GALGOTIAS UNIVERSITY

(Established under Galgotias University Uttar Pradesh Act No. 14 of 2011)

SIGN LANGUAGE TRANSLATOR

A Project Report of Capstone Project – 2

Submitted by

ASHISH KUMAR SAH

(1613101194)

in partial fulfillment for the award of the degree

of

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE AND ENGINEERING

SCHOOL OF COMPUTING SCIENCE AND ENGINEERING

Under the Supervision of

Mr. A.Arul Prakash

Assistant Professor

APRIL / MAY- 2020

SCHOOL OF COMPUTING AND SCIENCE AND ENGINEERING

BONAFIDE CERTIFICATE

Certified that this project report "SIGN LANGUAGE TRANSLATOR" the bonafide work of "ASHISH KUMAR SAH (1613101194)" who carried out the project work under my supervision.

SIGNATURE OF HEAD	**SIGNATURE OF SUPERVISOR**
Dr. MUNISH SHABARWAL, **PhD (Management), PhD (CS)** **Professor & Dean,** **School of Computing Science & Engineering**	**Mr. A. ARUL PRAKASH** **Assistant Professor** **School of Computing Science & Engineering**

TABLE OF CONTENTS

LIST OF TABLES:

LISTS OF FIGURES:

FIGURE TITLE **FIGURE NO.**

ABSTRACT

Every normal human being sees, listens, and reacts to surrounding. There are some unlucky individuals who does not have this important blessing. Such individuals, mainly deaf and dumb, they depend on communication via sign language to interact with others. However, communication with ordinary individuals is a major impairment for them since not every typical people comprehend their sign language. Furthermore, this will cause a problem for the deaf and dumb communities to interact with others, particularly when they attempting to involve into educational, social and work environments. In this project, the objectives are to develop a sign language translation system in order to assist the hearing or speech impaired people to communicate with normal people, and also to test the accuracy of the system in interpreting the sign language. For the methodology, several researches have been done with a specific end goal to choose the best method in gesture recognition. The result for the this experiment shows that total average accuracy for translating alphabets is 95%, numbers is 93.33% and gestures is 78.33%. For the average accuracy for translating all type of gestures is 89%.

The objectives are to develop a sign language translation system in order to assist the hearing or speech impaired people to communicate with normal people and to test the accuracy of the system in interpreting the sign language .

The main objective is to translate sign language to text/speech. The framework provides a helping-hand for speech-impaired to communicate with the rest of the world using sign language. This leads to the elimination of the middle person who generally acts as a medium of translation. This would contain a user-friendly environment for the user by providing speech/text output for a sign gesture input.

INTRODUCTION

(i) Overall description

The mute/deaf individuals have a communication problem dealing with other people. It is hard for such individuals to express what they want to say since sign language is not understandable by everyone. The objective of this project is to develop a Sign Language Translator software that translates the sign language into text that can be read by anyone. This system is called Sign Language Translator and Gesture Recognition. We use OpenCV and Machine Learning to develop a python program that captures the gesture of the hand and interprets these gestures into readable text. This text can be sent to a smart phone or shown in an embedded LCD display/monitor screen. The experimental results that gestures captured is trained by a number of sample gif/picture etc , which measure the positions and the orientation of the fingers. The current version of the system is able to interpret letters with a recognition accuracy of 96%.

(ii) Purpose

The main objective of this project is to design a system that can assists the impaired people to communicate with normal people. This project also aims to meet the following objectives:

i. To develop gesture recognizing system that can recognize sign gesture of Sign Language and translate it into text.

ii. To test the accuracy of the system.

(iii) Motivation and Scope

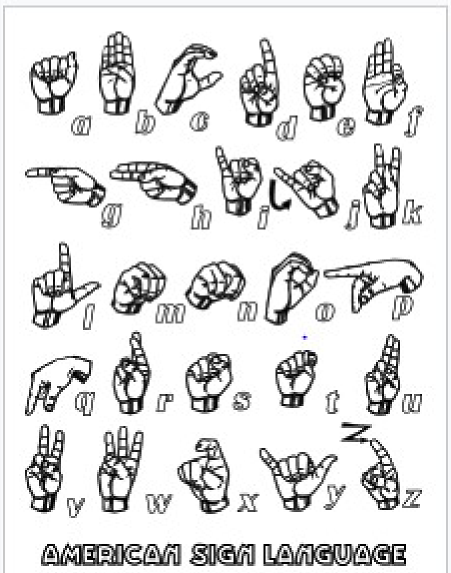
Each ordinary individual sees, tunes in, and responds to encompassing. Notwithstanding, there are some less blessed individuals who are denied of this important blessing. Such individuals, mainly deaf and dumb, they depend on sign language to communicate with others. Statistic shows in India, there are about 120 million people who have disabilities.

