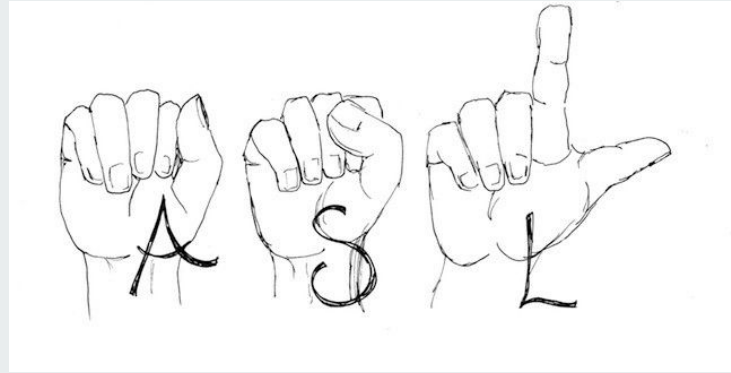


Sign Language To Text Conversion For Dumb and Deaf



Submitted By -
Parul Mahajan
19103078

Mentored By -
Dr. Kunwar Pal
Assistant Professor

Deaf and Dumb Alphabet





Introduction

Over 5% of the world's population – or 430 million people – require rehabilitation to address their 'disabling' hearing loss (432 million adults and 34 million children). It is estimated that by 2050 over 700 million people – or one in every ten people – will have disabling hearing loss and the same such stats can be shown about dumb people. And communication is an important part of our life these people can take all the help they want.

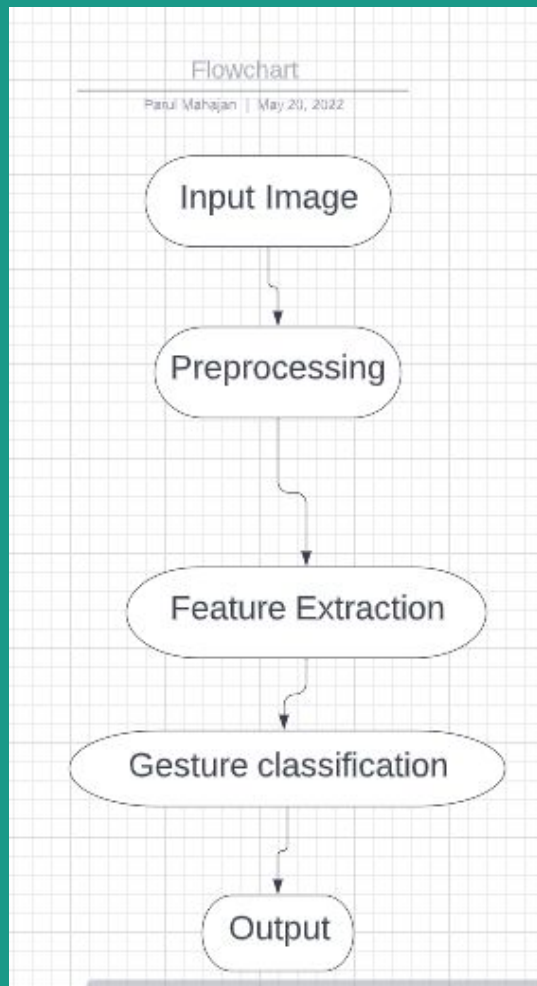
This project aims to create a computer application and train a model which when shown a real time video of hand gestures of American Sign Language shows the output for that particular sign in text format on the screen.



Sign language is a visual language and consists of 3 major components:

| Fingerspelling | Word level sign vocabulary | Non-manual features |
|--|---|---|
| Used to spell words letter by letter . | Used for the majority of communication. | Facial expressions and tongue, mouth and body position. |

Work-Flow



Dataset Generation

Took 800 frames for training and 200 for testing with RGB values then used gaussian filter on gray scale image to generate black and white image for ROI of hand gesture.

