Indian Sign Language Recognition using Convolutional Neural Network

Rachana Patil[1,*], *Vivek* Patil[1,**], *Abhishek* Bahuguna[1,***], and *Mr. Gaurav* Datkhile[1,****]

[1]Ramrao Adik Institute Of Technology,Nerul, India

Abstract. Communicating with the person having hearing disability is always a major challenge. The work presented in paper is an exertion(extension) towards examining the difficulties in classification of characters in Indian Sign Language(ISL). Sign language is not enough for communication of people with hearing ability or people with speech disability. The gestures made by the people with disability gets mixed or disordered for someone who has never learnt this language. Communication should be in both ways. In this paper, we introduce a Sign Language recognition using Indian Sign Language.The user must be able to capture images of hand gestures using a web camera in this analysis, and the system must predict and show the name of the captured image. The captured image undergoes series of processing steps which include various Computer vision techniques such as the conversion to gray-scale, dilation and mask operation. Convolutional Neural Network (CNN) is used to train our model and identify the pictures. Our model has achieved accuracy about 95%

Keywords: Convolutional Neural Network(CNN), Hand Gesture, Deaf people, Sign Language, Sign language Recognition(SLR), ISL(Indian Sign Language).

1 Introduction

One of the most important requirements for social survival is communication. Deaf and dumb peoples communicate with one another using sign language, but it is difficult for non-deaf and dumb people to understand them. While much study has been done on the recognition of American sign language, Indian sign language varies greatly from American sign language. ISL communicates with two hands (20 out of 26), while ASL communicates with a single hand. Because of the overlapping of hands when using both hands, features are often obscured. Furthermore, a lack of datasets, combined with the fact that sign language varies depending on location, has limited ISL gesture detection efforts. This paper aims to take the first step in using Indian sign language to bridge the communication gap between normal people and deaf people. The extension of this project to words and common phrases will not only make it easier for deaf and dumb people to communicate with the outside world, but it may also help in the development of autonomous systems for understanding and assisting them.

The aim of this paper is to use the corresponding gesture to recognise alphabets in Indian Sign Language. The identification of gestures and sign languages is a well-studied subject in American Sign Language, but it has received little attention in Indian Sign Language. We want to solve this issue, but instead of using high-end technologies like gloves or the Kinect, we want to recognise gestures from photographs (which can be accessed from a webcam), and then use computer vision and machine learning techniques to extract specific features and classify them.

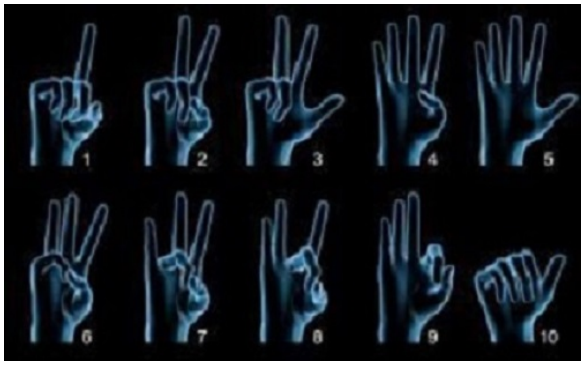

Figure 1. Indian Sign Language Alphabets.

1.1 Problem statement

Understanding the precise meaning of deaf and dumb people's symbolic gestures and converting it into understandable language(Text).

2 Literature Survey

A analysis of the literature for proposed framework reveals that many attempts have been made to tackle sign recognition in videos and images using various methods and algorithms.

