International Journal of Engineering Applied Sciences and Technology, 2021
Vol. 5, Issue 12, ISSN No. 2455-2143, Pages 135-139
Published Online April 2021 in IJEAST (http://www.ijeast.com)

INDIAN SIGN LANGUAGE TO TEXT CONVERSION IN REAL-TIME USING MACHINE LEARNING

Yash Narang, Aditya Sharma
Computer Science & Engineering, SRM Institute of Science & Technology,
Delhi-NCR Campus, Ghaziabad, Uttar Pradesh

ABSTRACT — Before language as we know it today existed, correspondence between individual people was emblematic and comprised of the utilization of hand signs. Sign language is one of the oldest and most natural form of language for communication. In our community today, contact with hearing impaired (deaf/mute) persons is a major challenge. This can be attributed to the fact that their method of communication (sign language or hand gestures at community level) requires an intermediary at all levels. We have thought of a constant strategy utilizing neural organizations for fingerspelling dependent on Indian gesture-based communication. In our technique, the hand is first gone through a filter and after the filter is applied the hand is passed through a classifier that predicts the class of the hand motions. Our strategy provides 95.7% precision for the 26 letters of the alphabet.

Keywords — Algorithms, Cybersecurity, Network- Security, Machine Learning.

I. INTRODUCTION

The only form of challenges the deaf and mute people face is communication-based, sign language is an important part of their lives because it is virtually the only way they can communicate. Just like we use verbal signals for communication, deaf and mute people use hand signals to express their thoughts to other people. Communication using these gestures is understood with the help of visual senses.

The American Sign Language and the British Sign Language are the most widely used sign languages and therefore, much research has been done in developing systems for converting them to text or speech. Our project focuses on making a system for recognizing the Indian Sign Language. India has around 1.1 million people who are both hearing impaired and vocally challenged. The total number of sign language literate people is about 2% of that number which is too low.
This is because people with no physical challenges, like us, do not know sign language and are not able to communicate with the hearing impaired as a result a majority of them find no point in learning sign language.

This project focuses on building and training a neural network that recognizes the Indian Sign Language fingerspelling individually and then combines these gestures to make words.

The system uses Open CV to take pictures of the hand signs using the web camera of the computer and then applies the preprocessing filters to convert the RGB images from the camera to black and white threshold maps which are further processed by the neural network to classify them from among the 26 letters of the alphabet or from among the 0-9 digits. The signs that we aim to train on our neural network are:

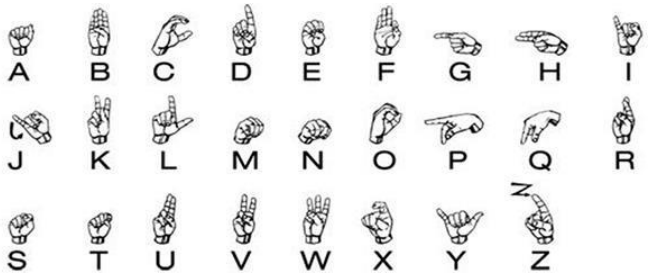


Fig 1. Indian Sign Language System

II. METHODOLOGY

A. DATA COLLECTION

- **Vision-Based Technique**

In this procedure, the only input appliance required to us is the computer camera for monitoring the movements of hand and fingers, making it user friendly because of the use of an unpolished communication between humans and computer and also converting the biological perception into artificial perception for the great understanding by the application software. The most important ultimatum of using this to detect gestures is to encounter inconsistency of gestures possibly due to the considerable amount of movements, diverse nature of skin-color, and also due to the capturing speed of the camera. [6]

- **Sensory Equipment**

These appliances use an electro-mechanical approach to determine hand gestures with maximum accuracy. One of the gadgets is Glove based technique which is quite expensive and not very user-friendly.

International Journal of Engineering Applied Sciences and Technology, 2021
Vol. 5, Issue 12, ISSN No. 2455-2143, Pages 135-139
Published Online April 2021 in IJEAST (http://www.ijeast.com)

B. DATA PRE-PROCESSING AND FEATURE EXTRACTION

In [5] the procedure for apprehending the hand movement involves the association of Threshold- based exposure and diminution of background. Also, the use of Ada boost face locator to categorize and analyze which is a hand movement and which is not. A filter named Gaussian blur is used to uproot the imperative picture which has the necessity to be trained. For Gaussian blur to work properly Open CV is used so that the filter can be handled efficiently. The procedure for this is briefly discussed. Another filter is applied to the RGB images to convert them to black and white images. This filter is available in the OpenCV library under the name cvtColor() and is used to convert images from one color space to another.

