International Journal of Science and Research (IJSR)
ISSN: 2319-7064
SJIF (2022): 7.942

Sign Language Recognition Application for Deaf and Dumb People

Ruchika Gaidhani[1], Payal Pagariya[2], Aashlesha Patil[3], Tejaswini Phad[4], Dhiraj Birari[5]

[1, 2, 3, 4, 5]Department of Information Technology, MVPs KBT College of Engineering, Maharashtra, India
[1]*ruchikagaidhani[at]gmail.com*
[2]*pagariyapayal67[at]gmail.com*

Abstract: ***Our goal is to develop a model that can detect and movements and signs. We'll train a simple gesture detecting model for sign language conversion, which will allow people to communicate with persons who are deaf and mentally challenged. This project can be performed using a variety of methods, including KNN, Logistic Regression, Nave Bayes Classification, Support Vector Machine, and CNN. The method we have chosen is CNN because it has a higher level of accuracy than other methods. A computer program written in the programming language Python is used for model training based on the CNN system. By comparing the input with a pre-existing dataset created using Indian sign language, the algorithm will be able to understand hand gestures. Users will be able to recognize the signs offered by converting Sign Language into text as an output. by a sign language interpreter This approach is implemented in Jupyter Lab, which is an add-on to the Anaconda documentation platform. To improve even more, we'll convert the inputs to black and white and accept input from the camera after applying the Background subtraction approach. Because the mask is configured to identify human skin, this model doesn't need a simple background to work and can be constructed with just a camera and a computer.***

Keywords: SLR, Data set, Convolutional Neural Network, Accuracy, EfficientNetB0

1. Introduction

Sign language is the most popular and effective way for communication among hard hearing and normal people. Normal people find little difficulty in understanding and interpreting the meaning of sign language expressed by the hearing impaired, it is inevitable to have an interpreter for the translation of sign language. Understanding sign language is the primary enablers in helping hard of hearing people with the rest of society. since most people do not know sign language and interpreters are very difficult to come by we have come up with a real time method using neural networks for finger spelling based Indian sign language.

To achieve this, a computer must first learn how to react to inputs and variables. The computer should be taught with a large amount of data, the data required to train the computer is determined by the desired output and the machine's operation. We create a computer model that can identify human hand motions; there are numerous apps that function with hand gestures that we observe in our daily lives. Look at the console in our living room; connect it to a sensor, and we'll be able to play tennis with our hands. We have created a touch detection model that translates sign language into speech. There are a number of devices that rely on touch detection, whether for security or entertainment. Sign language is a vision-based language that uses a combination of material, body language, and gestures, fingers, and orientation, posture, and hand and body movements, as well as eyes, lips, and wholeness. facial expressions and speech. A variety of signature circuits exist, just as there are regional variations in spoken language. Through gestures and gestures, facial expressions, and lip movements, we can spell the letters of each word with our fingers and maintain a certain vocabulary of Indian Sign Language (ISL), American Sign Language (ASL), and Portuguese Signature (PSL). Sign language can be separated or used continuously. People interact with a single sign language by performing a single word movement, while continuous sign language can be a series of steps that form a coherent sentence. All methods of identifying hand movements are often classified as based on vision and based on measurements taken by sensors embedded in gloves. This vision-based process uses human computer interactions to detect touch with bare hands. OpenCV is used to detect sign languages in this project. Uses a webcam to detect user touch; Our model can distinguish handmade touches with bare hands, so we will not need gloves for this project. OpenCV is used to detect sign languages in this project. Uses a webcam to detect the movement of a user's hand, with words displayed on the screen as $_{output}$. Our project aims to help people who do not know sign language well by identifying and converting man-made symbols into legible characters. Machine learning, especially CNN, can help us achieve this. We want to use an image from the web camera/phone camera to train a model that can predict text (using IP camera software and OpenCV). Because the model was trained using a well-known dataset (ASL), there will be enough data for the training algorithm to produce a precise and accurate model. Because the model's code is written in Python, the project can be completed on a simple computer without the use of high-end processing units or GPUs. The software Jypter Lab, where this model is trained and implemented, is built on the Anaconda Documentation platform. The many concepts involved, as well as the technique for carrying out this project, are addressed in detail in the following sections of the paper.

