Sign Recognition Project

Kanchan Patil
Vishwa Mehta

January, 2023

Contents

1. Abstract
2. Introduction
3. Overall Description
4. LSTM
5. Data-set Generation Process
6. Data-set Training Process
7. Data-set Testing Process
8. Future Scope

Abstract

Sign language is one of the oldest and most natural forms of language for communication. Most people do not know sign language and interpreters are very difficult to come by, therefore we have come up with a real time method using neural networks for finger-spelling based Indian Sign Language (ISL).

Introduction

1. Purpose
   The purpose of the project is to build an AI assistant designed specifically for speech impaired individuals that converts sign language to text using a video call interface
2. Project Scope
   2.1. The app provides a cheap solution as compared to sign language interpreters which are just costly to buy.
   2.2. It can simply be used with a web camera, making it efficient as compared to interpreters.
   2.3. This project provides ease of access for speech impaired individuals, as no additional equipment is needed

Overall Description

1. Project Functions
   1.1. Accepting input from webcam
   1.2. Detecting the signs
   1.3. Displaying correct output(translated sign)
2. Techstack
   2.1. Interfaces/Tools
      2.1.1. Jupyter
      2.1.2. Webcam
   2.2. Software dependencies
      2.2.1. OpenCV
      2.2.2. Mediapipe
      2.2.3. Tensorflow
      2.2.4. Keras

LSTM

Our project uses the LSTM network. In concept, an LSTM recurrent unit tries to "remember" all the past knowledge that the network is seen so far and to "forget" irrelevant data. This is done by introducing different activation function layers called "gates" for different purposes.

Pros:

1. LSTM is capable of dealing with more complex problems than the RNN by keeping a constant flow of error throughout the backpropagation from cell to cell.
2. They are much better at handling long-term dependencies.
3. LSTMs are much less susceptible to the vanishing gradient problem.

Cons:

1. LSTMs take longer and more memory to train
2. LSTMs are easy to overfit
3. LSTMs are sensitive to different random weight initializations

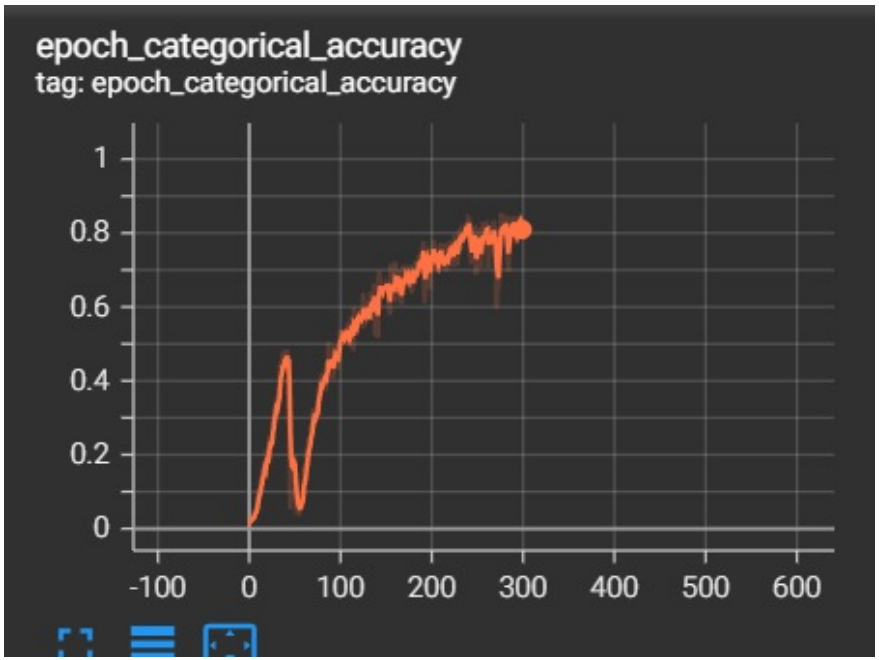
Data-set Generation Process

We use the OpenCV library of python to capture images and snippets of the training data that gets stored into a folder "MP_Data" with each sign as a separate sub-folder. As of now the following are the signs that have been included:

1. The Alphabet(A-Z)
2. First 10 numbers(1-10)
3. 12 Common phrases(hello, how are you, my name is, welcome, thank you, please, sorry, bye, good morning, good afternoon, good evening, good night)

Data-set Training Process

The model is trained using 3 layers of LSTM and 3 Dense layers with the Relu activation function and Softmax for the last layer. The optimiser used is Adam. The following data set has been trained for 300 epochs giving an accuracy of 84.60% when it is tested against itself. Finally this model is saved and used for the training phase.



