Using Computer Vision and Deep Learning to Aid the Deaf

V Vivek, L V Vishak, Vipin R Bharadwaj and H S Gururaja

October 9, 2021

USING COMPUTER VISION AND DEEP LEARNING TO AID THE DEAF

Vivek V
dept. of Information Science,
B.M.S College of Engineering
Bengaluru, India
vivekv.contact@gmail.com

Vishak LV
dept. of Information Science,
B.M.S College of Engineering
Bengaluru, India
lvvishak@gmail.com

Vipin R Bharadwaj
dept. of Information Science,
B.M.S College of Engineering
Bengaluru, India
vipinrbharadwaj@gmail.com

Gururaja H S
dept. of Information Science,
B.M.S College of Engineering
Bengaluru, India
gururajhs@bmsce.ac.in

***Abstract*—This paper talks about the use of computer vision and machine learning to create a sign language translator for the deaf to convey their message to the general public. Majority of the world doesn't understand sign language and it makes it harder for the deaf to have a normal interaction. However, by applying these latest technologies it can bridge this gap and make the life of both the less abled and the abled easier. By detecting the gestures made by a person using sign language, it is possible to translate it into a spoken language. Thus, making sign language translatable like any other language. Giving a voice to the people who haven't been born with a voice is the primary goal of this piece of technology. We propose a solution where there is no bulky equipment required or any new modifications needed for the translation.**

I. INTRODUCTION

With technologies improving all around the world, it is bridging the gap between the less abled and the abled. The less-abled are able to do the same jobs and work that abled can. One such example is of Stephen hawking's chair which allowed the great scientist to have a voice when he didn't have one. A deaf person is not able to communicate using their voice and instead rely on gestures and signs with their hands to communicate. This is called sign language and there are many variations of it like the ASL(American Sign Language), ISL(Indian Sign Language) etc. However, the people who understand these sign languages are very few. This makes it extremely hard for the deaf to communicate.

In India, it is estimated that there are 18 million deaf people[1], which is not a small number. These people do not have a voice and giving them a way to express themselves is possible with current technologies. Many of the proposed technologies are either too bulky and require separate equipment like motion sensors to translate sign language.

Gesture detection is already used in many applications and now adapting it to sign language translation is fairly simple. A hand gesture recognition system for the general public will increase the interaction with deaf and decrease the discrimination towards them [2]. Open source tools like Python's OpenCV and Scikitlearn is used to implement project. It is easily available and accessible to all. This application can also be incorporated into mobile phones and used as applications [2].

To make the experience as organic as possible after the gesture has been detected, it relays the output to an audio device to speak the translated output. This makes the communication feel more natural instead of just reading out text on a screen.

Because of the various sign languages present, a feature to learn the user inputted signs apart from the pre-determined signs can also be made possible. Theoretically, the user can add any number of gestures they prefer and train the application instead of using the already trained sign language model[3].

Only using gesture recognition with computer vision is not enough. For higher flexibility and adaptability, the use of deep learning with computer vision will give rise to a highly accurate application that will benefit the deaf..

II. LITERATURE SURVEY

Presently, the most commonly used algorithms for object detection are Convolutional Neural Networks (RCNN), Faster-RCNN, Single Shot Detector (SSD) and You Only Look Once (YOLO). RCNN and SSD provide higher accuracy, whereas YOLO provides higher speed for object recognition. The combination of SSD & Mobile Nets algorithms is used to implement object-based recognition. SSD is implemented using Google based technology called VGG-16 architecture, which is a simple to implement classifier algorithm. Mobile Nets is a neural based ML algorithm which uses depth-wise separable convolution method for object detection. Mobile Nets optimizes latency in processing. Together with these two algorithms implements: frame differencing, optical flow, background subtraction and object tracking.[4]

Multiple objects are detected using Open CV on an embedded platform as well as a regular platform. In this technique cascade classifier algorithm is used for object detection. In this algorithm Haar-Like feature selection is used by cascade classifier. This algorithm with Haar-Like features is high-accuracy object recognition technique. The final implementation is compared between embedded platform & regular platform. This whole implementation was shown useful in object detection & identification in surveillance camera solutions.[5]

Hand Gesture Recognition Using Different Algorithms Based on Artificial Neural Network is also possible. One such method is by using MATLAB, a computational, visualization & programmatic language is used for implementation.[6] The two key algorithms which are used are Edge detection & Skin detection in this method. After capturing an image from the webcam, the image is converted as frames. Then frames are converted to grey-scale format. Then using Histogram equalization, edges are detected. Finally, through ANN algorithm, movement & gestures are identified.

