Sign language, a form of communication used by people with impaired hearing and speech. People use sign language as a mean of non-verbal communication to express their thoughts and emotions. But the general population finds it extremely difficult to understand, hence trained sign language interpreters are needed during medical and legal appointments, and educational and training sessions. The spoken language is communication media for those who can speak and listen. But what about deaf people?

Automatic Speech Recognition has now advanced to the point of being commercially available, but Automatic Sign Language Recognition is still in its infancy. Currently, all commercial translation services are human-based, and therefore expensive. Due to this, most public services lack Sign Language Interpreters. So, I want to propose a Sign Language interpreter Device.

1. Market/Customer Need Assessment:

Due to high expense and high training requirement of human–based Sign Language Interpreters, there is a need for affordable devices that can interpret sign language and can be used in day-to-day places like banks, mall, restaurants, etc.

So, the goal is to create a device / system that will interpret the sign language and will convert it into a language like English in text as well as audio format.

The Interpreter device should be portable and energy efficient so that it just be added to public places without need of additional requirements.

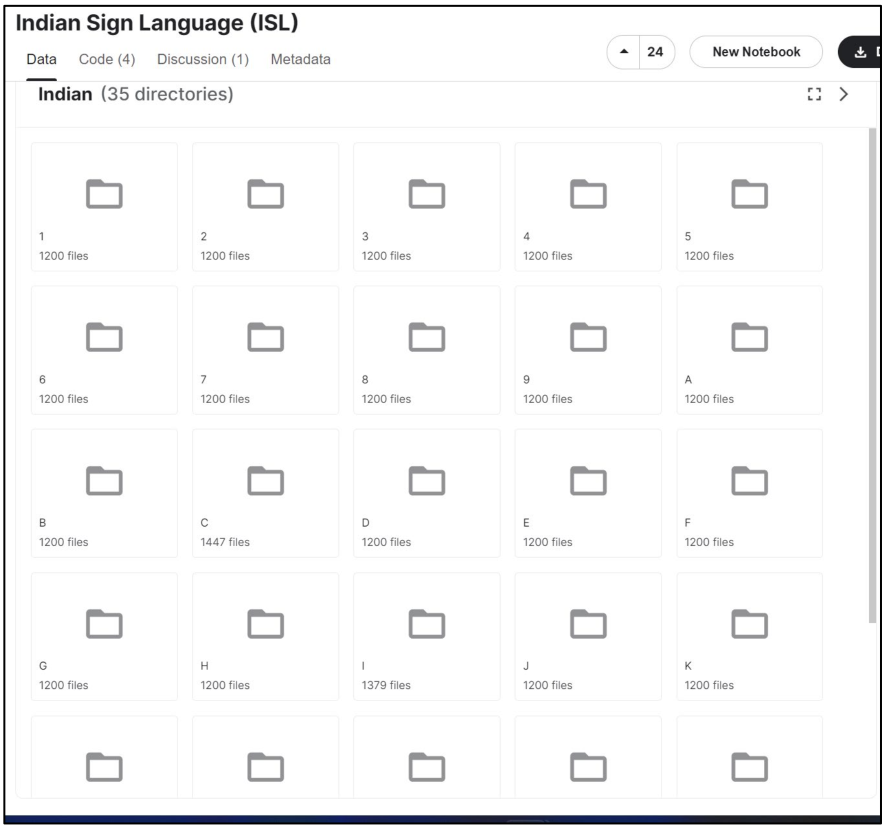
2. Target Specifications and Characterization:

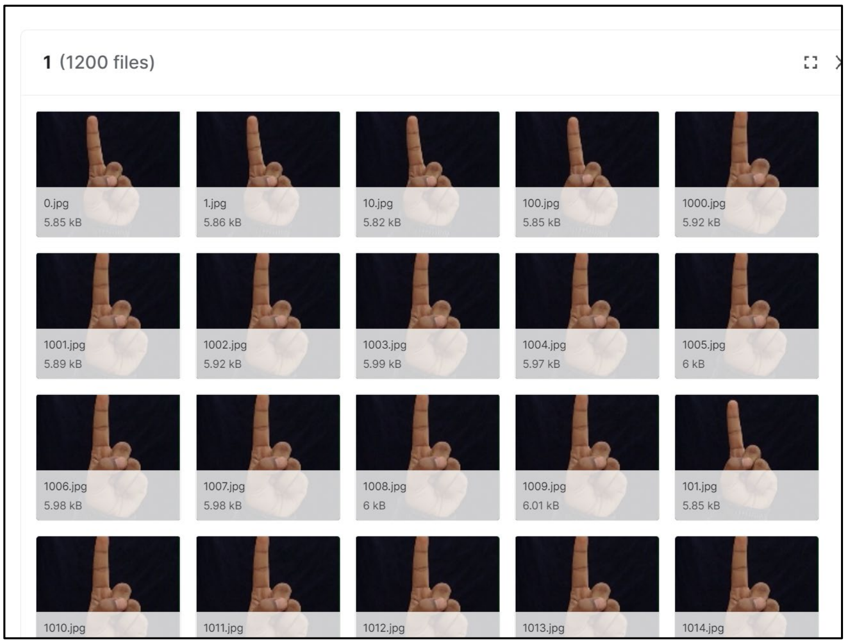
- Sign Language Gesture Recognition: The device will be able to identify sign language.
- Translation into suitable language: The device should will translate the identified sign into a language like English.
- Output into text and audio: The device will convert the sign into both text and audio.
- The device should be affordable: Even small business would be able to use it.
- Improve the Sign Language Dataset via updates.
- Improve / Add Languages to which Sign Language is to be translated to.

