SIGN LANGUAGE DETECTION

Sadhavi Thapa [1] , Kanishka Gaur [1], Department of Data Science and Business Studies, SRM Institute of Science & Technology

Abstract. Sign language is a lingua among the speech and hearing-impaired community. It is hard for most people unfamiliar with sign language to communicate without an interpreter. Sign language recognition appertains to tracking and recognizing the meaningful emotion of humanmade with the head, arms, hands, fingers, etc. The technique that has been implemented here, transcribes the gestures from sign language to a spoken language which the listener easily understands. The gestures that have been translated include alphabets and words from static images. This becomes more important for people who rely entirely on gestural sign language to communicate with someone who does not understand sign language. Most of the systems that are under use face a recognition problem with the skin tone, by introducing a filter will identify the symbols irrespective of the skin tone. The aim is to represent features that will be learned by a system known as convolutional neural networks (CNN), which contains four types of layers: convolution layers, pooling/subsampling layers, nonlinear layers, and fully connected layers.

Keywords: Sign language, CNN, training, dataset, filters

1. Introduction:

American sign language is a predominant sign language Since the only disability Deaf and Dumb (hereby referred to as D&M) people have is communication-related and since they cannot use spoken languages, the only way for them to communicate is through sign language. Communication is the process of the exchange of thoughts and messages in various ways such as speech, signals, behavior, and visuals. D&M people make use of their hands to express different gestures to express their ideas to other people. Gestures are non-verbally exchanged messages and these gestures are understood with vision. This nonverbal communication between deaf and dumb people is called sign language. Sign language is a language that uses gestures instead of sound to convey meaning by combining hand shapes, orientation, movement of the hands, arms, or body, facial expressions, and lip patterns. Contrary to popular belief, sign language is not international. These vary from region to region.

Sign language is a visual language and consists of 3 major components [6]:

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.

Figure 1

Minimizing the verbal exchange gap between D&M and non-D&M people turns into a want to make certain effective conversations among all. Sign language translation is among the growing lines of research and it enables the maximum natural manner of communication for those with hearing impairments. According to recent developments in the area of deep learning, neural networks may have far-reaching implications and implementations for sign language analysis. In the proposed system, Convolutional Neural Network (CNN) is used to classify images of sign language because convolutional networks are faster in feature extraction and classification of images over other classifiers. The environment may also recognize a sign as a compression technique for information transmission, which is then reconstructed by the receiver. The signs are divided into two categories: static and dynamic signs. The movement of body parts is frequently included in dynamic signs. Depending on the meaning of the gesture, it may also include emotions. A hand gesture recognition system offers an opportunity for deaf people to talk with vocal humans without the need for an interpreter. The system is built for the automated conversion of ASL into textual content and speech.

2. Literature Survey:

In recent years there has been tremendous research done on hand gesture recognition. With the help of a literature survey, we realized that the basic steps in hand gesture recognition are:

- Data acquisition
- Data pre-processing
- Feature extraction
- Gesture classification

2.1 Data acquisition:

The different approaches to acquiring data about the hand gesture can be done in the following ways:

1. Use of sensory devices: It uses electromechanical devices to provide exact hand configuration and position. Different glove-based approaches can be used to extract information. But it is expensive and not user-friendly.

2. Vision-based approach: In vision-based methods, the computer webcam is the input device for observing the information of hands and/or fingers. The Vision Based methods require only a camera, thus realizing a natural interaction between humans and computers without the use of any extra devices, thereby reducing cost. These systems tend to complement biological vision by describing artificial vision systems implemented in software and hardware. The main challenge of vision-based hand detection ranges from coping with the large variability of the human hand's appearance due to a huge number of hand movements, to different skin-color possibilities as well as to the variations in viewpoints, scales, and speed of the camera capturing the scene.

2.2 Data Pre-Processing and 2.3 Feature extraction for vision-based approach:

1. In [1] the approach for hand detection combines threshold-based color detection with background subtraction. We can use the AdaBoost face detector to differentiate between faces and hands as they both involve similar skin color.
2. We can also extract the necessary image which is to be trained by applying a filter called Gaussian Blur (also known as Gaussian smoothing). The filter can be easily applied using open computer vision (also known as OpenCV) and is described in [3].
3. For extracting the necessary image which is to be trained we can use instrumented gloves as mentioned in [4]. This helps reduce computation time for Pre-Processing and gives us more concise and accurate data compared to applying filters on data received from video extraction.

