JETIR.ORG ISSN: 2349-5162 | ESTD Year : 2014 | Monthly Issue

JOURNAL OF EMERGING TECHNOLOGIES AND INNOVATIVE RESEARCH (JETIR)

An International Scholarly Open Access, Peer-reviewed, Refereed Journal

Sign Language Recognition Android App using Convolution Neural Network

Hritika Dhasal[1, a], Vaishali Sonawane[1, b], Yash Bari[1, c], Sharvari Raut[1, d], A. V. Sagare[2, e]

[1]Student, CSE Department, PCET's - Nutan College Of Engineering And Research, Talegaon Dabhade, Pune, India
[2]Assistant Professor, CSE Department, PCET's - Nutan College Of Engineering And Research, Talegaon Dabhade, Pune, India
[a]dhasalhritika1@gmail.com
[b]vaishali.sonawane0018@gmail.com
[c]yashcb1234@gmail.com
[d]sharuraut7official@gmail.com
[e]jeet.sagare@gmail.com

Abstract- India rank 2nd in the world population, which also means the number of mute or deaf people in India can also be. According to a World Health Organization report - 'India has 63 million people who are either suffering from complete or partial deafness. This is equal to 6.3% of the total population of India. This part of India's population will remain disconnected without technology. In 2017, India only had 250 certified interpreters for sign language available. Therefore, there is a need for cost-effective and easily available solutions. Due to advancements in the latest technology both hardware and software solutions are available like leap motion devices and translation through computer vision. Our project is about sign language translation to English. We have used custom data of approximately 500 images for each item ((A-Y), (0-9), and (other words)) for training and testing data which gives an accuracy of 94%. We have acquired this accuracy through a convolutional neural network (CNN). The dataset is prepared using a laptop/mobile webcam. The suggested solution is able to detect gestures in real-time accurately and form words and sentences. Further, we have converted our model into an android application using a java programming language with real-time word-formation.

Keywords- CNN, Sign language recognition, mobile application, mobile sign language app, computer vision, gesture recognition, Edge Detection

I. Introduction

Technology has helped us solve very critical problems too. Research is carried out to help mute people communicate commonly with us. In summary, the vision-based approach and sensory gloves are two commonly used methods for sign language detection [1]. We have implemented sign language project using vision-based approach because it is cost effective and doesn't require the user to have gloves and other sensory devices continuously on the body.

A. Artifical Neural Network

ANN has overcome complicated applications and nonlinearity that was difficult to model using the standard model. Through training and validation of knowledge, ANN performance is increased. ANN model merely requires user input and output data for training [2]. ANN isn't suggested for the image dataset. As an example – consider you've got a giant RGB image of a dog with (1920 x 1080 x 3) resolution. Then, the primary layer neurons = 1920 x 1080 x 3 = 6220800 which approximately is 6 million. Now for the hidden layer neuron, let's consider it 5 million then weights between input and the hidden layer in ANN = 6 x 5 = 35 million. The deep neural network can have more than 5 hidden layers and this will go up to 500 million to 1 billion weights to compute. a traditional computer cannot manage such a lot of computation. Disadvantages of using ANN for this project:

- It will require too much computation
- Similar pixels are treated same
- It is sensitive to the location of an object in an image.

B. Convolution Neural Network

Convolution Neural Network is at the forefront of image processing. As Artificial Neural (ANN) networks require a longer time to process image data. CNN algorithm is based on feature learning. In a large image like a dog, the CNN algorithm will learn the smallest features of a dog's eyes, ears, face, legs, body, etc., and cover everything under a group. (Figure 1) demonstrates the function of a deep neural convolution network. The extracted features include a fully integrated layer, which integrates the activation function.