The scope for this project are as followed:

i. Sign language recognition system for deaf and mute people.

ii. The sign language used is Indian Sign Language.

iii. This project implement a python program that use OpenCV and Machine Learning.



American manual alphabet, as used in American Sign Language

LITERATURE SURVEY

This section reviews the research on the important elements in developing the sign language recognition device. The first research study focuses on gesture recognition method for detecting the movements of the hand. The second research study discusses the hardware that will be used in this project.

Gesture recognition method

Nowadays, automatic sign language translation systems generally use two approaches, which are data-glove and visual-based approaches. However, new hand gesture recognition method has been introduced, called virtual button. This section will clarify the detail about all methods of gesture recognizing and the comparison between these methods.

(i) Data-glove approach

The data-glove approach utilize a unique assembled electronic glove, which has in- fabricated sensors that utilized to distinguish the hand stance. Most commercial sign language translation systems use the data-glove method, as it simple to acquire data on the bending of finger and 3D orientation of the hand using gloves. The framework require less computational force, and continuous interpretation is much simpler to accomplish.

The data glove is outlined with ten flex sensors, two on every finger. The flex sensors work as variable resistance sensor that change resistance as indicated by the sensor's flexing. These sensors can recognize the bending point of every joint of the fingers and send the information to microcontroller. It is mounted in the outer layer of the data glove, from the association joints of fingers and palm to fingertips.

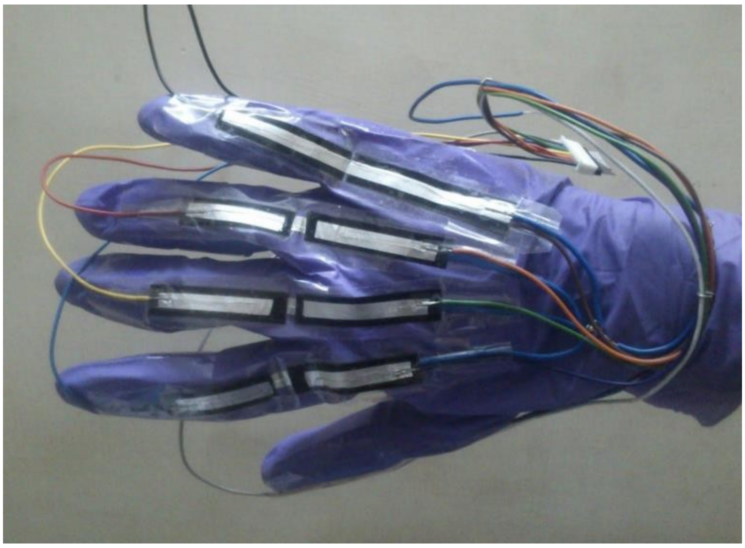


Figure : Data glove with flex sensors.

(ii) Visual-based approach

With late progression in PC and data innovation, there has been an expanded regard for visual-based methodology. Images of the signer is captured by a camera and video processing is done to perform acknowledgment of the sign language. Contrasted with data glove approach, the fundamental advantage of visual-based methodology is the adaptability of the framework.

The recognition of facial expression and head movements additionally can be incorporated to the framework and perform lip-perusing. This system can be separated into two strategy, which are utilization hand crafted shading gloves and in light of skin-colour recognition.

For the specially crafted glove, the signer is furnished with colour-coded gloves. The colour will give the extraction of information from the images of the signer through colour segmentation. These gloves are essentially normal pair of glove with particular shading on every fingertip and palm. Some way or another, these gloves are less expensive contrasted with electronic data gloves. This system is use insignificant equipment by utilizing just essential webcam and basic glove.

Webcam is used to acquire images from the signer in type of still images and video streams in RGB (red-green-blue) shading.

For the recognition based on skin-colour, the framework require just a camera to catch the pictures of the signer for the normal collaboration in the middle of human and computer and no additional gadgets are needed. It is turn out to be more common and helpful for constant applications. This system utilize an uncovered hand to concentrate information required for recognition, and it is simple, and the can user directly communicate with the system. In order to track the position of hand, the skin colour region will be fragmented utilizing colour threshold technique, then the region of interest can be determined.