3. External Search:

4.1 Dataset for Indian Sign Language Alphabets and Numbers:

The Dataset can be found on Kaggle:
Link: https://www.kaggle.com/datasets/prathumarikeri/indian-sign-language-isl





4. **Applicable Regulations:**
   (Government and Environmental)

   a. Patents on ML algorithms developed
   b. Laws related to privacy for collecting video of users.
   c. Protection/ownership regulations.

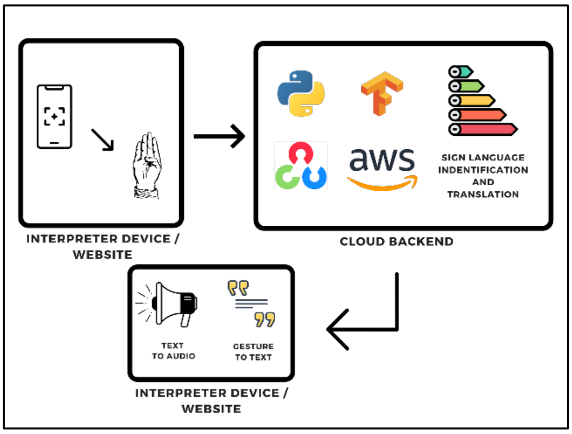
5. **Constraints:**
   a. Expertise in Sign Language: There are numerous Sign Languages used around the world. We need experts in the sign language we decide to use for the device.
   b. Expertise in Data Collection, Image processing: Sign Language is vast and complex hence requiring experts for collection of such complex data.
   c. Expertise in Machine Learning: For training a classification model for identification of signs.
   d. Expertise in Cloud: The Backend of the system will be deployed on the cloud hence the requirement.
   e. Expertise in Embedded Systems Development: To develop our device meeting all the requirements.
   f. Etc.

6. **Business Opportunity**:

   While Human translators are good at translating sign language, they are expensive and need more time if we want text as well for output. They also have down time and cannot work 24/7. Our device will be affordable and can run 24 / 7 with low maintenance required.

7. **Concept Generation**:
   Communication is an act of transferring information from one person to another. Nearly 6.3% population lacks the power of communication. Sign Language in turn work as a medium to express their feelings. So, I came up with concept of designing a system which will convert hand gestures into a text and audio.

8. **Concept Development and Details:**



The device will act as our frontend, it will capture the real-time video feed of the person using sign language. It will then send this to the cloud backend for translation and conversion of gesture to text and audio. Our device will then output this translated text and audio.

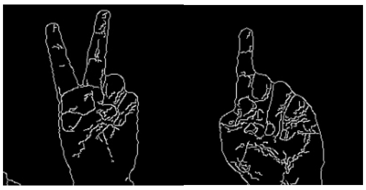
By keeping the Major portion of process in the cloud backend, we can update the ML Model to increase performance, add more signs and languages to translate to.