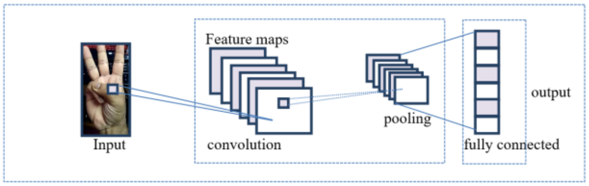


Fig. 1. Architecture of Convolutional Neural Network

C. Sign language Recognition App

Our app is design to recognize sign language gestures like alphabets A to Y, numbers 0 to 9 and few common words like hello, thank you, welcome, I love you. etc. Which are used in everyday life. User can also form own sentences. We have built custom data for the application which has 500 images each in total approximately 20000 images and it is made open source for others to contribute. The interface of the app is very user-friendly. The application is made free to download and install.

The technical aspects of the application include the following details –

- Version: 1.0
- The Platform: Android v7+
- Language: java
- Algorithm: CNN model

II. Literature Review

According to [1] from 2001 to 2021 total 1312 journal paper were published on the topic sign language recognition. There has been research done on both vision-based sign language recognition and by using sensory gloves.

The paper DeepASLR has added new American sign language dataset. It has a dataset of 1,04,000 images and while comparing with other 2 dataset using CNN model it gave nearby same accuracy of 99% [3]. In Real time Indian sign language recognition system, Indian dataset was used to detect sign language in real time using CNN [4] similar thing was implemented in [6].

Sign language can also be recognized using video as input. In this video is cut into frames and a useful meaning is formed out of it. This could be used for complete sentence formation. Refer [7].

SL projects also use sensors. In paper [8], the wearable sensor is used to detect language and comparisons are made between the ANN, SVM, and HMM models. "Test results show that using ANN provides the highest accuracy in recognizing ASL words, compared to other machine learning techniques".

III. Proposed Method

This project is divided into two parts – Making of trained ML model and developing an android application.

A. Making of Model

For making a trained model follow these steps – Image acquisition, Image Pre-processing, Image segmentation, Feature Extraction, and Classification shown in Figure 2.

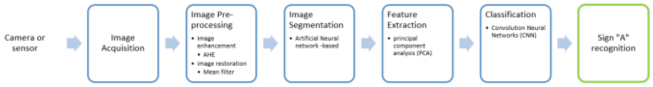


Fig. 2. Flowchart for sign language recognition model

1) Image Acquisition

The camera or webcam is the most used by most researchers because it provides the best and most natural interaction of a human-computer without additional devices, unlike data gloves [1]. In our project, we have captured a custom dataset using an android phone camera. It includes almost 20000 images of A to Y alphabets, 0 to 9 digits, and a few everyday words. In the dataset, there are many variations in lighting, distance, skin, invalid dataset, etc. which will help train the model to cover almost all corner test cases.

2) Image Pre-processing

Pre-processing techniques are applied to an input image to remove unwanted noise and also enhance the quality [1]. The image of the dataset was enhanced using Adaptive Histogram Equalization (AHE).

And, the image was restored using the Mean Filter. The equation of Mean filter is:

$$A[i,j] = \frac{1}{M}\sum_{k=i=1}^{i}\sum_{i=j=1}^{j+1} fk,l$$

Fig. 3. Equation of Mean Filter

3) Image Segmentation

Image segmentation is the process of partitioning a picture into meaningful regions called segments[1]. The pictures are segmented to get the region of interest. An artificial neural network-based approach was implemented for the project for image segmentation.

4) Feature Extraction

Feature extraction may be a technique accustomed to obtaining the foremost relevant features from the input image [1]. It aims at finding the foremost distinctive features within the acquired image. Principal Component Analysis (PCA) was used for feature extraction.

5) Classification

A convolution Neural Network is best for image classification. Consider figure 1, The input image is to obtain the hand region using color segmentation [1]. The proposed CNN model consists of the input layer, convolution layers and pooling. An accuracy of 94% was obtained.

B. Android Appliction

By use of the TensorFlow library, we can import our trained model as a model.tflite file in the asset folder of the android application. The application is developed using java language for the front-end as well as backend. The code is first divided into modules such as home page, real-time detection, and translation.