2.4 Gesture Classification:

1. In [1] Hidden Markov Models (HMM) is used for the classification of the gestures. This model deals with dynamic aspects of gestures. Gestures are extracted from a sequence of video images by tracking the skin-color blobs corresponding to the hand into a body–face space centered on the face of the user.
2. In [2] Naïve Bayes Classifier is used which is an effective and fast method for static hand gesture recognition. It is based on classifying the different gestures according to geometric-based invariants which are obtained from image data after segmentation.
3. According to the paper "Human Hand Gesture Recognition Using a Convolution Neural Network" by Hsien-I Lin, Ming-Hsiang Hsu, and Wei-Kai Chen (graduates of the Institute of Automation Technology National Taipei University of Technology Taipei, Taiwan), they have constructed a skin model to extract the hands out of an image and then apply a binary threshold to the whole image.

3. Methodology:

The system uses a vision-based approach. All signs are represented with bare hands and so it eliminates the problem of using any artificial devices for interaction.

3.1 Data Set Generation:

No proper dataset was available for our project which could be a proper fit, couldn't find a dataset in the form of raw images that matched our requirements. Hence, we decided to create our own data set. The steps we followed to create our data set are as follows.

Used the Open computer vision (OpenCV) library in order to produce our dataset.

Firstly, we captured around 800 images of each of the symbols in ASL (American Sign Language) for training purposes and around 200 images per symbol for testing purposes.

First, we capture each frame shown by the webcam of our machine. In each frame we define a Region Of Interest (ROI) which is denoted by a green bounded square as shown in the image below:

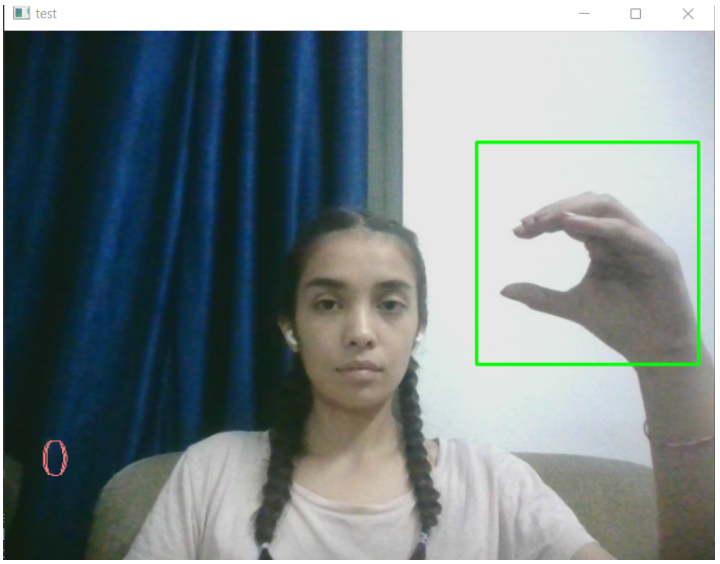

Figure 2

Then, we apply the Gaussian Blur Filter to our image which helps us extract various features of our image. The image, after applying Gaussian Blur, looks as follows:

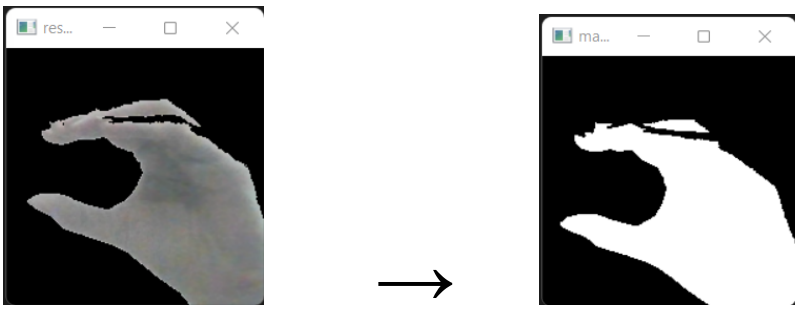

Figure 3

3.2 Gesture Classification:

Our approach uses one layer of the algorithm interconnected (convolution + max-pooling layers) to predict the final symbol of the user.

