J.K. Institute of Applied Physics and Technology

(Department of Electronics & Communication, University of Allahabad)

Sign Language to Text Conversion for Deaf and Dumb

Session: (2020-2022)

By,

Avtar Chandra – 2104754 (Enroll. no)

Under the supervision of:

Prof. Shiv Prakash

A Thesis on

Sign Language to Text Conversion for Deaf and Dumb

Submitted In Partial Fulfillment of the Requirement
For The Award of the Degree of
MASTER OF TECHNOLOGY
IN
COMPUTER SCIENCE OF TECHNOLOGY

Under the Supervision of:
PROF. SHIV PRAKASH

Submitted by:
Avtar Chandra
Enrollment no.: U2104754
M.TECH (CSE)
DEPARTMENT OF
ELECTRONICS & COMMUNICTION ENGINEERING
UNIVERSITY OF ALLAHABAD
Session (2020-2022)

Contents:

Abstract:

Sign language is one of the oldest and most natural form of language for communication, but 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 fingerspelling based American Sign Language. In our method, the hand is first passed through a filter and after the filter is applied the hand is passed through a classifier which predicts the class of the hand gestures. Our method provides **95.7 %** accuracy for the 26 letters of the alphabet.

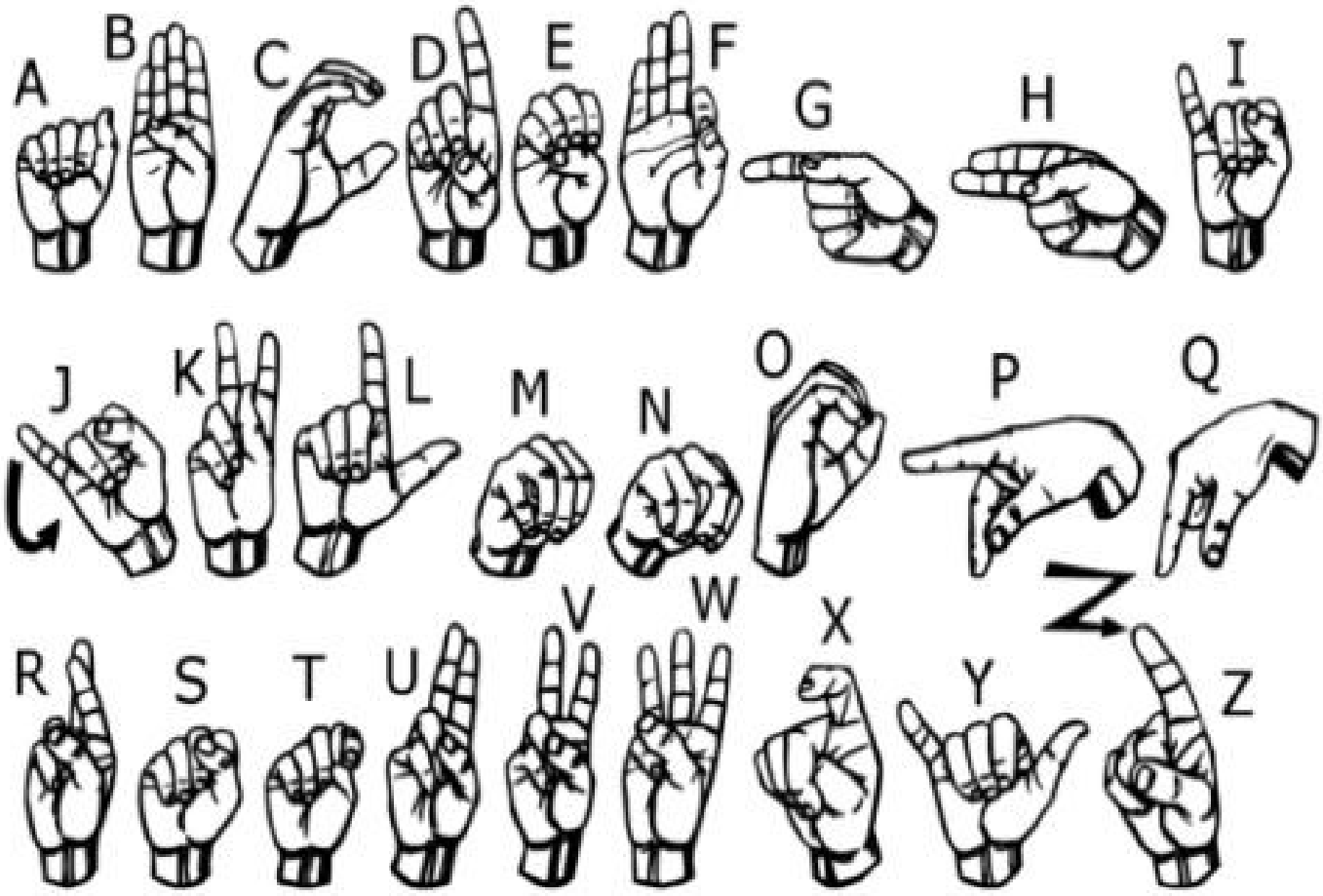
Introduction:

American Sign Language is a predominant sign language since the only disability D&M people have is communication related and they cannot use spoken languages hence the only way for them to communicate is through sign language. Communication is the process of exchange of thoughts and messages in various ways such as speech, signals, behavior and visuals. Deaf and dumb (D&M) people make use of their hands to express different gestures to express their ideas with other people. Gestures are the nonverbally exchanged messages and these gestures are understood with vision. This nonverbal communication of deaf and dumb people is called sign language.