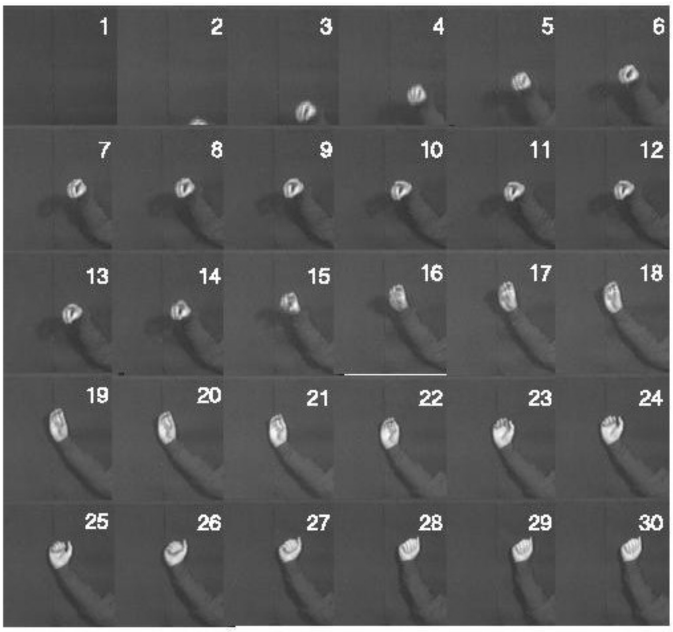


Figure: Image acquisition.

PROPOSED MODEL

The project will be structured into 3 distinct functional blocks, Data Processing, Training, Classify Gesture. The block diagram is simplified in detail to abstract some of the minutiae:

• **Data Processing**: The load data.py script contains functions to load the Raw Image Data and save the image data as numpy arrays into file storage. The process data.py script will load the image data from data.npy and preprocess the image by resizing/rescaling the image, and applying filters and ZCA whitening to enhance features. During training the processed image data was split into training, validation, and testing data and written to storage. Training also involves a load dataset.py script that loads the relevant data split into a Dataset class. For use of the trained model in classifying gestures, an individual image is loaded and processed from the filesystem.

• **Training**: The training loop for the model is contained in train model.py. The model is trained with hyperparameters obtained from a config file that lists the learning rate, batch size, image filtering, and number of epochs. The configuration used to train the model is saved along with the model architecture for future evaluation and tweaking for improved results. Within the training loop, the training and validation datasets are loaded as Dataloaders and the model is trained using Adam Optimizer with Cross Entropy Loss. The model is evaluated every epoch on the validation set and the model with best validation accuracy is saved to storage for further evaluation and use. Upon finishing training, the training and validation error and loss is saved to the disk, along with a plot of error and loss over training.

- **<u>Classify Gesture</u>**: After a model has been trained, it can be used to classify a new ASL gesture that is available as a file on the filesystem. The user inputs the filepath of the gesture image and the test data.py script will pass the filepath to process data.py to load and preprocess the file the same way as the model has been trained.

EXISTING SYSTEM

The ability to track a person's movements and determine what gestures they may be performing can be achieved through various tools. There were several attempts to resolve this problem and there were a large amount of research done in image/video based gesture recognition consequently there was some variation within the tools and environments used between implementations.

SignAloud

SignAloud is a technology that incorporates a pair of gloves made by a group of students at University of Washington that transliterate American Sign Language (ASL) into English. In February 2015 Thomas Pryor, a hearing student from the University of Washington, created the first prototype for this device at Hack Arizona, a hackathon at the University of Arizona. Pryor continued to develop the invention and in October 2015, Pryor brought Navid Azodi onto the SignAloud project for marketing and help with public relations. Azodi has a rich background and involvement in business administration, while Pryor has a wealth of experience in engineering. In May 2016, the duo told NPR that they are working more closely with people who use ASL so that they can better understand their audience and tailor their product to the needs of these people rather than the assumed needs. However, no further versions have been released since then. The invention was one of seven to win the Lemelson-MIT Student Prize, which seeks to award and applaud young inventors. Their invention fell under the "Use it!" category of the award which includes technological advances to existing products. They were awarded $10,000.

The gloves have sensors that track the users hand movements and then send the data to a computer system via Bluetooth. The computer system analyzes the data and matches it to English words, which are then spoken aloud by a digital voice.

The gloves do not have capability for written English input to glove movement output or the ability to hear language and then sign it to a deaf person, which means they do not provide reciprocal communication. The device also does not incorporate facial expressions and other nonmanual markers of sign languages, which may alter the actual interpretation from ASL.

ProDeaf

ProDeaf (WebLibras) is a computer software that can translate both text and voice into Portuguese Libras (Portuguese Sign Language) "with the goal of improving communication between the deaf and hearing."[14] There is currently a beta edition in production for American Sign Language as well. The original team began the project in 2010 with a combination of experts including linguists, designers, programmers, and translators, both hearing and deaf. The team originated at Federal University of Pernambuco (UFPE) from a group of students involved in a computer science project. The group had a deaf team member who had difficulty communicating with the rest of the group. In order to complete the project and help the teammate communicate, the group created Proativa Soluções and have been moving forward ever since. The current beta version in American Sign Language is very limited. For example, there is a dictionary section and the only word under the letter 'j' is 'jump'. The last update of the app was in June 2016, but ProDeaf has been featured in over 400 stories across the country's most popular media outlets.