2. Related Work

Sign Language Recognition (SLR) system, which is required to recognize sign languages, has been widely studied for years. The studies are based on various input sensors, gesture segmentation, extraction of features and classification methods. This paper aims to analyze and

compare the methods employed in the SLR systems, classification methods that have been used, and suggests the most promising method for future research. Due to recent advancement in classification methods, many of the recent proposed works mainly contribute on the classification methods, such as hybrid method and Deep Learning. Based on our review, HMM based approaches have been explored extensively in prior research, including its modifications. This study is based on various input sensors, gesture segmentation, extraction of features and classification methods. Due to recent advancement in classification methods, many of the recently proposed works mainly contribute to the classification methods, such as hybrid method and Deep Learning. Based on our review, HMM-based approaches have been explored extensively in prior research, including its modifications. Hybrid CNN-HMM and fully Deep Learning approaches have shown promising results and offer opportunities for further exploration

Communication between Deaf-Dumb People and Normal People using Chat application have become a powerful media that assist people to communicate in different languages with each other. There are lots of chat applications that are used different people in different languages but there are not such a chat application that has facilitate to communicate with sign languages. The developed system is based on Sinhala Sign language. The system has included four main components as text messages are converted to sign messages, voice messages are converted to sign messages, sign messages are converted to text messages and sign messages are converted to voice messages. Google voice recognition API has used to develop speech character recognition for voice messages. The system has been trained for the speech and text patterns by using some text parameters and signs of Sinhala Sign language is displayed by emoji. Those emoji and signs that are included inthis system will bring the normal people more close to the disabled people. This is a 2 way communication system but it uses pattern of gesture recognition which is not very reliable in getting appropriate output.

A System for Recognition of Indian Sign Language for Deaf People using Otsu's Algorithm: In this project, they are capturing hand gestures through webcam and convert this image into gray scale image. The segmentation of gray scale image of a hand gesture is performed using Otsu thresholding algorithm. Total image level is divided into two classes one is hand and other is background. The optimal threshold value is determined by computing the ratio between class variance and total class variance. To find the boundary of hand gesture in image Canny edge detection technique is used. In Canny edge detection we used edge based segmentation and threshold based segmentation. Then Otsu's algorithm is used because of its simple calculation and stability. This algorithm fails, when the global distribution of the target and background vary widely.

3. Methodology

In this work, we propose a computer-vision system for the task of sign language recognition. Our proposed method doesn't depend on using glove-based sensors because hand gestures are only a part of sign language. our dataset was collected in various real backgrounds and lightning conditions rather than the lab. Deep learning can be divided into 2 categories: supervised and unsupervised. In supervised learning, the data is labeled during training. Choosing the network type and architecture depends on the task at hand. CNN's have been used for isolated image recognition. Modern image recognition models have millions of parameters. Training them from scratch requires a lot of labeled training data and a lot of computing power (hundreds of GPU-hours or more). In this work, We have used EfficientNetB0 architecture.

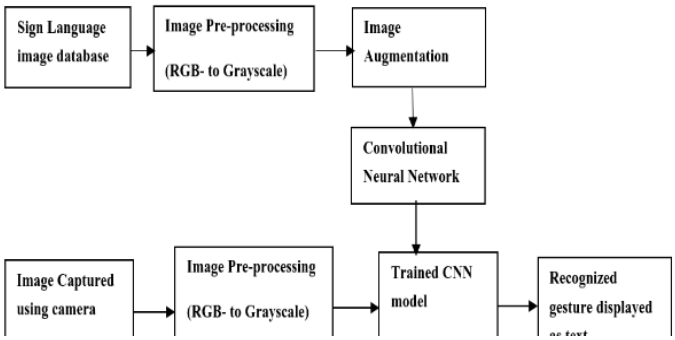

Figure 1: System Description

a) Image Database: The database contains images of different hand signs. These images are taken from different users with multiple repetitions. The resolution of the images may be varying. Different datasets are available for Sign Language.
b) Image Pre-processing: Training the raw images as it is might lead to poor performance. Thus, simple image processing algorithms can be implemented to achieve maximum accuracy. Image processing algorithms such as RGB to gray conversion reduce the training time and power consumption. The noise from the images can be eliminated.
c) Image Augmentation: Data augmentation helps in the case of a small database. Image augmentation is achieved by doing various operations, including Mirroring – Flip the image horizontally; Cropping – Cutting out a certain portion of an image; Rotating, shearing, local warping; Color shifting – For RGB dataset, the pixel values can be modified.
d) CNN Training & Training Options: Deep learning is used for the project. Training options are set accordingly before training the database using any CNN architecture. The training options are maximum batch size, number of the epoch, and learning rate.
e) Image Acquisition: Any camera, even a laptop webcam can be used to acquire the image to be recognized. Because in the end the image captured will be reduced to the input size of the CNN. Hence the camera need not be high-resolution.
f) Display Output: The recognized sign can is displayed in text format or can be also conveyed with audio description.