C. GESTURE CLASSIFICATION

T. Yang, Y. Xu, and "A. [5] in 1994 used Hidden Markov Models for hand-sign classification and made a model which dealt with the dynamic nature of the signs. In this model, they extracted skin- colored blobs from a video sequence of the gestures. A fast lookup indexing table was used to filter the images and recognized symbolic and deistic signs. Blob analysis involves the recognition of homogeneous areas based on the color and the location of similarly colored 'Blobs".

Hsien-I Lin , Ming-Hsiang Hsu, and Wei-Kai Chen
[2] in their paper extracted the image of the hand out of an image using a skin model they constructed and then applied a binary threshold filter to the image. They trained a CNN (Con- volution Neural Network) on 7 hand signs and obtained a 95% accuracy after calibrating the threshold image about its principal axis.

Pujan Ziaie, Thomas Muller, Mary Ellen Foster, and Alois Knoll [7] used a Naive Bayes classifier which proved fast and effective in hand-sign recognition.

The model was used in recognizing static hand gestures and not dynamic ones which we intend to recognize. This method is not based on skin color recognition but on the geometric invariants which are acquired after segmentation. In this system, the signs extracted from a static background video are segmented and labeled first and then geometric variants are extracted.

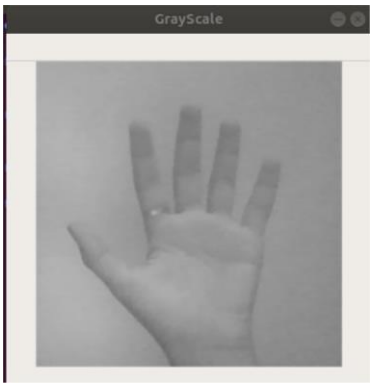
Then a K-nearing neighbor with Distance Weighting (KNNDW) (K-Nearest Neighbour with Distance Weighing) is used to supply the locally weighted Naive Bayes classifier to classify the hand-signs.

Data Collection and Dataset Generation

We could not find existing datasets for our project and those datasets which we found did not fulfill the requirements of this project. We needed images of hand-signs of Indian Sign Language with filters like grayscale, gaussian blur, and threshold.

The dataset we generated consists of around 600 images of each hand-sign in the training set and 150 images of each hand-sign in the testing set which were captured using the OpenCV (Open Computer Vision) library.
We capture a whole frame at first as shown in image 2:

and then extract an RoI (Region of Interest) from the said frame (the area where the hand-sign is showing) and then apply our filters on the extracted RoI. We first apply the Grayscale filter to convert the extracted RoI to image 6:



The greyscale image is then blurred using the Gaussian Blur filter to give the following image 7:

The blurred grayscale image is then passed through an Adaptive Threshold filter which gives the final desired image which will be used in training the neural network. The final image looks like this:

III. FILTERS APPLIED

1. Grayscale Filter

In OpenCV, images are converted to grayscale using the cvtcolor() function. The cvtcolor() function is used for conversion of images between color spaces. In python, it is

International Journal of Engineering Applied Sciences and Technology, 2021
Vol. 5, Issue 12, ISSN No. 2455-2143, Pages 135-139
Published Online April 2021 in IJEAST (http://www.ijeast.com)

called using the following syntax: cv2.cvtColor(src, code[, dst[, dstCn]]) For converting the images to grayscale, we use the following syntax:

dst = cv2.cvtColor(src, cv2.CV_BGR2GRAY)

The conversion happens by removing the alpha channels using the following method:
Y <– 0.299.R + 0.587.G + 0.114.B

2. Threshold Filter

The threshold filter is used to give a binary image from a grayscale image in OpenCV. The filter converts the image to threshold using the following formula:
dst(x,y) = maxValueif src(x, y) > T (x, y)0otherwise The Adaptive Threshold filter can be used in the code using the syntax:

dst = cv2.adaptiveThreshold(src, maxValue, adaptiveMethod, thresholdType, blockSize, C[, dst])