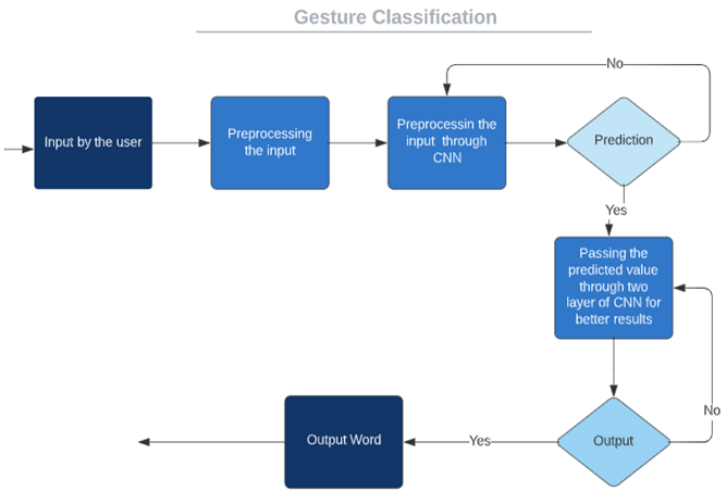

Figure 4

Algorithm Layer:

1. Apply color space conversions, gaussian blur filter, and threshold to the frame taken with OpenCV to get the processed image after feature extraction.
2. This processed image is passed to the CNN model for prediction.
3. If a letter is detected for more than 30 frames then the letter is printed.

CNN Model:

1. 1st Convolution Layer: The input picture has a 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.

2. 1st Pooling Layer: The pictures are downsampled using max pooling of 2x2 i.e we keep the highest value in the 2x2 square of an array. Therefore, our picture is down-sampled to 63x63 pixels.

3. 2nd Convolution Layer: Now, these 63 x 63 from the output of the first pooling layer is served as an input 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.

4. 2nd Pooling Layer: The resulting images are down-sampled again using a max pool of 2x2 and are reduced to 30 x 30 resolution of images.

5. 1st Densely Connected Layer: 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 30x30x32 =28800 values. The input to this layer is an array of 28800 values. The output of this layer is fed to the 2nd Densely Connected Layer. We are using a dropout layer of value 0.5 to avoid overfitting.

6. 2nd Densely Connected Layer: Now the output from the 1st Densely Connected Layer is used as an input to a fully connected layer with 96 neurons.

7. Final layer: 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).

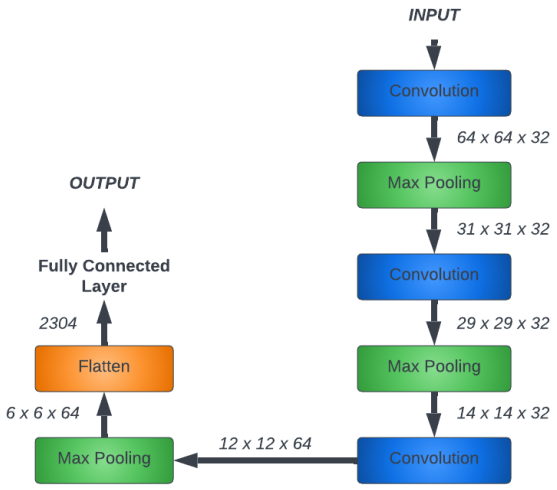

Figure 5

3.3 Training and Testing:

We convert our input images (RGB) into grayscale and apply gaussian blur to remove unnecessary noise. We apply an adaptive threshold to extract our hand from the background and resize our images to 128 x 128.

We feed the input images after pre-processing to our model for training and testing after applying all the operations mentioned above.

The prediction layer estimates how likely the image will fall under one of the classes. So, the output is normalized between 0 and 1 such that the sum of each value in each class sums to 1. We have achieved this using the SoftMax function.

At first, the output of the prediction layer will be somewhat far from the actual value. To make it better we have trained the networks using labeled data. The cross-entropy is a performance measurement used in the classification. It is a continuous function that is positive at values which is not the same as the labeled value and is zero exactly when it is equal to the labeled value. Therefore, we optimized the cross-entropy by minimizing it as close to zero. To do this in our network layer we adjust the weights of our neural networks. TensorFlow has an inbuilt function to calculate the cross entropy.

4. Challenges Faced:

Many challenges were faced during the project. The very first issue we faced was concerning the data set. We wanted to deal with raw images and that too square images as CNN in Keras since it is much more convenient working with only square images.

We couldn't find any existing data set as per our requirements and hence we decided to make our own data set. The second issue was to select a filter that we could apply to our images so that proper features of the images could be obtained and hence then we could provide that image as input for the CNN model.

Tried various filters including binary threshold, canny edge detection, Gaussian blur, etc. but finally settled with Gaussian Blur Filter.

More issues were faced relating to the accuracy of the model we had trained in the earlier phases. This problem was eventually improved by increasing the input image size and also by improving the data set.