3.1 Algorithm

Different types of Neural Networks are used for different purposes, for example for predicting the sequence of words we use Recurrent Neural Networks, similarly for image

classification we use Convolution Neural networks. Convolutional neural networks are distinguished from other neural networks by their superior performance with image, speech, or audio signal inputs. They have three main types of layers, which are:

- Convolutional layer
- Pooling layer
- Fully-connected (FC) layer

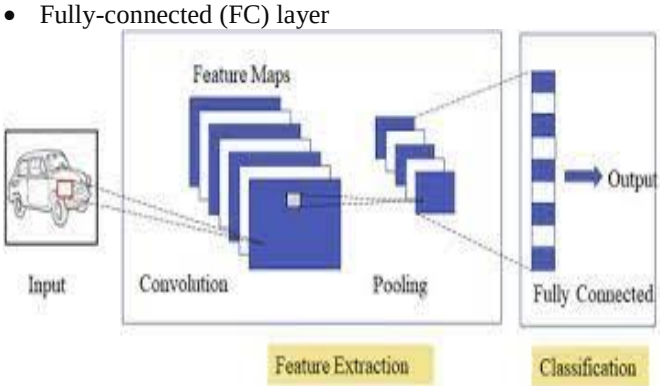

Figure 2: CNN Architecture

The convolutional layer is the first layer of a convolutional network. While convolutional layers can be followed by additional convolutional layers or pooling layers, the fully-connected layer is the final layer. With each layer, the CNN increases in its complexity, identifying greater portions of the image. Earlier layers focus on simple features, such as colors and edges. As the image data progresses through the layers of the CNN, it starts to recognize larger elements or shapes of the object until it finally identifies the intended object.

3.1.1 Convolutional Layer

The convolutional layer is the core building block of a CNN, and it is where the majority of computation occurs. It requires a few components, which are input data, a filter, and a feature map. Let's assume that the input will be a color image, which is made up of a matrix of pixels in 3D. This means that the input will have three dimensions—a height, width, and depth—which correspond to RGB in an image. We also have a feature detector, also known as a kernel or a filter, which will move across the receptive fields of the image, checking if the feature is present. This process is known as a convolution.

3.1.2 Pooling Layer

Pooling layers, also known as down sampling, conducts dimensionality reduction, reducing the number of parameters in the input. Similar to the convolutional layer, the pooling operation sweeps a filter across the entire input, but the difference is that this filter does not have any weights. Instead, the kernel applies an aggregation function to the values within the receptive field, populating the output array. There are two main types of pooling:

a) **Max pooling:** As the filter moves across the input, it selects the pixel with the maximum value to send to the output array. As an aside, this approach tends to be used more often compared to average pooling.
b) **Average pooling:** As the filter moves across the input, it calculates the average value within the receptive field to send to the output array.

While a lot of information is lost in the pooling layer, it also has a number of benefits to the CNN. They help to reduce complexity, improve efficiency, and limit risk of overfitting.

3.1.3 Fully-Connected Layer

The name of the full-connected layer aptly describes itself. As mentioned earlier, the pixel values of the input image are not directly connected to the output layer in partially connected layers. However, in the fully-connected layer, each node in the output layer connects directly to a node in the previous layer.

This layer performs the task of classification based on the features extracted through the previous layers and their different filters. While convolutional and pooling layers tend to use ReLu functions, FC layers usually leverage a softmax activation function to classify inputs appropriately, producing a probability from 0 to 1.

3.3 Implementation

It depends on machine learning algorithms. In this projects case, it was neural networks. Such an algorithm looks like:

1) begin with its object: model = Sequential ()
2) then consist of layers with their types: model. add (type_of_layer ())
3) after adding a sufficient number of layers the model is compiled. At this moment Keras communicates with TensorFlow for construction of the model.

We have added EfficientNetB0 pre-trained model to sequential () function.

EfficientNetB0 function

```
tf.keras.applications.EfficientNetB0(
    include_top=True,
    weights="imagenet",
    input_tensor=None,
    input_shape=None,
    pooling=None,
    classes=1000,
    classifier_activation="softmax",
    **kwargs
)
```

EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks

Each Keras Application expects a specific kind of input preprocessing. For EfficientNet, input preprocessing is included as part of the model (as a Rescaling layer), and thus tf. keras. applications. efficientnet. preprocess_input is actually a pass-through function. EfficientNet models expect their inputs to be float tensors of pixels with values in the [0-255] range.