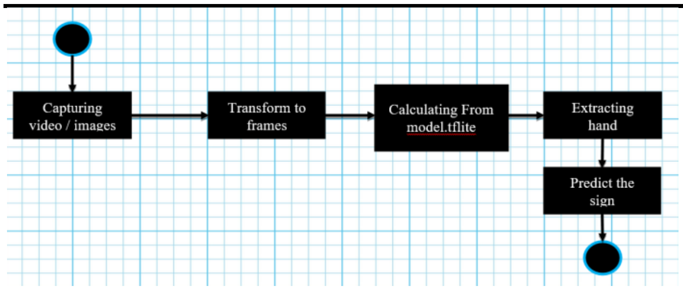


Fig. 4. State diagram of SLR system

IV. Tools & Code Implementation

A. *Software Used*

- Operating System: Windows 10, Android v7+
- IDE: jupyter notebook, Android Studio
- Libraries: TensorFlow, Keras, NumPy, OpenCV

B. *Hardware Used*

- Camera: Android camera
- Device: Android mobile
- GPU: 4GB dedicated
- Ram: 8GB or higher
- Processor: Intel
- HDD: 20GB and above
- Monitor: 15" color monitor

C. *Implementation*

A custom dataset was captured using the android camera for all letters, numbers, and few words. The dataset includes a few duplicates, random shapes, and distance, blank images, images having many background elements, rotation, mirror images, underexposed lighting, and bright light images too. In total, there are approximately 500 images for each. Figure 5 shows a sample dataset of the letter "C".

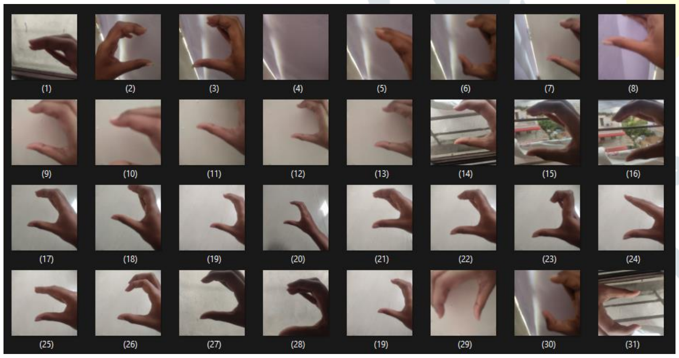
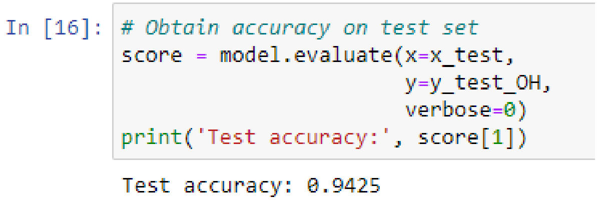


Fig. 5. Custom dataset for letter 'C'

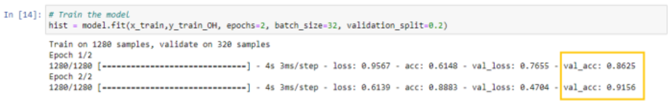
After dataset collection, pre-processing steps were performed to enhance dataset quality. This includes cropping and reducing image size.

For model preparation, the entire dataset was divided into 85% test data and 15% train data. Figure 6 (a) and 6 (b) display the test accuracy of our model. We got an accuracy of 94% by CNN model



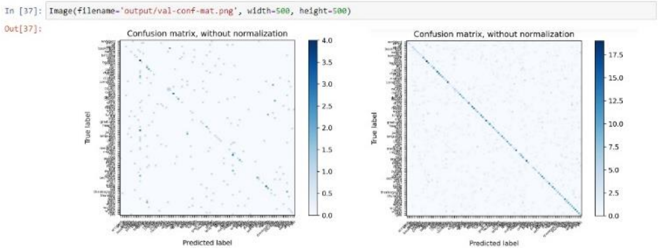
```
In [16]: # Obtain accuracy on test set
         score = model.evaluate(x=x_test,
                                y=y_test_OH,
                                verbose=0)
         print('Test accuracy:', score[1])

         Test accuracy: 0.9425
```