Real time Finger Tracking and Contour Detection for Gesture Recognition using OpenCV is also possible. Here again, Haar-Like features are used for object recognition, as this provides high accuracy. It is one of the most widely used digital image features used in object recognition. This is quite robust to noise and various lighting conditions. Firstly, the gesture is acquired and then they are segmented before applying filter. The filtered images are then represented using contours. Finally, classification is done using different techniques. Ada Boost learning algorithm which provides stage-by-stage improvements based on training data sets, provide 70% accuracy. Convex-Hull algorithm detects gesture with proper segmentation and skin color provides 90% accuracy. So, any of these techniques can be used for final detection of gesture.[7]

III. PRESENT IMPLEMENTATION

A. *TOOLS AND TECHNOLOGIES USED:*

i. Python

Python's concise, easy-to-learn syntax prioritises readability, which lowers software maintenance costs. Modules and packages are supported by Python, which fosters programme modularity and code reuse. The Python interpreter and its substantial standard library are free to download and are distributed in source or in a binary form for all major platforms.

Python's simplicity and consistency, as well as access to excellent libraries and frameworks for AI and machine learning (ML), flexibility, platform freedom, and a large community, makes it the best choice for Deep Learning, Machine Learning and AI applications. Implementing deep learning algorithms can be difficult and time-consuming. To enable developers to come up with a great coding solution, it is critical to have a well-structured and well-tested environment. Python frameworks and libraries are used by programmers to reduce development time.

A software library is a collection of pre-written code that programmers can utilise to tackle common programming challenges. Python has a large number of such libraries for Artificial Intelligence and Machine Learning because to its robust technology stack. A few examples are Keras, TensorFlow, and Scikit-learn ML libraries which we have used to develop our project.

ii. TensorFlow

TensorFlow is a Machine Learning software library that is free and open-source. It can be used for a variety of applications, but it focuses on deep neural network training and inference.

It's an open-source artificial intelligence package that builds models using data flow graphs. It enables programmers to build large-scale neural networks with multiple layers. Classification, perception, understanding, discovering, prediction, and creation are some of the most common uses for TensorFlow.

Tensorflow combines Machine Learning and Deep Learning models and algorithms into one package. It makes use of Python as a user-friendly front-end and runs it in optimised C++. Developers can use Tensorflow to design a graph of computations to run. TensorFlow is designed & built to be user-friendly. The Tensorflow library includes a variety of APIs for creating large-scale deep learning architectures such as CNNs and RNNs.

TensorFlow is a graph-based programming language that allows developers to see the neural network's formation using Tensorboad. This application debugging tool is quite useful. Finally, Tensorflow is designed to be used in large-scale deployments. It runs on both the CPU and the GPU.

iii. Keras

Keras is one of the most powerful and easy to use python library, which is built on top of popular deep learning libraries like TensorFlow, Theano, etc., for creating deep learning models.

Keras runs on top of open-source machine libraries like TensorFlow, Theano or Cognitive Toolkit (CNTK). Theano is a python library used for fast numerical computation tasks. TensorFlow is the most famous symbolic math library used for creating neural networks and deep learning models. TensorFlow is very flexible and the primary benefit is distributed computing. CNTK is deep learning framework developed by Microsoft. It uses libraries such as Python, C#, C++ or standalone machine learning toolkits.

Theano and TensorFlow are very powerful libraries but difficult to understand for creating neural networks. Keras is based on minimal structure that provides a clean and easy way to create deep learning models based on TensorFlow or Theano. Keras is designed to quickly define deep learning models. Well, Keras is an optimal choice for deep learning applications.

iv. OpenCV

OpenCV is a cross-platform library using which we can develop real-time computer vision applications. It mainly focuses on image processing, video capture and analysis including features like face detection and object detection.

Computer Vision can be defined as a discipline that explains how to reconstruct, interrupt, and understand a 3D scene from its 2D images, in terms of the properties of the structure present in the scene. It deals with modelling and replicating human vision using computer software and hardware.