*e-mail: rachanakp22@gmail.com
**e-mail: vivek.a.patil202@gmail.com
***e-mail: abhigolu1229@gmail.com
****e-mail: gaurav.datkhile@rait.ac.in

In Jing-hao Sun[1] The human hand was separated from the complex context, and the CamShift algorithm was used to detect real-time hand gestures. Then, using a convolutional neural network, the region of hand movements that was observed in real time is recognised, resulting in the identification of 10 common digits. The proposed system has dataset of total 1600 pictures for training dataset, 4000 hand gesture, 400 images for each type. This experiment shows accuracy about 98.3 percent. Hasan[2] used scaled normalisation to recognise gestures using brightness factor matching. With a black background, thresholding techniques are used for segmenting the input images . At the X and Y axis origins, the coordinates of any segmented image are shifted to match the centroid of the hand unit. and the image's centre mass is determined. Using a boundary histogram, Wysoski et al[3]. provided rotation invariant postures. The input image was captured with a camera, a filter for skin colour detection was applied, and then a clustering procedure was used to find the border line of each category in the pooling image using a standard contour tracking algorithm. Grids were created from the picture, and the boundaries were normalised. Geethu Nath and Arun C.S. [6] developed an ASL symbol recognition system based on the ARM CORTEX A8 processor. The machine recognises numbers using the Jarvis algorithm and alphabets using the template matching algorithm.Using Principal Component Analysis (PCA) and various distance classifiers, Kumud Tripathi [7] developed a framework for recognising continuous ISL gestures. The features from the keyframes are extracted from the own data set using Orientation Histogram and provided as input to the device.Noor Tubaiz [8] proposed using the Modified k-Nearest Neighbor (MKNN) approach to classify sequential data. Data gloves are used to detect hand movements.To supplement the raw data, window-based statistical features are calculated from previous raw feature vectors and future raw feature vectors. To recognise terms in ISL, the proposed framework was developed using novel techniques based on existing systems (ISL).Describe an approach for a continuous sign language recognition method (B. Bauer et al.). It is a framework that depends on continuous hidden Markov models images (HMM). It employs German sign language (GSL). Feature vectors that represent manual signs are fed into the device. [9]

3 Proposed Methodology

3.1 Image Aquistion:

It is the action of extracting an image from a source, typically a hardware-based source, for process of image processing. WebCamera is the hardware-based source in our project. It is the first step in the workflow sequence because no processing can be done without an image. The picture that is obtained has not been processed in any way.

3.2 Segmentation:

The method of separating objects or signs from the context of a captured image is known as segmentation. Con-

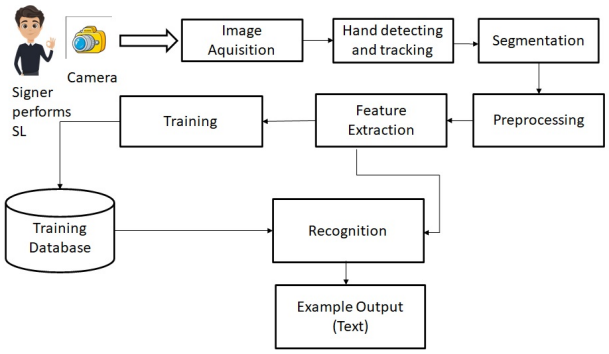

Figure 2. Proposed Methodology.

text subtracting, skin-color detection, and edge detection are all used in the segmentation process. The motion and location of the hand must be detected and segmented in order to recognise gestures.fig:threshold

3.3 Features Extraction:

Predefined features such as form, contour, geometrical feature (position, angle, distance, etc.), colour feature, histogram, and others are extracted from the preprocessed images and used later for sign classification or recognition. Feature extraction is a step in the dimensionality reduction process that divides and organises a large collection of raw data.reduced to smaller, easier-to-manage classes As a result, processing would be simpler. The fact that these massive data sets have a large number of variables is the most important feature. To process these variables, a large amount of computational power is needed. As a result, function extraction aids in the extraction of the best feature from large data sets by selecting and combining variables into functions.reducing the size of the data These features are simple to use while still accurately and uniquely describing the actual data collection.