Fig. 6-a Test accuracy after model evaluation



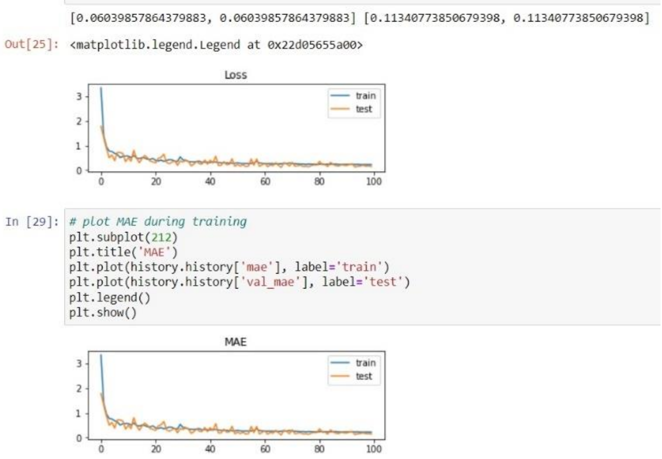
```
In [14]: # Train the model
         hist = model.fit(x_train,y_train_OH, epochs=2, batch_size=32, validation_split=0.2)

         Train on 1280 samples, validate on 320 samples
         Epoch 1/2
         1280/1280 [==============================] - 4s 3ms/step - loss: 0.9567 - acc: 0.6148 - val_loss: 0.7655 - val_acc: 0.8625
         Epoch 2/2
         1280/1280 [==============================] - 4s 3ms/step - loss: 0.6139 - acc: 0.8883 - val_loss: 0.4704 - val_acc: 0.9156
```

Fig. 6-b. The accuracy at each Epoch



```
In [37]: Image(filename='output/val-conf-mat.png', width=500, height=500)
Out[37]:
```

Figure 7-a. The confusion matrix



```
[0.06039857864379883, 0.06039857864379883] [0.11340773850679398, 0.11340773850679398]

Out[25]: <matplotlib.legend.Legend at 0x22d05655a00>
```

```
In [29]: # plot MAE during training
         plt.subplot(212)
         plt.title('MAE')
         plt.plot(history.history['mae'], label='train')
         plt.plot(history.history['val_mae'], label='test')
         plt.legend()
         plt.show()
```

Figure 7-b. loss and MAE graph

The model is export as model.tflite and used in android studio for real time recognition. Figure 8 represent the architecture of sign language recognition

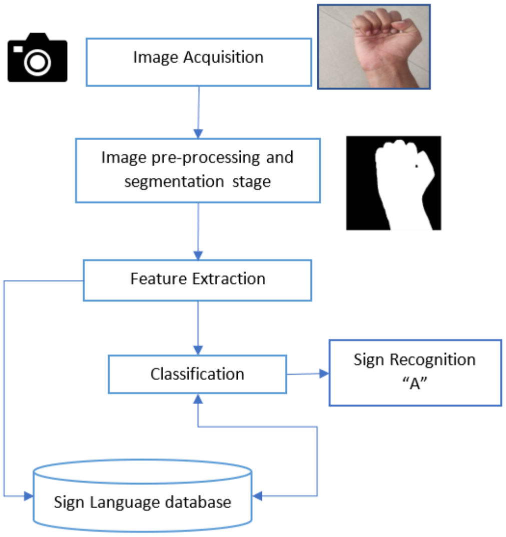


Fig. 8. Architecture of sign language recognition

D. *Output Through App*

The app can detect real time signs and also form sentences /word output it. The figure 9 shows few of the sample gestures recognized. And Figure 10 shows, 'WELCOME' text as output.