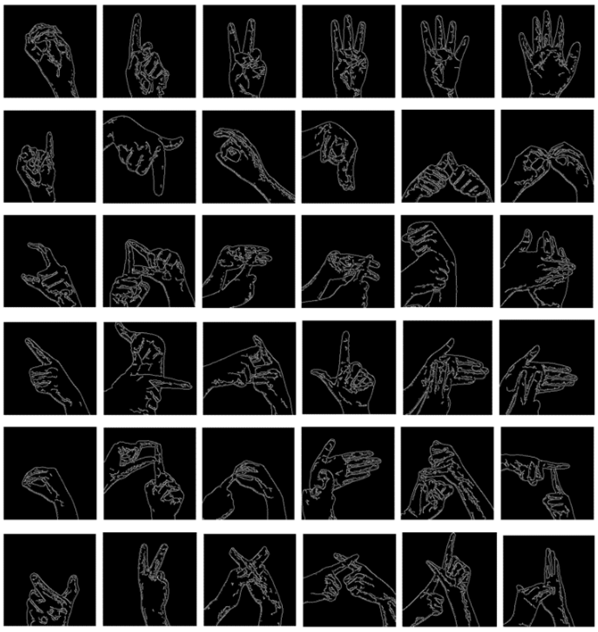
9. Code Implementation:

Sign Language is very vast. In this paper, I have implemented a CNN model for classification of Signs.
For Convolutional Neural networks (CNN), I have used the TensorFlow library, NumPy and pandas library for dataset handling, sklearn library for accuracy prediction purposes, and matplotlib library for graph plotting purposes.

9.1 Dataset:

Dataset is a very essential part of machine learning algorithm. Using OpenCV, the camera is used to capture multiple images. The image is converted to Grayscale and then masked. After grayscale conversion, canny edge detection is done. Then it is stored in a defined path. 500 images of each class are created.





9.2 CNN Model and training:

```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 54, 54, 96)        11712

 activation (Activation)     (None, 54, 54, 96)        0

 max_pooling2d (MaxPooling2D  (None, 26, 26, 96)       0
 )

 conv2d_1 (Conv2D)           (None, 26, 26, 256)       614656

 activation_1 (Activation)   (None, 26, 26, 256)       0

 max_pooling2d_1 (MaxPooling  (None, 12, 12, 256)      0
 2D)

 conv2d_2 (Conv2D)           (None, 12, 12, 384)       885120

 activation_2 (Activation)   (None, 12, 12, 384)       0

 conv2d_3 (Conv2D)           (None, 12, 12, 384)       1327488

 activation_3 (Activation)   (None, 12, 12, 384)       0

 conv2d_4 (Conv2D)           (None, 12, 12, 256)       884992

 activation_4 (Activation)   (None, 12, 12, 256)       0

 max_pooling2d_2 (MaxPooling  (None, 5, 5, 256)        0
 2D)

 flatten (Flatten)           (None, 6400)              0

 dense (Dense)               (None, 4096)              26218496

 activation_5 (Activation)   (None, 4096)              0

 dropout (Dropout)           (None, 4096)              0

 dense_1 (Dense)             (None, 4096)              16781312

 activation_6 (Activation)   (None, 4096)              0

 dropout_1 (Dropout)         (None, 4096)              0

 dense_2 (Dense)             (None, 10)                40970

=================================================================
Total params: 46,764,746
Trainable params: 46,764,746
Non-trainable params: 0
```

Training ¶

In [11]:
```
start_time = time.time()
with tf.device(tf.DeviceSpec(device_type="GPU", device_index=0)):
    history = Model.fit(train_dataset, epochs=5)
training_time = time.time() - start_time
```

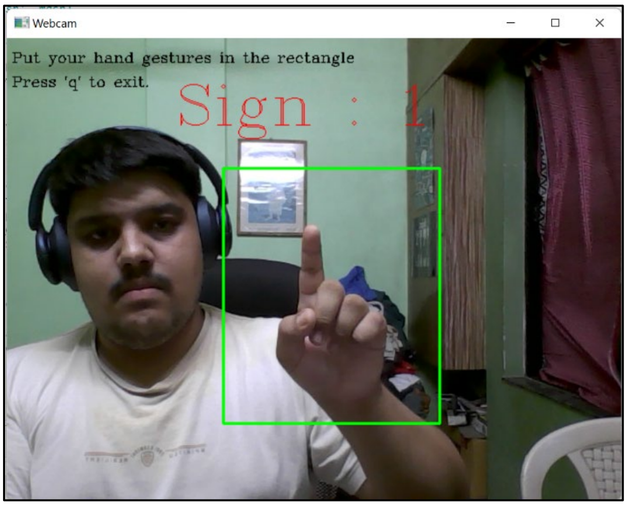
```
Epoch 1/5
500/500 [==============================] - 31s 43ms/step - loss: 1.2780 - accuracy: 0.5500
Epoch 2/5
500/500 [==============================] - 14s 29ms/step - loss: 0.1192 - accuracy: 0.9597
Epoch 3/5
500/500 [==============================] - 13s 27ms/step - loss: 0.0460 - accuracy: 0.9847
Epoch 4/5
500/500 [==============================] - 14s 29ms/step - loss: 0.0267 - accuracy: 0.9930
Epoch 5/5
500/500 [==============================] - 14s 27ms/step - loss: 0.0106 - accuracy: 0.9965
```