3.4 Preprocessing:

Each picture frame is preprocessed to eliminate noise using a variety of filters including erosion, dilation, and Gaussian smoothing, among others. The size of an image is reduced when a colour image is transformed to grayscale. A common method for reducing the amount of data to be processed is to convert an image to grey scale.The phases of preprocessing are as follows:

3.4.1 Morphological Transform (Morphological Transform) :

Morphological operations use a structuring feature on an input image to create a similar-sized output image. It compares the corresponding pixel in the input image with its neighbours to determine the value of each pixel in the output image.There are two different kinds of morphological transformations Erosion and Dilation.

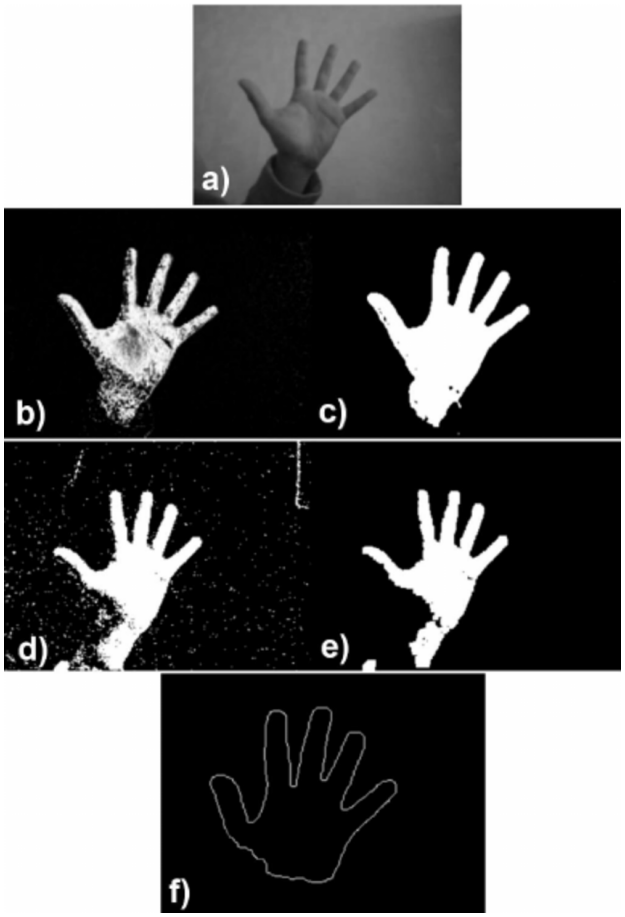

Figure 3. (a) original image with hand; (b) image of hand after skin color detection; (c) after morphological operations and binarization; (d) image of hand after background extraction ; (e) after binarization and morphological operations; (f) hand's contour made of image c and image e concatenation

i. Dilation: The maximum value of all pixels in the neighbourhood is the value of the output pixel. A pixel in a binary image is set to 1 if all of its neighbours have the value 1 Morphological dilation increases the visibility of artefacts and fills in small gaps.

ii. Erosion: The o/p pixel's value is the minimum of all pixels in the neighbourhood. A pixel in a binary image is set to 0 if all of its neighbours have the value 0.small artefacts are eroded away by morphological erosion, leaving behind substantial objects.

3.4.2 Blurring:

Adding a low-pass filter to an image is an example of blurring. The word "low-pass filter" refers to eliminating noise from an image while leaving the rest of the image intact in computer vision. A blur is a simple operation that must be completed before other tasks such as edge detection.

3.4.3 Thresholding:

Thresholding is a form of image segmentation in which the pixels of an image are changed to make it easier to interpret the image. Thresholding is the process of converting a colour or grayscale image into a binary image, which is simply black and white. We most commonly use thresholding to pick areas of interest in a picture while ignoring the sections we are not concerned with.

3.4.4 Recognition:

We'll use classifiers in this case. Classifiers are the methods or algorithms that are used to interpret the signals. Popular classifiers that identify or understand sign language include the Hidden Markov Model (HMM), K-Nearest Neighbor classifiers, Support Vector Machine (SVM), Artificial Neural Network (ANN), and Principle Component Analysis (PCA), among others. However, in this project, the classifier will be CNN. Because of its high precision, CNNs are used for image classification and recognition. The CNN uses a hierarchical model that builds a network, similar to a funnel, and then outputs a fully-connected layer in which all neurons are connected to each other and the output is processed.