The application cannot read sign language and turn it into word or text, so it only serves as a one-way communication. Additionally, the user cannot sign to the app and receive an English translation in any form, as English is still in the beta edition.

IMPLEMENTATION

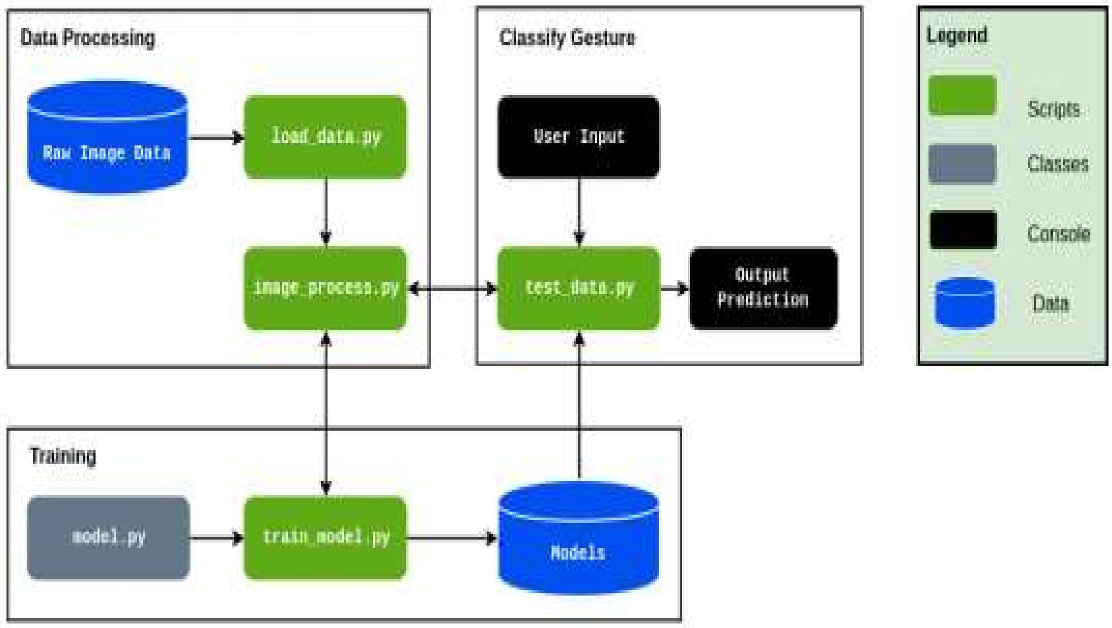


Figure 3 : Block Digram of Software

A. Image capturing through webcam

Client is made to produce motions before the camera, utilizing exclusively a webcam as portrayed in the paper by Rautauray and Agarwal and handed-off to the program for additional preparing. The camera should be fixed, and brightening gradually changing. Ongoing imperatives are being forced for a cautious plan of the preparing framework.

B. Segmentation

Separation of external and unnecessary factors from the image captured forms the crux of this section. Any sort of background disturbance or components of the image not required for processing are to be separated from the image of the gesture.

The unnecessary information is first removed. In particular, a background suppression procedure has been performed in the HSV colour space, in which the scene can be modelled discarding illumination variations .Thus focusing the attention on areas corresponding to human skin colour.

C. Translation Process

To beat the problem identified with equipment sensors in the Data glove innovation as proposed by Liang and Ouhyoung, I utilize the picture produced by the webcam. When the picture is taken from the foundation and other unimportant issue; the forms in the motion are estimated by the shape framed by the hand [5]. The database built contains all the endorsed and acknowledged motions by the ASL show. The form concluded from the picture is coordinated to the significant sign in the database.

1. Creating a gesture:

1. First set your hand histogram. To do so type the order given underneath and adhere to the guidelines beneath. Run file set_hand_hist.py

- A windows "Set hand histogram" will show up.
- "Set hand histogram" will have 50 squares (5x10).
- Put your submit those squares. Ensure your hand covers all the squares.
- Press 'c'. 1 other window will show up "Thresh".
- On squeezing 'c' just white patches relating to the pieces of the picture which has your skin shading ought to show up on the "Thresh" window.
- Make sure all the squares are covered by your hand.
- In case you are not fruitful at that point move your hand a little bit and press 'c' once more. Repeat this until you get a decent histogram.
- After you get a decent histogram press 's' to save the histogram.

2. I as of now have included 44 (0-43) gestures. To make your own signals or supplant my motions do the accompanying. It is finished by the order given beneath. On beginning executing this program, you should enter the gesture number and motion name/content. At that point an OpenCV window called "Capturing gestures" which will show up. In the webcam feed you will see a green window (inside which you should do your gesture) and a counter that checks the number of pictures stored. Run file create_gestures.py to create new gesture for the framework.