Stage i	Operator $\hat{\mathcal{F}}_i$	Resolution $\hat{H}_i \times \hat{W}_i$	#Channels $\hat{C}_i$	#Layers $\hat{L}_i$
1	Conv3x3	224 × 224	32	1
2	MBConv1, k3x3	112 × 112	16	1
3	MBConv6, k3x3	112 × 112	24	2
4	MBConv6, k5x5	56 × 56	40	2
5	MBConv6, k3x3	28 × 28	80	3
6	MBConv6, k5x5	14 × 14	112	3
7	MBConv6, k5x5	14 × 14	192	4
8	MBConv6, k3x3	7 × 7	320	1
9	Conv1x1 & Pooling & FC	7 × 7	1280	1

Figure 3: Architecture of EfficientNetB0

The Figureshown denotes the kernel size for convolution operations along with the resolution, channels, and layers in EfficientNet-B0.

4. Result

We have achieved an accuracy of 95.8% in our model using EfficientNet and other additional layers which is a better accuracy then most of the current research papers on Indian sign language. our main aim was to create a project which can be used with readily available resources. So we have converted in into an android application which is user friendly and handy to use. A sensor like kinect not only isn't readily available but also is expensive for most of audience to buy and our model uses a normal webcam of the mobile phone hence it is great plus point. Below are the confusion matrices for our results.

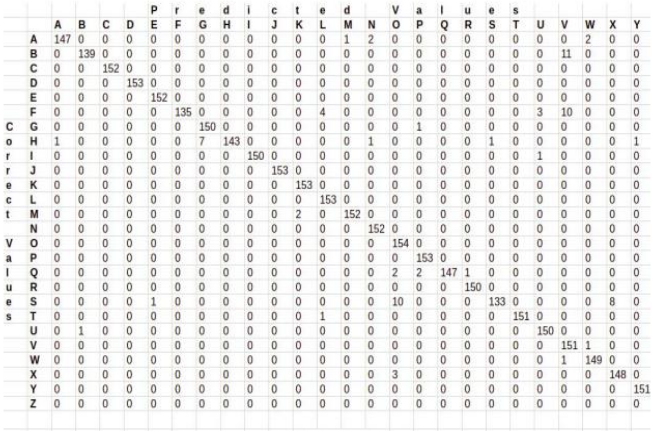

Figure 4: Confusion Matrix of result.

5. Conclusion

In this report, a functional real time vision based sign language recognition for deaf and hearing impaired people have been developed for asl alphabets. We achieved final accuracy of 95.8% on our dataset. We are able to improve our prediction after implementing two layers of algorithms in which we verify and predict symbols which are more similar to each other. This way we are able to detect almost all the symbols provided that they are shown properly, there is no noise in the background and lighting is adequate.

References

[1] T. Yang, Y. Xu, and "A., Hidden Markov Model for Gesture Recognition", CMU-RI-TR-94 10, Robotics Institute, Carnegie Mellon Univ., Pittsburgh, PA, May 1994.

[2] PujanZiaie, Thomas M¨uller, Mary Ellen Foster, and Alois Knoll"A Na¨ive Bayes Munich, Dept. of Informatics VI, Robotics and Embedded Systems, Boltzmannstr.3, DE-85748 Garching, Germany.

[3] Mohammed Waleed Kalous, Machine recognition of Auslan signs using PowerGloves: Towards large-lexicon recognition of sign language.

[4] aeshpande3. github. io/A-Beginner%27s-Guide-To-Understanding-Convolutional-Neural-Networks-Part-2/

[5] http: //www-i6. informatik. rwth-aachen. de/~dreuw/database. php

[6] Pigou L., Dieleman S., Kindermans PJ., Schrauwen B. (2015) Sign Language Recognition Using Convolutional Neural Networks. In: Agapito L., Bronstein M., Rother C. (eds) Computer Vision-ECCV 2014 Workshops. ECCV 2014. Lecture Notes in Computer Science, vol 8925. Springer, Cham

[7] Zaki, M. M., Shaheen, S. I.: Sign language recognition using a combination of new vision based features. Pattern Recognition Letters 32 (4), 572–577 (2011) 25

[8] N. Mukai, N. Harada and Y. Chang, "Japanese Fingerspelling Recognition Based on Classification Tree and Machine Learning, " 2017 Nicograph International (NicoInt), Kyoto, Japan, 2017, pp.19-24. doi: 10.1109/NICOInt.2017.9

[9] ByeongkeunKang, Subarna Tripathi, Truong Q. Nguyen "Real-time sign language fingerspelling recognition using convolutional neural networks from depth map" 2015 3rd IAPR Asian Conference on Pattern Recognition (ACPR)

[10] https: //opencv. org/

[11] https: //en. wikipedia. org/wiki/TensorFlow

[12] https: //en. wikipedia. org/wiki/Convolutional_neural_network