3.4.5 Text output:

Understanding human behaviour and identifying various postures and body movements, as well as translating them into text.

4 Proposed Algorithm

4.1 Creating the sign language recognition dataset:

Any frame that detects a hand within the ROI (region of interest) generated can be transferred to a directory that contains a pair of directories, train and take a look at, every containing 10 folders containing images captured mistreatment the produce gesture knowledge.py perform. Now, to create the dataset, we have a tendency to use OpenCV to urge the live cam feed and create a ROI, that is solely the portion of the frame wherever we have a tendency to want to find the hand for the gestures. For differentiating between the background we have a tendency to calculate the accumulated weighted avg for the background then cipher this from the frames that contain some object ahead of the background which will be distinguished as foreground.This can be accomplished by computing the accumulated weight for specific frames and the context's accumulated average.After we've the accumulated average for the background,we tend to subtract it from each frame that we read after sixty frames to seek out any object that covers the background.

4.2 Calculate threshold value:

We now measure the threshold value for each frame and evaluate the contours using cv.findContours.The max contours i.e. object's most outermost contours is returned using function segment. We may decide whether there is

some foreground object identified in the ROI using the contours, in other words, whether there is a hand in the ROI. When model detects contours, it starts saving the picture of the ROI in the train and test sets for the letter or number we're looking for (or a hand is present in the ROI). The dataset for 1 is generated in the preceding example, and the ROI's thresholded image is displayed in the following window.In the train dataset, we save 600 images for each number to be detected, and in the test dataset, we generate 80 images for each number.

4.3 CNN Layer:

To classify the static images in our first dataset, we used a Convolutional Neural Network, or CNN, model. While building neural network our main objective was to define input layer. Each of the 784 pixels in a 28x28 image is represented by a grayscale value ranging from 0 (black) to 1 (white) (white). We translate the data into a computer-readable format by converting each image to a series of numbers. The hidden layers of the neural network will process the input layer once it has been prepared. Our neural network's architecture can be seen in the fig:blk2 The first hidden layer is made up of a number of nodes, each of which receives a weighted total of the 784 input values. The inputs is then passed through an activation function named as rectified linear unit, or ReLU. The ReLU will produce 0 when the input is negative, but will not alter the input otherwise, as seen in the graph above. The ReLU's outputs will be used as inputs for the network's next hidden layer.

Table 1. CNN Layers

Layer (type)	Output shape	Param#
conv2d(Conv2D)	(None, 62, 62, 32)	896
max_pooling2d (MaxPooling2D)	(None, 31, 31, 32)	0
conv2d_1 (Conv2D)	(None, 31, 31, 64)	18496
max_pooling2d_1 (MaxPooling2D)	(None, 15, 15, 64)	0
conv2d_2(Conv2D)	(None, 13, 13, 128)	73856
max_pooling2d_2 (MaxPooling2D)	(None, 6, 6, 128)	0
flatten (Flatten)	(None, 4608)	0
dense (Dense)	(None, 64)	294976
dense_1 (Dense)	(None, 128)	8320
dropout_1 (Dropout)	(None, 128)	0
dense_2 (Dense)	(None, 128)	16512
dropout_2 (Dropout)	(None, 128)	0
dense_3 (Dense)	(None, 12	1548

4.4 Training CNN:

We now train a CNN on the newly generated data collection.To begin, we'll use keras ImageDataGenerator, which lets us use the flow from directory function to load the train and test set data, with the names of the number folders serving as the class names for the images loaded.Reduced LR on plateau and early training callbacks are used, all of which are based on the validation dataset failure. The validation dataset is used to measure the accuracy and loss after each epoch, and if the validation loss is not decreasing, the model's LR(Learning Rate) is reduced using ReduceLR to prevent the model from overshooting the loss