3. Press 'c' when you are prepared with your gesture. Catching gestures will start following a couple of moments. Move your hand a tad to a great extent. You can pause catching by squeezing 'c' and resume it by squeezing 'c'. Catching resumes following a couple of second. After the counter arrives at 1200 the window will close naturally.

4. Subsequent to capturing all the gestures you can flip the pictures. For flipping the images run the flip_images.py.

5. When you are finished including new gestures run the load_images.py document once. You don't have to run this document again until and except if you include another gesture.

2.Displaying all gestures

To see all the gestures that are stored in 'gestures/' folder run this file display_all_gestures.py.

3.Training the model

Training can be done with either Tensorflow or Keras. I trained my model using Keras . To train model run file cnn_keras.py.

You do not need to retrain your model every time. In case you added or removed a gesture then you need to retrain it.

4. Testing the model

For recognition to start run the recognize_gesture.py file.

You will have a small green box inside which you need to do you gestures

And see the output of your hand gesture in the console output.

RESULTS

Model Training Report

(8800, 44)
Model: "sequential_2"

Layer (type)	Output Shape	Param #
conv2d_4 (Conv2D)	(None, 49, 49, 16)	80
max_pooling2d_4 (MaxPooling2	(None, 25, 25, 16)	0
conv2d_5 (Conv2D)	(None, 23, 23, 32)	4640
max_pooling2d_5 (MaxPooling2	(None, 8, 8, 32)	0
conv2d_6 (Conv2D)	(None, 4, 4, 64)	51264
max_pooling2d_6 (MaxPooling2	(None, 1, 1, 64)	0
flatten_2 (Flatten)	(None, 64)	0
dense_3 (Dense)	(None, 128)	8320
dropout_2 (Dropout)	(None, 128)	0
dense_4 (Dense)	(None, 44)	5676

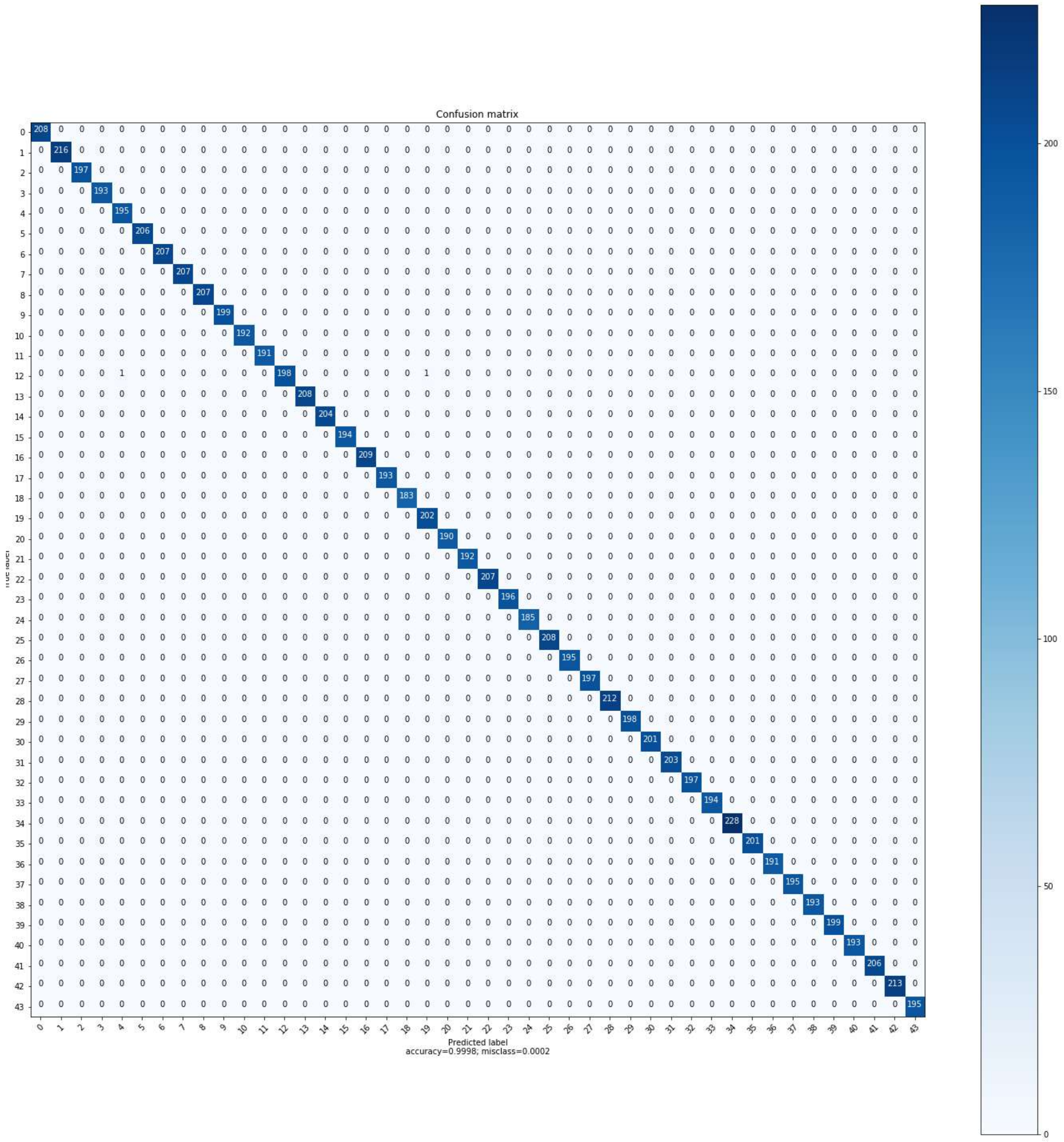
Total params: 69,980
Trainable params: 69,980
Non-trainable params: 0