Gray Scale Image



**Image Post Gaussian
Blur**





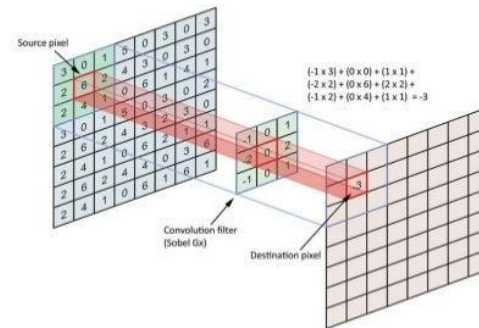
Gesture Classification

- Apply a gaussian blur filter and threshold to the frame taken with OpenCV to get the processed image after feature extraction.
- This processed image is passed to the CNN model for prediction and if a letter is detected for more than 50 frames then the letter is printed and taken into consideration for forming the word.
- Space between the words is considered using the blank symbol.
- In layer 2 we detect various sets of symbols which show similar results on getting detected. We then classify between those sets using classifiers made for those sets only.

Convolutional Neural Networks

- CNNs consist of multiple convolutional layers each layer containing numerous “filters” which perform feature extraction
- Initially these “filters” are random and by training, the feature extraction gets better by better.
- It’s primarily used for image classification.

Convolutional Filter





CNN


It primarily has 4 layers

- Pooling layer
- Fully Connected Layer
- Activation Layer
- Final Output Layer



ASL CNN MODEL

- The input picture has resolution of 128x128 pixels. It is first processed in the first convolutional layer using 32 filter weights (3x3 pixels each). This will result in a 126X126 pixel image, one for each Filter-weights.
- The pictures are downsampled using max pooling of 2x2 i.e we keep the highest value in the 2x2 square of array. Therefore, our picture is downsampled to 63x63 pixels.
- Now, these 63 x 63 from the output of the first pooling layer is served as an nput to the second convolutional layer.It is processed in the second convolutional layer using 32 filter weights (3x3 pixels each).This will result in a 60 x 60 pixel image
- The resulting images are downsampled again using max pool of 2x2 and is reduced to 30 x 30 resolution of images.

- 
- 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 of $30 \times 30 \times 32 = 28800$ values. The input to this layer is an array of 28800 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
 - Now the output from the 1st Densely Connected Layer are used as an input to a fully connected layer with 96 neurons.
 - The output of the 2nd Densely Connected Layer serves as an input for the final layer which will have the number of neurons as the number of classes we are classifying (alphabets + blank symbol)