5. Results:

We have achieved an accuracy of **93.8%** in our model using only layer 1 of our algorithm, and using **layer 1** we achieve an accuracy of **98.0%**, which is a better accuracy than most of the current research papers on American sign language.

Most of the research papers focus on using devices like Kinect for hand detection.

In [7] they build a recognition system for Flemish sign language using convolutional neural networks and Kinect and achieve an error rate of **2.5%.**

In [8] a recognition model is built using a hidden Markov model classifier and a vocabulary of 30 words and they achieve an error rate of **10.90%**.

In [9] they achieve an average accuracy of **86%** for 41 static gestures in Japanese sign language.

Using depth sensors map [10] achieved an accuracy of **99.99%** for observed signers and **83.58%** and **85.49%** for new signers.

They also used CNN for their recognition system. One thing should be noted that our model doesn't use any background subtraction algorithm whiles some of the models present above to do that.

So, once we try to implement background subtraction in our project the accuracies may vary. On the other hand, most of the above projects use Kinect devices but our main aim was to create a project which can be used with readily available resources. A sensor like Kinect not only isn't readily available but also is expensive for most of the audience to buy and our model uses a normal webcam of the laptop hence it is a great plus point.

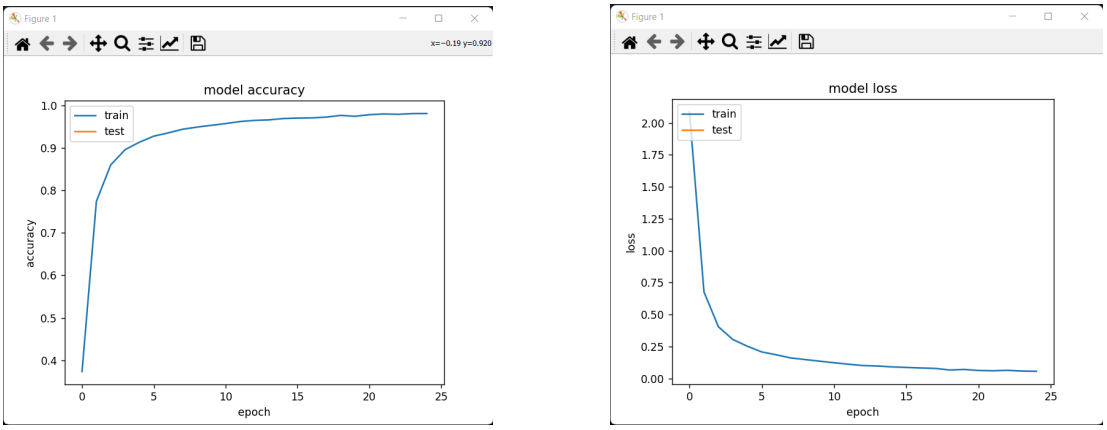

Figure 6

6. Conclusion:

The proposed system successfully predicts the signs of signs and some common words under different lighting conditions and different speeds. Accurate masking of the images is being done by giving a range of values that could detect human hands dynamically. The proposed system uses CNN for the training and classification of images. For classification and training, more informative features from the images are finely extracted and used. A total of 1199 static images for each sign are used for training to get the accurate output. Finally, the output of the recognized sign is shown in the form of text as well as converted into speech. The system is capable of recognizing all 26 alphabets out of which 10 are predicted with 98% accuracy. Thus this is a user-friendly system that can be easily accessed by all the deaf and people.

7. Future Work:

We are planning to achieve higher accuracy even in the case of complex backgrounds by trying out various background subtraction algorithms.

We are also thinking of improving the Pre Processing to predict gestures in low-light conditions with higher accuracy.

This project can be enhanced by being built as a web/mobile application for the users to conveniently access the project. Also, the existing project only works for ASL; it can be extended to work for other native sign languages with the right amount of data set and training. This project implements a finger spelling translator; however, sign languages are also spoken on a contextual basis where each gesture could represent an object or verb. So, identifying this kind of contextual signing would require a higher degree of processing and natural language processing (NLP).

8. 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] Pujan Ziaie, 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]https://docs.opencv.org/2.4/doc/tutorials/imgproc/gausian_median_blur_bilateral_filter/gausian_median_blur_bilateral_filter.html

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

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

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

[7] 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

[8] 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).

[9] 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

[10] Byeongkeun Kang, 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)

[11] Number System Recognition (https://github.com/chasinginfinity/number-sign-recognition)

[12] https://opencv.org/

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

[14] https://en.wikipedia.org/wiki/Convolutional_neural_nework