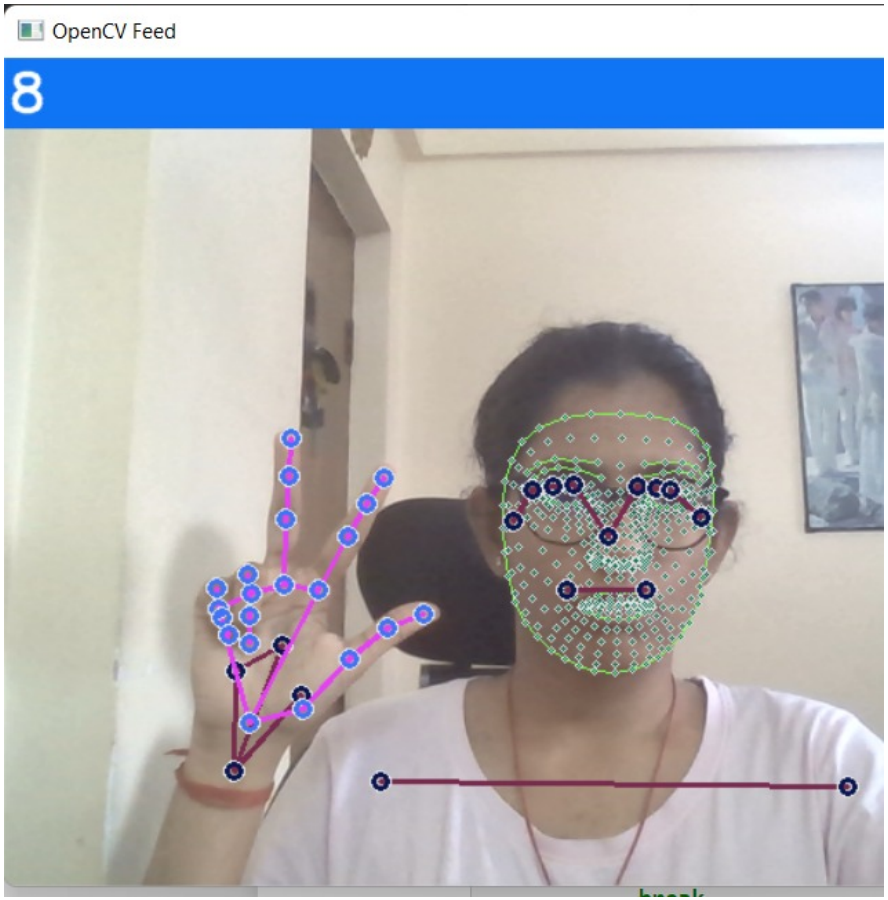

```
Model: "sequential_7"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 lstm_21 (LSTM)              (None, 30, 32)            216960

 lstm_22 (LSTM)              (None, 30, 128)           82432

 lstm_23 (LSTM)              (None, 32)                20608

 dense_21 (Dense)            (None, 64)                2112

 dense_22 (Dense)            (None, 32)                2080

 dense_23 (Dense)            (None, 48)                1584

=================================================================
Total params: 325,776
Trainable params: 325,776
Non-trainable params: 0
_________________________________________________________________
```

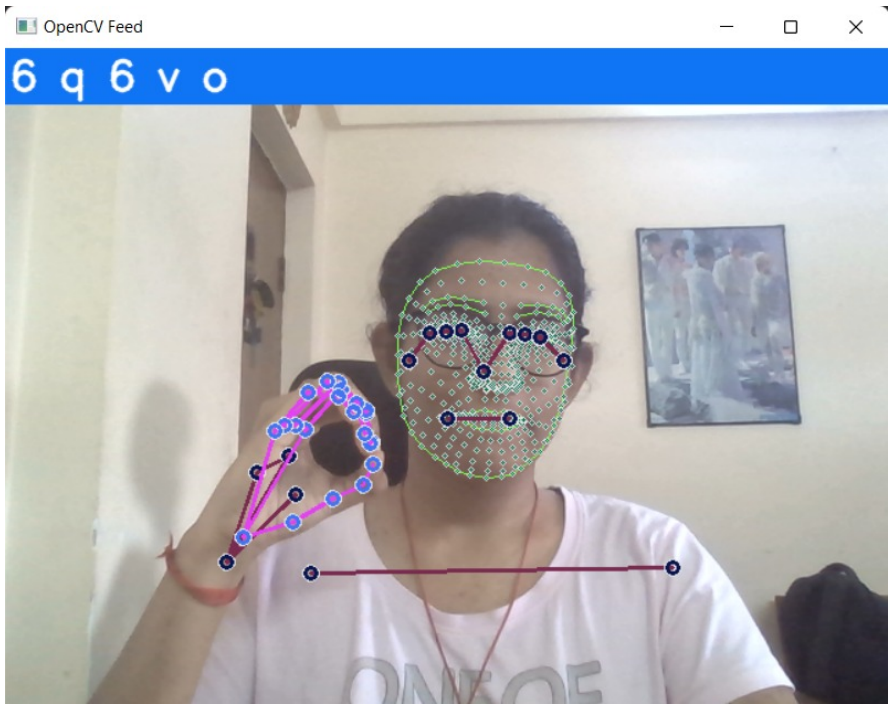
```
accuracy_score(ytrue, yhat)
```

```
0.8460648148148148
```

Data-set Testing Process

The testing of the model is done once we have the saved trained model. Here we have tested for some of the signs like 6, o, q, v, 8.





Future Scope

This project is still in its initial stages and definitely needs much improvement in terms of accuracy and additional signs to the data set, in order to predict a wider range of signs. Other scopes include:

1. Front-end/UI for the application
2. Conversion into a browser extension

You can checkout our project and contribute to it on GitHub 'here' by creating issues.