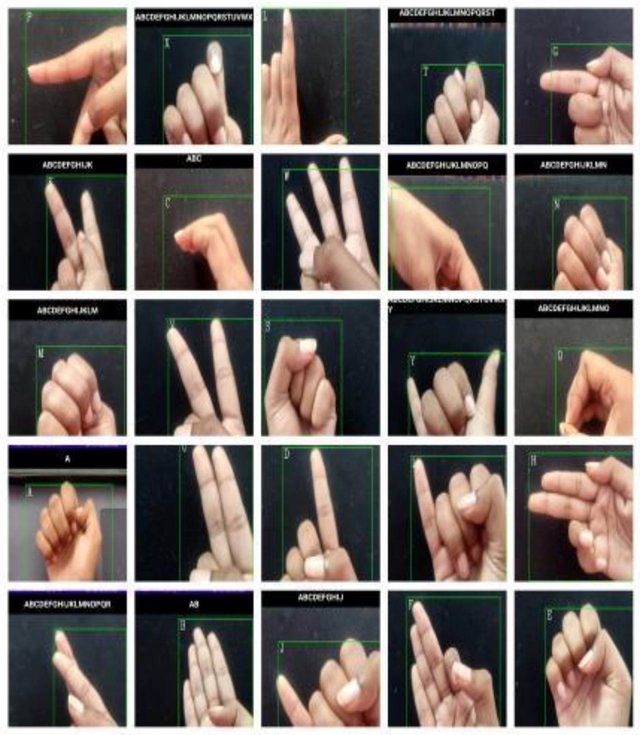


Fig. 9. Output of sign letters

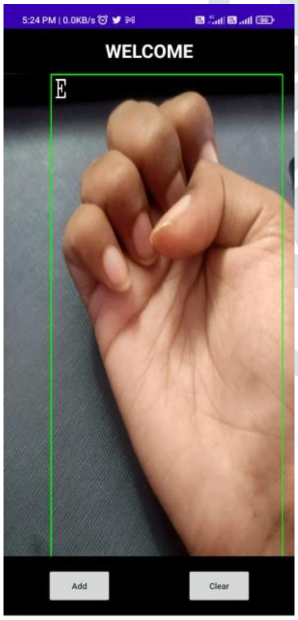


Fig. 10. Output as “WELCOME” text

V. User Interface

The user needs to give camera permission for app to work. Then the user can click on the live detection button to recognize signs or the user can click on sentence formation to add letters and word though gesture to form a complete word or sentence.

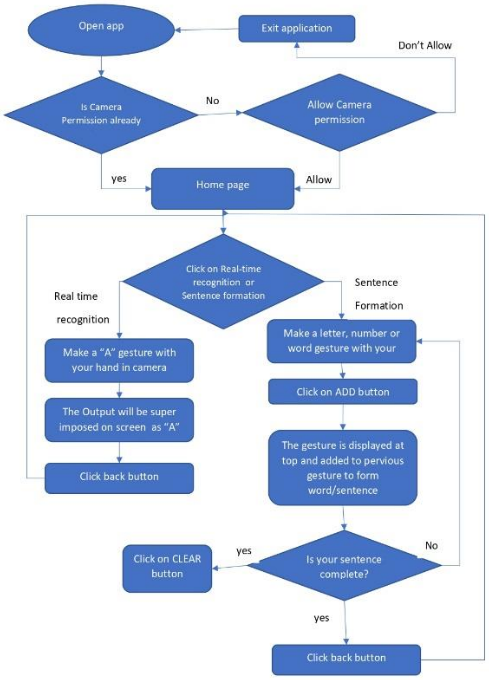

Open app
Exit application
Don’t Allow
Is Camera Permission already
No
Allow Camera permission
yes
Home page
Allow
Click on Real-time recognition or Sentence formation
Real time recognition
Sentence Formation
Make a “A” gesture with your hand in camera
The Output will be super imposed on screen as “A”
Click back button
Make a letter, number or word gesture with your
Click on ADD button
The gesture is displayed at top and added to pervious gesture to form word/sentence
Click on CLEAR button
yes
Is your sentence complete?
No
yes
Click back button

Fig. 11. Sign Language App – User Activity Diagram

VI. Conclusion & Future Scope

Compared to other sign language applications, this app gives a good user-friendly experience, and translate sign language to the English language effectively. The proposed model has an accuracy of 94% which is effective too. A new dataset is used and made available to others to contribute.