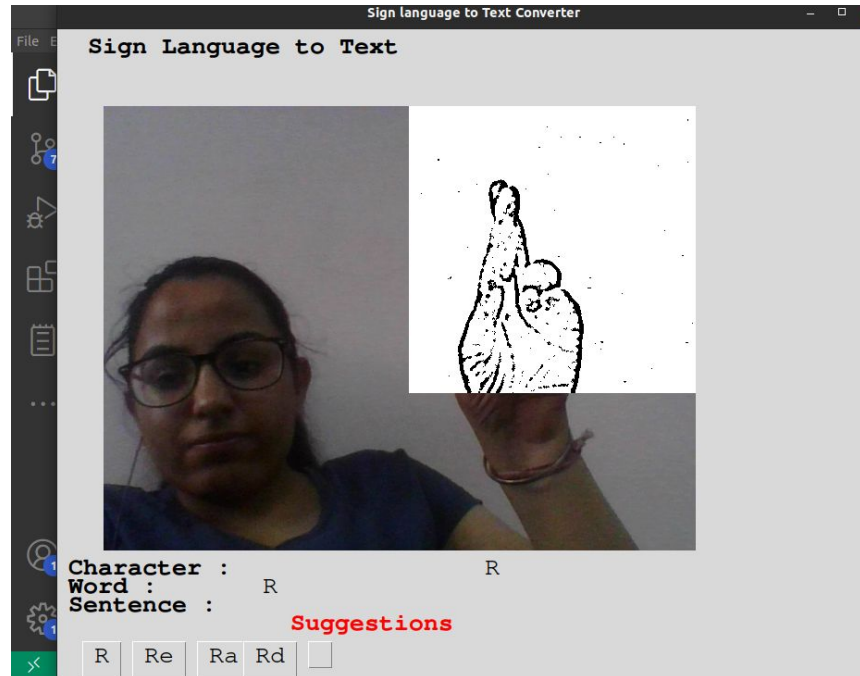


Finger Spelling

- Whenever the count of a letter detected exceeds a specific value and no other letter is close to it by a threshold we print the letter and add it to the current string(In our code we kept the value as 50 and the difference threshold as 20).
- Whenever the count of a blank(plain background) detected exceeds a specific value and if the current buffer is empty no spaces are detected.
- In another case, it predicts the end of a word by printing space, and the current gets appended to the sentence below.
- A python library hunspell is used which gives alternatives to the sentence being formed and the user can select a word to append it to the current sentence. This helps in reducing mistakes committed in spelling and assists in predicting complex words.

Results

Few of the predictions can we be shown below:

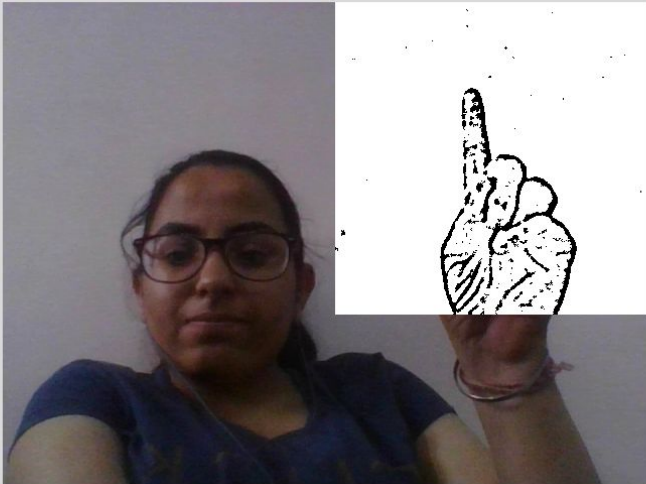


Activities Tk May 19 18:18

Sign language to Text Converter

File Edit

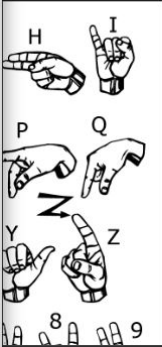
Sign Language to Text



Character : D
Word :
Sentence :

Suggestions

D Di Dr Dy



```
python3
```

once-critical operations: AVX2

flow with the appropriate compil

1280x960 91.76KB Go Live



Software Requirements

- Python 3
- Tensorflow 1.11.0
- OpenCV 3.4.3.18'
- NumPy 1.15.3
- Matplotlib 3.0.0
- Hunspell 2.0.2
- Keras 2.2.1



Conclusion

- In this report, a functional real time vision based american sign language recognition for D&M people have been developed for asl alphabets.
- We achieved an accuracy of 84.7% on our dataset.
- Prediction has been improved after implementing two layers of algorithms in which we verify and predict symbols which are more similar to each other.



Limitations of our model

- Accuracy is just 84.7% which can be improved.
- Model works better in plain background only. Gives wrong answer even in small noise.
- Model is having issues predicting alike signs some of them are:
 - For D : R and U
 - For U : D and R
 - For I : T, D, K and I
 - For S : M and N



Future Scope

- We can add SIGN-TEXT-SPEECH feature also so that we can make it more relevant for people with disabilities.
- We can achieve higher accuracy even in case of complex backgrounds by trying out various background subtraction algorithms.
- We can also improve the preprocessing to predict gestures in low light conditions with a higher accuracy

THANK YOU!!