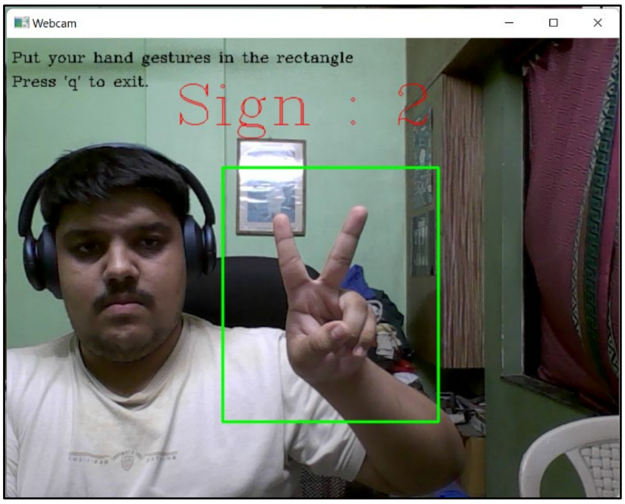
Testing

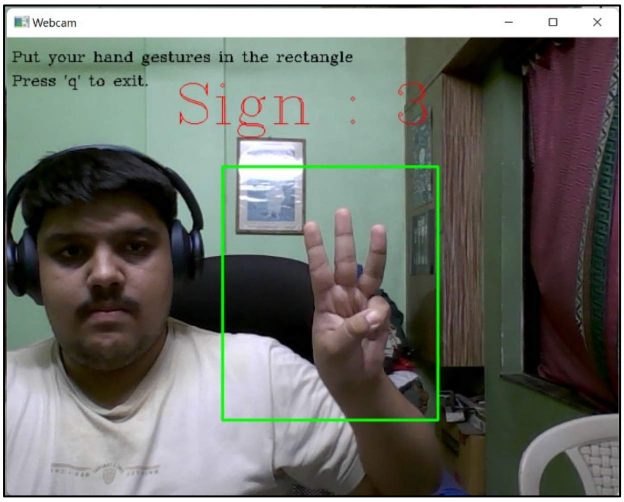
In [13]:
```
print('\nAccuracy on training set:',history.history['accuracy'][-1],'\nLoss on training set:', round(history.history['loss'][-1])
```

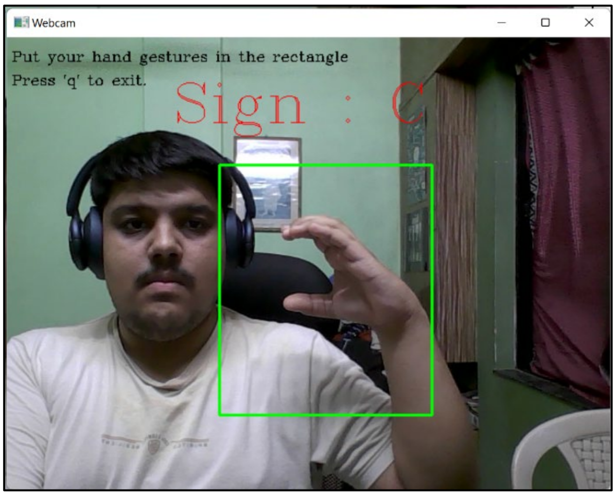
```
Accuracy on training set: 0.9964982271194458
Loss on training set: 0
```

9.3 Prediction









9.4 Code link: https://github.com/PranavUnkule/FeynnLabTask01/

10. Final Product:

a. **Frontend: (Device and Website / APP)**
   i. **User interface:** Options for user to chose between different signs languages and Language to translate to. Options to Mute the audio output. Options for updating the app.
   ii. **Realtime Capture:** Capturing real-time video of the user for Sign Language Interpretation.
   iii. **Feedback:** Feedback system forgetting user feedback for improving the device / system.

b. **Backend: (Cloud)**
   i. **Storage:** For storing Dataset, Model Parameters, User Info.
   ii. **ML Model:** For Identification of Signs.
   iii. **Text Conversion:** Converting Identified signs to proper sentences
   iv. **Text to Speech:** For converting converted text to audio.
   v. **Update Framework:** To update model, add more signs and languages to translate to.

Conclusion:

This Prototype will be a huge help to the deaf and speech impaired community. The Sign Language Interpreter device will be affordable and easy to use. It can be used in various places like banks, restaurants, etc. Further more the device can be updated frequently to improve performance and add more signs and languages to it.