Epoch No.	Loss	Accuracy	Val_Loss	Val_Acc
1	12.71	0.18	0.81	0.8
2	1.44	0.53	0.30	0.96
3	0.84	0.71	0.15	0.99
4	0.56	0.81	0.02	1.0
5	0.38	0.86	0.01	1.0
6	0.30	0.88	0.0059	1.0
7	0.25	0.89	0.00	1.0
8	0.27	0.90	0.00	1.0
9	0.16	0.94	0.00	1.0
10	0.15	0.95	0.00	1.0

minima. we also use the early stopping algorithm to stop the training if the validation accuracy continues to decrease for some epochs. The two optimization algorithms used are SGD i.e. stochastic gradient descent which means the weights are changed at any training instance and Adam i.e. a combination of Adagrad and RMSProp. We discovered that the model SGD had higher accuracies. As can be seen, we achieved 100 training accuracy and an 81 percent validation accuracy while training.

4.5 Predicting the gesture

We generate a bounding box for detecting the ROI and measuring the cumulative average, just as we did when creating the dataset. This is done in order to recognize a foreground entity. Now we look for the maximum contour, and if one is found, it means a hand has been identified, so the ROI's threshold is used as a test picture. Using keras.models.load_ Model we load the previously saved model and then feed the threshold picture of the ROI containing the hand to the model for prediction as an input.

5 Result:

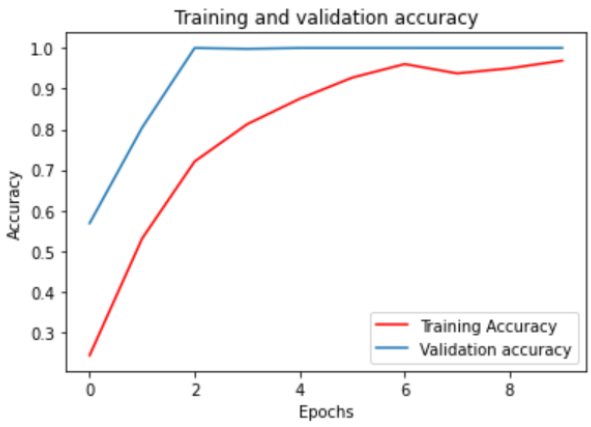

Figure 4. Outputs from the first hidden layer.

When we are training the model, accuracy and loss in model for validation data could be variating with various cases. Normally with every increasing epoch, loss should be going lower and exactness should be going higher.But with validation loss(keras validation loss) and validation

accuracy, numerous cases can be conceivable like underneath:
1)Validation loss starts increasing, validation accuracy starts diminishing. This implies model is cramming values not learning.
2)Validation loss starts increasing, validation accuracy likewise will increase. This could be instance of overfitting or diverse probability values in situations where softmax is being used in output layer.
3)Validation loss starts decreasing, validation accuracy starts increasing. This is additionally fine as that implies model fabricated is learning and dealing fine. After testing our model we have aquired the following results,we have plotted the graph of accuracy and loss with respect to epochs.

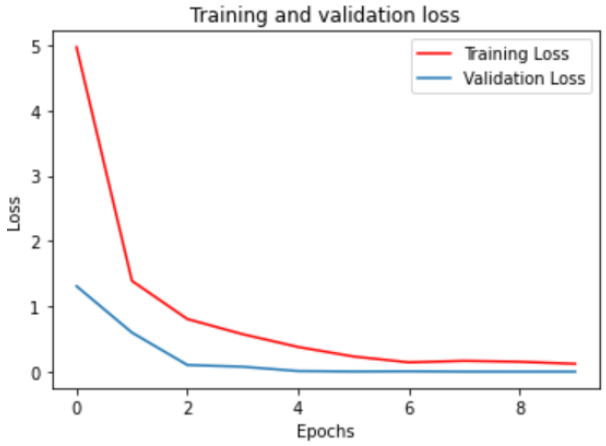

Figure 5. Training and validation loss

Fig 6 shows overall accuracy evolution of model. In which it has been seen that validation loss is decreasing and validation accuracy is increasing noticeably.