Train on 88000 samples, validate on 8800 samples
Epoch 1/20
88000/88000 [==============================] - 129s 1ms/step - loss: 3.0812 - accuracy: 0.2897 - val_loss: 0.4699 - val_accuracy: 0.9053
Epoch 2/20
88000/88000 [==============================] - 141s 2ms/step - loss: 0.3587 - accuracy: 0.8929 - val_loss: 0.0470 - val_accuracy: 0.9875
Epoch 3/20
88000/88000 [==============================] - 154s 2ms/step - loss: 0.1123 - accuracy: 0.9655 - val_loss: 0.0185 - val_accuracy: 0.9953
Epoch 4/20
88000/88000 [==============================] - 131s 1ms/step - loss: 0.0636 - accuracy: 0.9810 - val_loss: 0.0103 - val_accuracy: 0.9974
Epoch 5/20
88000/88000 [==============================] - 128s 1ms/step - loss: 0.0454 - accuracy: 0.9863 - val_loss: 0.0078 - val_accuracy: 0.9977

```
Epoch 6/20
88000/88000 [==============================] - 122s 1ms/step - loss: 0.0337 - accuracy: 0.9896 - val_loss: 0.0046 - val_accuracy: 0.9991
Epoch 7/20
88000/88000 [==============================] - 140s 2ms/step - loss: 0.0262 - accuracy: 0.9922 - val_loss: 0.0043 - val_accuracy: 0.9994
Epoch 8/20
88000/88000 [==============================] - 129s 1ms/step - loss: 0.0219 - accuracy: 0.9933 - val_loss: 0.0033 - val_accuracy: 0.9992

Epoch 9/20
88000/88000 [==============================] - 138s 2ms/step - loss: 0.0181 - accuracy: 0.9947 - val_loss: 0.0026 - val_accuracy: 0.9995
Epoch 10/20
88000/88000 [==============================] - 140s 2ms/step - loss: 0.0155 - accuracy: 0.9953 - val_loss: 0.0025 - val_accuracy: 0.9994
Epoch 11/20
88000/88000 [==============================] - 138s 2ms/step - loss: 0.0144 - accuracy: 0.9954 - val_loss: 0.0020 - val_accuracy: 0.9995
Epoch 12/20
88000/88000 [==============================] - 132s 1ms/step - loss: 0.0127 - accuracy: 0.9962 - val_loss: 0.0020 - val_accuracy: 0.9994
Epoch 13/20
88000/88000 [==============================] - 132s 2ms/step - loss: 0.0120 - accuracy: 0.9967 - val_loss: 0.0019 - val_accuracy: 0.9995
Epoch 14/20
88000/88000 [==============================] - 141s 2ms/step - loss: 0.0103 - accuracy: 0.9969 - val_loss: 0.0017 - val_accuracy: 0.9995
Epoch 15/20
88000/88000 [==============================] - 128s 1ms/step - loss: 0.0092 - accuracy: 0.9972 - val_loss: 0.0013 - val_accuracy: 0.9997
Epoch 16/20
88000/88000 [==============================] - 124s 1ms/step - loss: 0.0092 - accuracy: 0.9973 - val_loss: 0.0015 - val_accuracy: 0.9997
Epoch 17/20
88000/88000 [==============================] - 133s 2ms/step - loss: 0.0076 - accuracy: 0.9977 - val_loss: 0.0011 - val_accuracy: 0.9998
Epoch 18/20
88000/88000 [==============================] - 130s 1ms/step - loss: 0.0073 - accuracy: 0.9979 - val_loss: 0.0013 - val_accuracy: 0.9998
Epoch 19/20
88000/88000 [==============================] - 128s 1ms/step - loss: 0.0070 - accuracy: 0.9979 - val_loss: 9.8177e-04 - val_accuracy: 0.9999
Epoch 20/20
88000/88000 [==============================] - 127s 1ms/step - loss: 0.0060 - accuracy: 0.9983 - val_loss: 0.0011 - val_accuracy: 0.9997
```

Classification reports about the model



we get the confusion matrix, f scores, precision and recall for the predictions by the model.