B. METHODOLOGY

The first step in any machine learning/deep learning algorithm is to obtain the dataset. To obtain the dataset we must provide input images of a particular sign. Multiple photographs of the images are taken and stored in the database and this will later be used to train the model. There needs to be an internal mapping of the sign and the textual meaning so that translation can occur.

Before, the storing of these images, the image must be pre-processed to remove the useless information. For example, the background, the person's face and all other aspects of the image is useless data to the algorithm. This sanitization of the input can be done in two ways. Firstly, we can limit the area at which input is being given and make sure that the user gives their input only on that part of the screen. Secondly, we can use object detection to track the hand and later use contour detection to track the fingers and other contours of the hand and take input from that. Both methods have their advantages and disadvantages. The former method is easier to implement and takes up lesser memory and processing power, however, unlike the latter method it is not very practical as users cannot be asked to sign in the same place every time, they want their actions to be translated.

Once the images have been extracted to give us only the sign part of it, we need to do further cleaning. An image is made up of three layers, namely - red, blue and green. The combination of these 3 images gives us a full coloured image. However, for what we are looking to do, we do not need colour of the image. We only need the shape and size of the hand that is signing in the image. To achieve this, we first convert the image to a greyscale image and then on this greyscale image we perform thresholding. Thresholding is a process of segmenting images; it converts the grayscale image into a binary image. the simplest form of thresholding is to convert the pixel of an image of a certain intensity to either a white or black pixel. The thresholding number should be decided from testing to see which is the best value to get a binary image of the hand where it's shape and size is clearly noticeable.

After these processes, the image is finally ready to be used for training the model. Multiple images of these are stored in a folder and then it is mapped to it's textual meaning in a simple database. This means that any form of sign language can be trained and used as long there is some training data. Finally, after all the images have been collected, we apply Deep learning to these images to obtain a model. Here, we have used Convolutional Neural Networks as the algorithm to build the model. CNN's is a very popular algorithm used in image recognition. It uses a convolution layer as the hidden layer in a multi-layered perceptron. It is very fast compared to other image detection algorithms. We do not need many layers in this project because we do not need to detect only the edges and contours of the hand and not very fine details. We have used the help of Keras and Tensorflow to build our model.

Once our model has been built, the input images are fed into the software. The input images undergo the same amount of image pre-processing and then the model is used to predict what the sign's textual meaning is. Finally, when sign is predicted with reasonable accuracy, it is converted into speech by using a simple python text to speech library. This is done in a separate thread and not the main thread as not to hinder the further sign prediction. The sentence is finally put together by adding the individual predicted words to finally help the deaf to successfully convert sign language into speech.