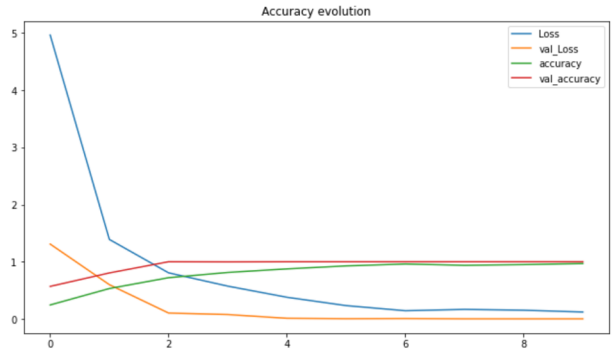

Figure 6. Accuracy Evolution

6 Conclusion and Future Work:

The Sign Language Recognition (SLR) system is a method for recognising a collection of formed signs and translating them into text or speech with the appropriate context. The significance of gesture recognition can be seen in the development of effective human-machine interactions. We attempted to build a model using a Convolutional Neural Network in this project. This results in a validation accuracy of about 95%
The Image Processing section of future work should be enhanced so that the system can interact in both directions, i.e. it should be capable of translating normal language to sign language and vice versa.

References

[1] Jing-Hao Sun,Ting-Ting Ji, Shu-Bin Zhang, Jia-Kui Yang, Guang-Rong Ji "Research on the Hand Gesture Recognition Based on Deep Learning",07 February 2019

[2] Mokhtar M. Hasan, Pramoud K. Misra, (2011). “Brightness Factor Matching For Gesture Recognition System Using Scaled Normalization”, International Journal of Computer Science Information Technology (IJCSIT), Vol. 3(2).

[3] Simei G. Wysoski, Marcus V. Lamar, Susumu Kuroyanagi, Akira Iwata, (2002). “A Rotation Invariant Approach On Static-Gesture Recognition Using Boundary Histograms And Neural International Journal of Artificial Intelligence Applications (IJAIA), Vol.3, No.4, July 2012

[4] Stergiopoulou, N. Papamarkos. (2009). “Hand gesture recognition using a neural network shape fitting technique,” Elsevier Engineering Applications of Artificial Intelligence, vol. 22(8), pp. 1141– 1158, doi: 10.1016/j.engappai.2009.03.008

[5] V. S. Kulkarni, S.D.Lokhande, (2010) “Appearance Based Recognition of American Sign Language Using Gesture Segmentation”, International Journal on Computer Science and Engineering (IJCSE), Vol. 2(3), pp. 560-565.

[6] Geethu G Nath and Arun C S, "Real Time Sign Language Interpreter," 2017 International Conference on Electrical, Instrumentation, and Communication Engineering (ICEICE2017)

[7] Kumud Tripathi, Neha Baranwal and G. C. Nandi, "Continuous Indian Sign Language Gesture Recognition and Sentence Formation", Eleventh International MultiConference on Information Processing-2015 (IMCIP-2015), Procedia Computer Science 54 (2015) 523 – 531

[8] Noor Tubaiz, Tamer Shanableh, and Khaled Assaleh, "Glove-Based Continuous Arabic Sign Language Recognition in User-Dependent Mode," IEEE Transactions on Human-Machine Systems, Vol. 45, NO. 4, August 2015.

[9] B. Bauer,H. Hienz "Relevant features for video-based continuous sign language recognition", IEEE International Conference on Automatic Face and Gesture Recognition, 2002.

[10] Pigou, L., Dieleman, S., Kindermans, P.-J., Schrauwen, B. (2014). Sign Language Recognition Using Convolutional Neural Networks.