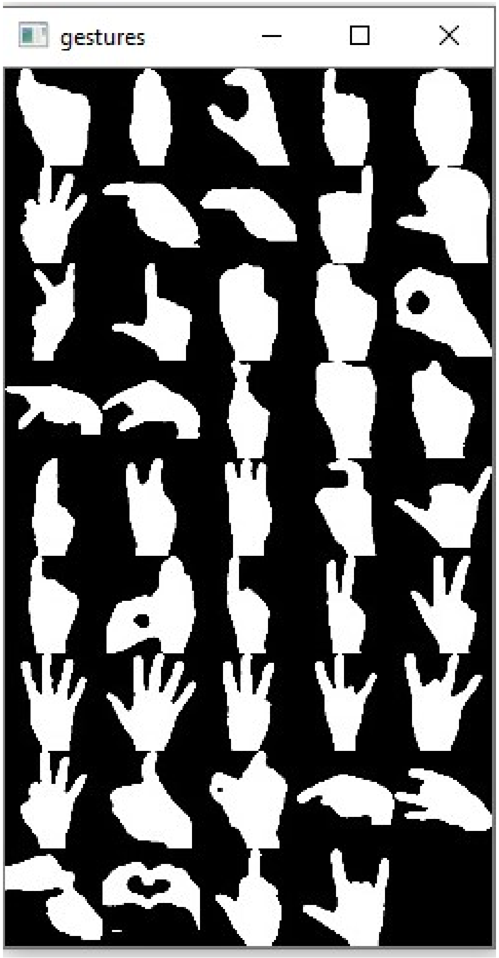
Time taken to predict 8800 test images is 6s