IV. HIGH LEVEL DIAGRAMS

1. Data Flow Diagram:

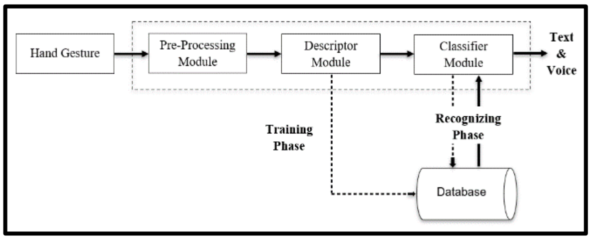


Figure 3 - Data Flow Diagram

2. System Architecture

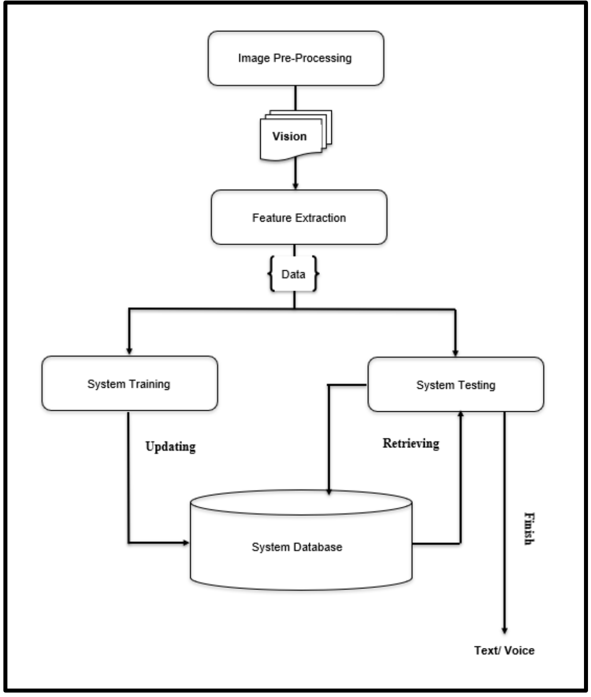


Figure 1 - System Architecture

3. State Transition Diagram

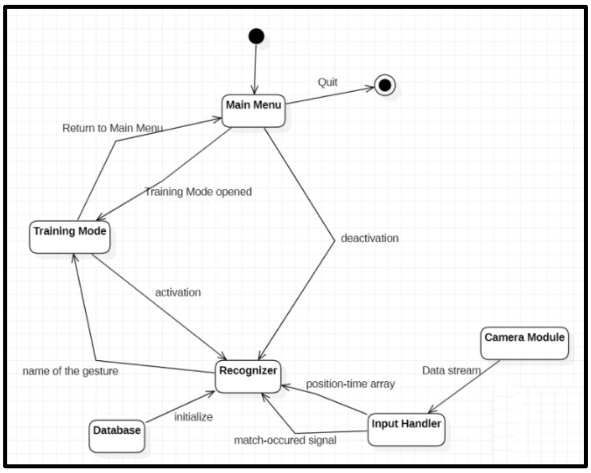


Figure 2 - State Transition Diagram

4. Complete Data Model

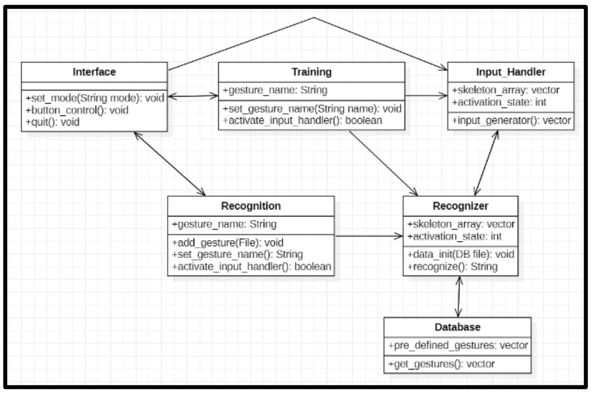
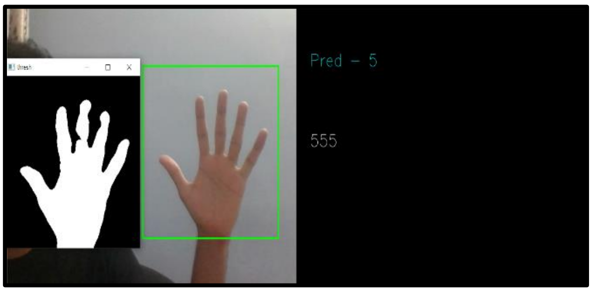