3. Blur Filter

For blurring the image, we use a Gaussian Blur filter. The Gaussian blur filter combines the image to be blurred with a Gaussian kernel.
The syntax for using a Gaussian Blur filter is:

dst = cv2,GaussianBlur(src, ksize, sigmaX[, dst[, sigmaY[, borderType]]])

The parameter definition is as follows:
src is the input image that is to be blurred,
dst is the output variable in which the blurred image will be stored,
ksize is the size of the gaussian kernel,
sigmaX and **sigmaY** are both standard deviations for the Gaussian kernel in the X and Y direction respectively.
maxValue decides which algorithm will we be using for thresholding

thresholdingType is the type of thresholding which we will be using like THRESH_BINARY_ or THRESH_BINARY and
blocksize is the number of pixels in the proximity of the selected pixel which will be used to calculate the value of the threshold for the pixel.
code is the code used for conversion of color space and
dstcn is the number of channels we want to include in the destination image.

IV. GESTURE RECOGNITION

This algorithm includes the steps for capturing the image and processing these captured images for sending them to the neural network.

- Capture the image from the web camera of the computer using the OpenCV library.
- Process the image and apply the grayscale, blur, and threshold filters using the OpenCV library.
- Input the processed images to the trained Convolutional Neural Network and classify them.

Activation Function

The activation function used in these layers is Rectified Linear Unit (aka. ReLu). This activation function calculates a max(x, 0) for each one of the pixels in the input. This helps the network in learning complicated features by adding irregularity to the learning formula. Activation function also plays a part in improving the time complexity of the process.

Dropout Layers

Overfitting is a common problem when training neural networks. It means that a neural network gets so used to (tuned) to the data provided as input in the training dataset that when new data in the testing set is provided, the performance decreases dramatically. The function of dropout layers is to drop a certain percentage of the activations in the training set so that the network can provide the right results even if random activations are dropped out from the dataset. In other words, this means training the neural network to correctly classify the input even if the input is slightly different from the training data.

Optimizer

When the advantages of two stochastic gradient descent algorithms (Root Mean Square Propagation and Adaptive Gradient Algorithm in this case) are combined, we get the Adam optimizer which we have used in our network.

Sentence formation and hand-sign recognition

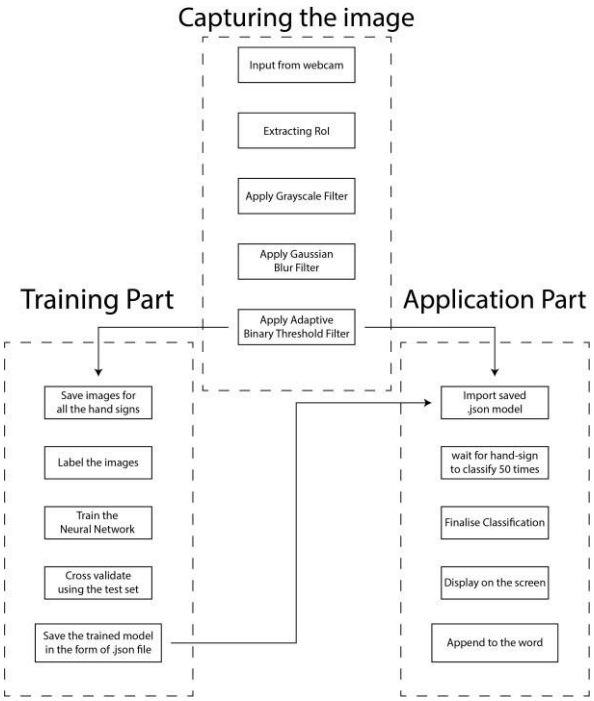
For recognizing hand signs, we use two values, one the count and the other the threshold to avoid any confusion. In our program, the count value is 50 and the threshold value is 20 which means that when a certain hand-sign is classified as a letter overcount (50) times, and the same hand-sign has not been classified as another letter more times than the threshold (20) value, then the letter is correctly recognized. If the value of the threshold is greater than 20, then the current buffer value (dictionary) is cleared so that the next hand-sign does not use the previous values for classification. Every time a blank space is classified correctly, then the current word recognized is added to the sentence field below the word recognition area.