In the future, this application can be integrated with assistance to do the simple tasks of searching on the web, opening an application, and sending SMS or WhatsApp messages. We can also increase the dataset and add new words. Overall, this project can help deaf/mute people to communicate with us easily.

References

[1] I.A. Adeyanju, O.O. Bello, M.A. Adegboye, Machine learning methods for sign language recognition: A critical review and analysis, Intelligent Systems with Applications, Volume 12, 2021, 200056, ISSN 2667-3053

[2] Ibham Veza, Asif Afzal, M.A. Mujtaba, Anh Tuan Hoang, Dhinesh Balasubramanian, Manigandan Sekar, I.M.R. Fattah, M.E.M. Soudagar, Ahmed I. EL-Seesy, D.W. Djamari, A.L. Hananto, N.R. Putra, Noreffendy Tamaldin,Review of artificial neural networks for gasoline, diesel and homogeneous charge compression ignition engine, Alexandria Engineering Journal, Volume 61, Issue 11, 2022, Pages 8363-8391, ISSN 1110-0168

[3] Ahmed KASAPBAŞI, Ahmed Eltayeb AHMED ELBUSHRA, Omar AL-HARDANEE, Arif YILMAZ, DeepASLR: A CNN based human computer interface for American Sign Language recognition for hearing-impaired individuals, Computer Methods and Programs in Biomedicine Update, Volume 2, 2022, 100048, ISSN 2666-9900

[4] Najla Musthafa, C.G. Raji, Real time Indian sign language recognition system, Materials Today: Proceedings, Volume 58, Part 1, 2022, Pages 504-508, ISSN 2214-7853

[5] E. Pereira-Montiel, E. Pérez-Giraldo, J. Mazo, D. Orrego-Metaute, E. Delgado-Trejos, D. Cuesta-Frau, J. Murillo-Escobar, Automatic sign language recognition based on accelerometry and surface electromyography signals: A study for Colombian sign language,

Biomedical Signal Processing and Control, Volume 71, Part B, 2022, 103201, ISSN 1746-8094

[6] Shagun Katoch, Varsha Singh, Uma Shanker Tiwary,Indian Sign Language recognition system using SURF with SVM and CNN, Array, Volume 14, 2022, 100141, ISSN 2590-0056

[7] P.K. Athira, C.J. Sruthi, A. Lijiya, A Signer Independent Sign Language Recognition with Co-articulation Elimination from Live Videos: An Indian Scenario, Journal of King Saud University - Computer and Information Sciences, Volume 34, Issue 3, 2022, Pages 771-781, ISSN 1319-1578

[8] Fatmi, R., Rashad, S., & Integlia, R. (2019). Comparing ANN, SVM, and HMM based Machine Learning Methods for American Sign Language Recognition using Wearable Motion Sensors. 2019 IEEE 9th Annual Computing and Communication Workshop and Conference (CCWC), 0290-0297.

[9] D. Anil Kumar, A.S.C.S. Sastry, P.V.V. Kishore, E. Kiran Kumar, 3D sign language recognition using spatio temporal graph kernels, Journal of King Saud University - Computer and Information Sciences, Volume 34, Issue 2, 2022, Pages 143-152, ISSN 1319-1578

[10] Boulares, Mehrez & Jemni, Mohamed. (2012). Mobile sign language translation system for deaf community. W4A 2012 - International Cross-Disciplinary Conference on Web Accessibility. 10.1145/2207016.2207049

[11] Rastgoo, Razieh & Kiani, Kourosh & Escalera, Sergio. (2020). Sign Language Recognition: A Deep Survey. Expert Systems with Applications. 164. 113794. 10.1016/j.eswa.2020.113794