Figure 4 - Complete Data Diagram

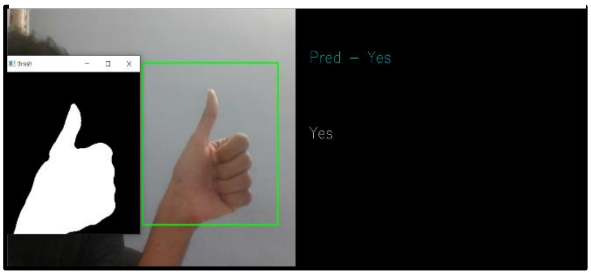
V. RESULT



Pred – 5

555

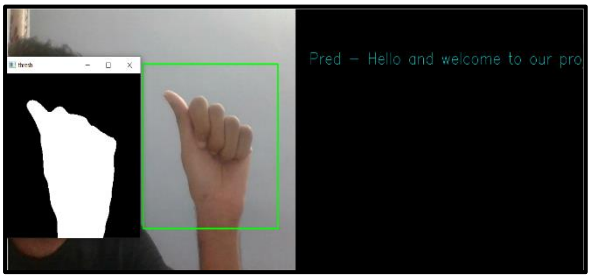
Figure VI - Predicting the sign 5



Pred – Yes

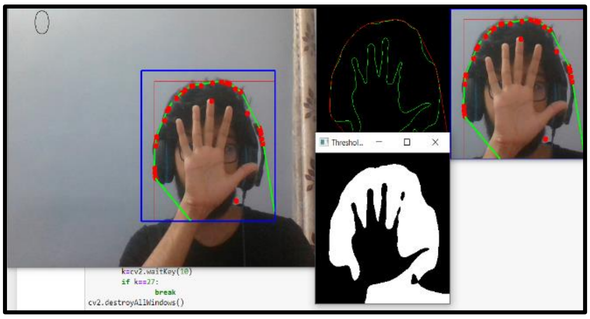
Yes

Figure V - Predicting of "Yes" gesture



Pred – Hello and welcome to our pro

Figure VII - Custom hand gesture trained for our project



```
        k=cv2.waitKey(10)
        if k==27:
                break
cv2.destroyAllWindows()
```

Figure VIII - Testing contour detection

VI. CONCLUSION

We have described the implementation of a system that converts Sign Language to English in this work. We've talked about how important a Sign Language Translator is when communicating with the deaf and mute. A webcam is used to capture a still hand image frame in this system. These frames are enhanced by post-processing. The sign language is then translated into English text using feature extraction and classification techniques. The text to speech API is used to transform this translation to speech. To obtain the final output, the system has implemented the aforesaid algorithms. A dataset of 26 signs from three distinct people is used to test the suggested model. The accuracy of our model is about 90%. Our future study will focus on developing a mobile application to implement this paradigm and also include a wider research to improve the efficiency of our model.

VII. FUTURE SCOPE

Before Additional Models: We concentrated our efforts on improving our trained model, but it's worth looking into other networks that have been shown to be effective in image categorization (e.g. a VGG or a ResNet architecture).

Image pre-processing: We believe that if the photos are heavily pre-processed, the classification task will be greatly simplified. Contrast correction, background subtraction, and maybe cropping are all examples of this. Using another CNN to localise and crop the hand would be a more robust technique.

Enhancement of the language model: Creating a bigram and trigram language model would allow us to handle phrases rather than single words. This necessitates improved letter segmentation as well as a more seamless procedure for retrieving photographs from users at a faster rate.

Camera Resolution & Platform Independence: Making our product work seamlessly across different range of resolutions of Cameras embedded across many platforms like Handheld devices, Tablets, PC, Laptops etc. We can also integrate our product to Closed Circuit Tele Vision (CCTV) to eliminate the need for any above-mentioned platforms. This provides a cost effective and commercially viable solution.

Video Processing: We can upgrade this existing product to include Video processing which enables it to recognize a wider range of gestures and movements. This upgrade will also significantly improve the accuracy of Gesture recognition, but this upgrade requires high storage space, high processing power and a fault tolerance system which can all be powered through commercially available Cloud systems.

REFERENCES

[1] Report by National Association of the Deaf, India - 2021

[2] Mahesh, M., Jayaprakash, A., & Geetha, M. (2017). Sign language translator for mobile platforms. 2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI). doi:10.1109/icacci.2017.8126001

[3] Memon, Z. A., Ahmed, M. U., Hussain, S. T., Baig, Z. A., & Aziz, U. (2017). Real Time Translator for Sign Languages. 2017 International Conference on Frontiers of Information Technology (FIT). doi:10.1109/fit.2017.00033

[4] G. Chandan, A. Jain, H. Jain and Mohana, "Real Time Object Detection and Tracking Using Deep Learning and OpenCV," 2018 International Conference on Inventive Research in Computing Applications (ICIRCA), 2018, pp. 1305-1308, doi: 10.1109/ICIRCA.2018.8597266.

[5] Guennouni, S., Ahaitouf, A., & Mansouri, A. (2014). Multiple object detection using OpenCV on an embedded platform. 2014 Third IEEE International Colloquium in Information Science and Technology (CIST). doi:10.1109/cist.2014.7016649

[6] S. K. Yewale and P. K. Bharne, "Hand gesture recognition using different algorithms based on artificial neural network," 2011 International Conference on Emerging Trends in Networks and Computer Communications (ETNCC), 2011, pp. 287-292, doi: 10.1109/ETNCC.2011.6255906.

[7] R. M. Gurav and P. K. Kadbe, "Real time finger tracking and contour detection for gesture recognition using OpenCV," 2015 International Conference on Industrial Instrumentation and Control (ICIC), 2015, pp. 974-977, doi: 10.1109/IIC.2015.7150886..