International Journal of Engineering Applied Sciences and Technology, 2021
Vol. 5, Issue 12, ISSN No. 2455-2143, Pages 135-139
Published Online April 2021 in IJEAST (http://www.ijeast.com)

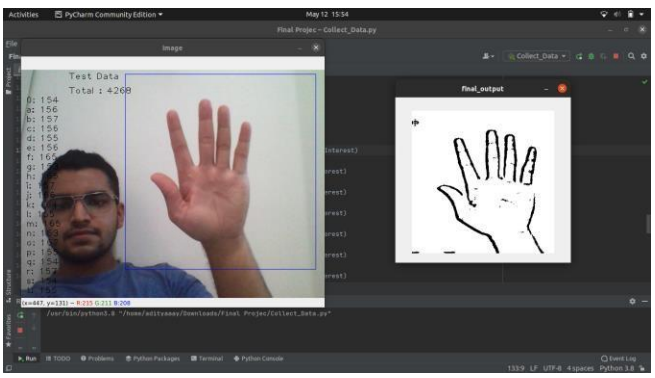
5.3 Training and Testing

First, the image extracted from the RoI (Region of Interest) is blurred slightly using the Gaussian Blur filter. This is done to remove any unnecessary noise which might cause problems when applying the threshold filter. After this, a grayscale filter is applied to remove any alpha channels, and then finally an adaptive threshold filter is applied to convert the image into a binary image. A Softmax function is used to normalize the output in a range of 0 and 1 in such a way that the sum of all the values in each class adds up to 1. The performance measurement used in training this neural network is cross-entropy. Cross entropy is a continuous function in which the value is positive for incorrectly classified values and is exactly 0 at correctly classified values. TensorFlow has an inbuilt function that automatically calculates the cross-entropy. To correctly classify as many hand signs as possible, we optimized the cross-entropy to be minimum by tweaking the weights of the neurons. The cross-entropy calculated was minimized by using the gradient descent Adam Optimizer as discussed above.

V. DETAILED DESIGN (DATA FLOW DIAGRAM)



DATA COLLECTION PROGRAM RUNNING



OUTPUT APP RUNNING

VI. CONCLUSION

This Project is a working system for the conversion of Indian Sign Language to text in real-time for deaf and mute people. For now, this project is focused on the single spelling-based signs where each alphabet is represented by a particular hand sign. The accuracy achieved is 97.3% on our testing dataset. The approach we have used is that the training data has some background noise like patterned clothes other objects in the background which might be present in real-world scenarios. However, the distance dataset has negligible background noise adequate lighting. This way we can train the model to ignore the background noise focus only on the hand signs and gestures, therefore getting better results.

VII. REFERENCES

[1] Byeongkeun Kang, Subarna Tripathi, T. Q. N. (2015). "Real-time sign language fingerspelling recognition using convolutional neural networks from the depth map." 2015 3rd IAPR Asian Conference on Pattern Recognition (ACPR). 10.1109/ACPR.2015.7486481

[2] Lin, H.-I., Hsu, M.-H., and Chen, W.-K. (2014). "Human hand gesture recognition using a convolution neural network." IEEE International Conference on Automation Science and Engineering,10.1109/CoASE.2014.6899454.

[3] Mukai, N., Harada, N., and Chang, Y. (2017). "Japanese fingerspelling recognition based on classification tree and machine learning.", 10.1109/NICOINT.2017.9, 19–24.

[4] Pigou, L., Dieleman, S., Kindermans, P.-J., and Schrauwen, B. (2015). "Sign language recognition using convolutional neural networks." 8925, 10.1007/978-3-319-16178-5_40, 572–578.

[5] Tie, Y. and Xu, Y. (1994). "Hidden Markov model for gesture recognition." CARNEGIE MELLON UNIV PITTSBURGH PA ROBOTICS INST, 10.21236/ada282845.

[6] Zaki, M. and Shaheen, S. (2011). "Sign language recognition using a combination of new vision-based features." Computer Engineering Department, Cairo University,Cairo,Egypt,10.1016/j.patrec.2010.11.01 3 572–577.

[7] Ziaie, P., Müller, T., Foster, M. E., and Knoll, A. (2008). "A Naïve Bayes classifier with distance weighting for hand-gesture recognition." Technical University of Munich, Dept. of Informatics VI, Robotics and Embedded Systems,10.1007/978-3- 540-89985-3_38 , 308–315.