Average prediction time: 0.000742s

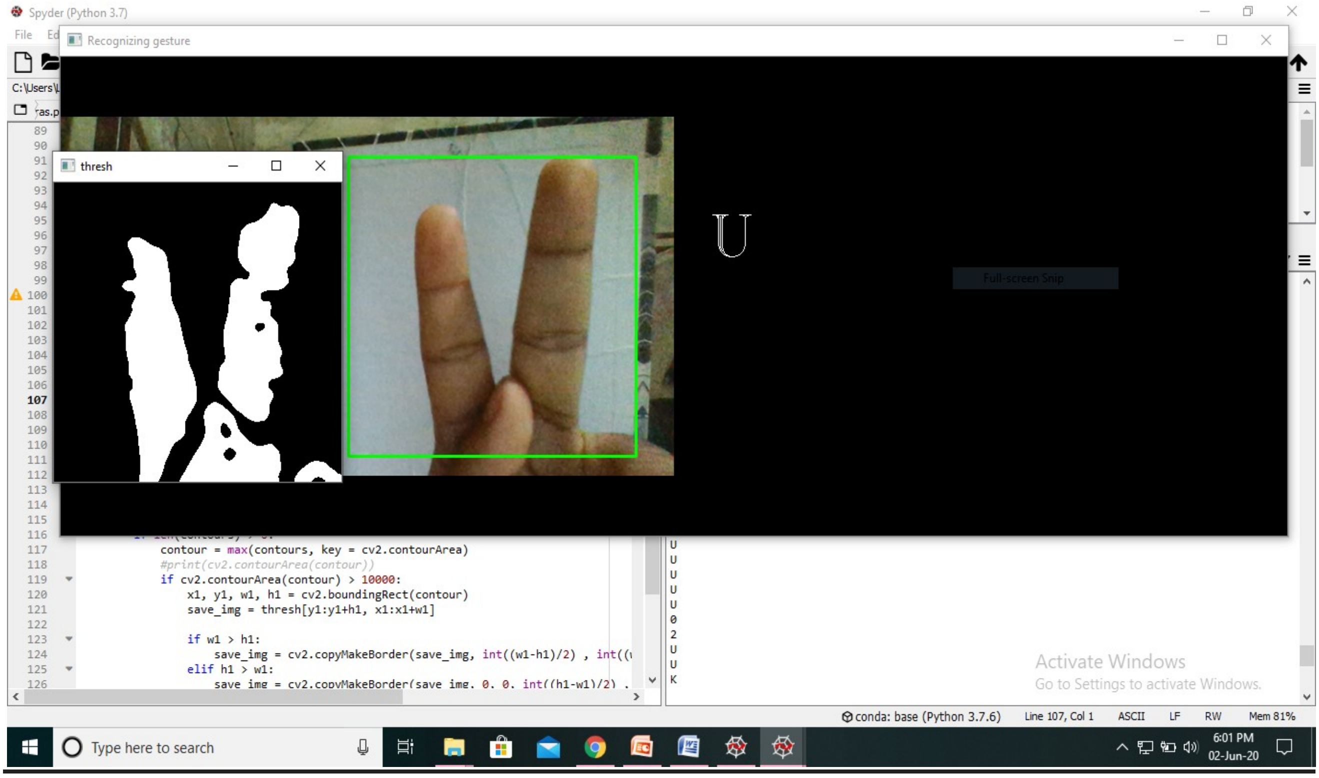
Classification Report
----

	precision	recall	f1-score	support
0	1.00	1.00	1.00	208
1	1.00	1.00	1.00	216
2	1.00	1.00	1.00	197
3	1.00	1.00	1.00	193
4	0.99	1.00	1.00	195
5	1.00	1.00	1.00	206
6	1.00	1.00	1.00	207
7	1.00	1.00	1.00	207
8	1.00	1.00	1.00	207
9	1.00	1.00	1.00	199
10	1.00	1.00	1.00	192
11	1.00	1.00	1.00	191
12	1.00	0.99	0.99	200
13	1.00	1.00	1.00	208
14	1.00	1.00	1.00	204
15	1.00	1.00	1.00	194
16	1.00	1.00	1.00	209
17	1.00	1.00	1.00	193
18	1.00	1.00	1.00	183
19	1.00	1.00	1.00	202
20	1.00	1.00	1.00	190
21	1.00	1.00	1.00	192
22	1.00	1.00	1.00	207
23	1.00	1.00	1.00	196
24	1.00	1.00	1.00	185
25	1.00	1.00	1.00	208
26	1.00	1.00	1.00	195
27	1.00	1.00	1.00	197
28	1.00	1.00	1.00	212
29	1.00	1.00	1.00	198
30	1.00	1.00	1.00	201
31	1.00	1.00	1.00	203
32	1.00	1.00	1.00	197
33	1.00	1.00	1.00	194
34	1.00	1.00	1.00	228
35	1.00	1.00	1.00	201
36	1.00	1.00	1.00	191
37	1.00	1.00	1.00	195
38	1.00	1.00	1.00	193
39	1.00	1.00	1.00	199
40	1.00	1.00	1.00	193
41	1.00	1.00	1.00	206
42	1.00	1.00	1.00	213
43	1.00	1.00	1.00	195
accuracy			1.00	8800
macro avg	1.00	1.00	1.00	8800
weighted avg	1.00	1.00	1.00	8800

OUTPUT



All 44 gestures stored and trained in the system



<u>Output : for the symbol in the green box output is displayed in the console as “U”</u>

FUTUER ENHANCEMENT

1. The system can be extended to incorporate the knowledge of facial expressions and body language too so that there is a complete understanding of the context and tone of the input speech.

2. A mobile and web based version of the application will increase the reach to more people.

3. Integrating hand gesture recognition system using computer vision for establishing 2-way communication system.

REFERENCES

[1] Scientific World Journal Volume 2014 (2014), Article ID 267872
http://dx.doi.org/10.1155/2014/267872.

[2] "The Cognitive, Psychological and Cultural Impact of Communication Barrier on Deaf Adults". In: Journal of Communication Disorders, Deaf Studies Hearing Aids 4 (2 2016).
doi: 10.4172/23754427.1000164.

[3] Akash. ASL Alphabet. url: https://www.kaggle.com/grassknoted/asl-alphabet. (accessed: 24.10.2018).

[4] Vivek Bheda and Dianna Radpour. "Using Deep Convolutional Networks for Gesture Recognition in American Sign Language". In: CoRR abs/1710.06836 (2017). arXiv: 1710.06836.
url: http://arxiv.org/abs/1710. 06836.

[5] en.wikipedia.org, https://en.wikipedia.org/wiki/Fingerspelling

[6] en.wikipedia.org, https://en.wikipedia.org/wiki/Machine_translation_of_sign_languages.

[7] Siddharth S. Rautaray, Anupam Agrawal," Real time hand gesture recognition system for dynamic applications," International Journal of UbiComp (IJU), Indian Institute of Information Technology Allahabad, India, Vol.3, No.1, January 2012.

[8] Rung-Huei Liang, Ming Ouhyoung," A Real-time Continuous Alphabetic Sign Language to Speech Conversion VR System," Communications & Multimedia Lab., Computer Science and Information Engineering Dept., National Taiwan University, Taipei, Taiwan.

[9] Sahib Singh1,Dr. Vijay Kumar Banga,"Gesture control algorithm for personal computers," ISSN: 2319 – 1163,Volume: 2 Issue: 5, Department of Electronics and Communication Engineering,Punjab, India.

[10] Link for files used in building this framework

https://drive.google.com/drive/folders/1lpyAIU08PwFoohm9TGyvICUupEPWXV9I?usp=sharing