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.

In our project we basically focus on producing a model which can recognize Fingerspelling based hand gestures in order to form a complete word by combining each gesture. The gestures we aim to train are as given in the image below.



Objective:

For interaction between normal people and D&M people a language barrier is created as sign language structure which is different from normal text. So they depend on vision based communication for interaction.

If there is a common interface that converts the sign language to text the gestures can be easily understood by the other people. So research has been made for a vision based interface system where D&M people can enjoy communication without really knowing each other's language.

The aim is to develop a user friendly human computer interfaces (HCI) where the computer understands the human sign language. There are varioussign languages all over the world, namely American Sign Language (ASL), French Sign Language, British Sign Language (BSL), Indian Sign language, Japanese Sign Language and work has been done on other languages all around the worl

Literature Review:

There has been a lot of research and experiments carried over the hand gesture recognition in past decade.
With the help of literature review done we faced the basic steps in hand gesture recognition as:-

- Data-acquisition.
- Data-preprocessing.
- Feature extraction.
- Gesture classification.

Data-acquisition:

The different procedures to acquire information about hand gesture may be executed in following ways:

1. Using sensory devices

It uses electromechanical gadgets to provide exact hand function (position) and configuration. Different glove based strategies may be used to extract records .But it isn't user friendly and steeply-priced.

2. Vision based approach

In vision primarily based strategies computer digital camera is the input tool for looking at the statistics of palms and fingers. The vision primarily based methods require only a digital camera, as aresult knowing a natural interaction between humans and computers without use of any extra devices. Those systems generally tend to supplement organic imaginative and prescient by describing artificial vision systems that are implemented in software and/or hardware.

The main challenge of vision-based hand detection is to cope with the large variability of human hand's appearance due to a huge number ofhand movements, to different skin-colour possibilities as well as to the variations in viewpoints, scales, and speed of the camera capturing the scene.

Data-preprocessing and Feature extraction for vision based approach:

- In [1] the approach for hand detection combines threshold based colour detection with background subtraction. We can use 'Adaboost face detector' to differentiate between faces and hands as both involves similar skin color.
- We can also apply a Gaussian Blur filter to extract the

images needed for training. Filters can be easily applied using Open Computer Vision**,** also known as OpenCV and is described in [3].

- For extracting necessary image which is to be trained we can use instrumented gloves as mentioned in [4]. This will help us reduce computation time for preprocessing and can give us more concise and accurate data in comparison to applying filters on data received from video extraction.

- We tried to segment the image manually using the color segmentation method, but as mentioned in the research article, the segmentation results we tried did not match because the skin color and tone greatly depend on the lighting conditions. The output we achieved by segmentation were not so best fit. Also, many characters that are similar to each other, such as the "V" sign and the gesture for the number “2”, needs to be trained for the project, so we decided to improve it in terms of better accuracy for a large number of symbols, rather than segmenting the hand out of a random background we keep background of hand a stable single color so that we don’t need to segment it on the basis of skin color. This would help us to get better results.

Gesture classification:

- In [1], Hidden Markov Models (HMMs) are used to classify gestures. This model addresses the dynamic side of gestures. Gestures are extracted from video image sequences by tracing skin-colored patches corresponding to hands in body-face space centered on the user's face. The goal is to recognize two kinds of gestures. Images are filtered using a quick lookup index table. After filtering, skin-colored pixels are collected as droplets. A blob is a statistical object based on the position (x, y) and colorimetric (Y, U, V) of a skin-colored pixel to define a uniform area.

- In [2], an efficient and fast method for recognizing static hand gestures is the Naive Bayes classifier. It is based on the classification of various gestures according to geometric invariants obtained from image data after segmentation. Therefore, unlike many other recognition methods, this method does not depend on skin color. Gestures are extracted from each frame of the video with a static background. The first step is to classify and label the object of interest and extract geometric invariants from that object. The next step is to perform gesture classification using the K-nearest neighbor

algorithm augmented with the distance weighting (KNNDW) algorithm to provide data suitable for a locally weighted Naive Bayes Classifier.

- According to the paper “Human Hand Gesture Recognition Using a Convolution Neural Network” by Hsien I Lin , Ming-Hsiang Hsu, andWei-Kai Chen the graduates of the Taiwan National Taipei Institute of Technology Automation Technology creates a skin model, extracts hands from an image, and then applies a binary threshold to the entire image. After capturing the threshold image, it is calibrated around the principal axis to center the image. They feed this image to a convolutional neural network model to train and predict the outcome. They trained the model on more than 7 hand gestures and using the model achieved about 95% accuracy for 7 hand gestures.

Key Words and Definitions:

.

Feature Extraction and Representation

The representation of an image as a 3D matrix having dimension as ofheight and width of the image and the value of each pixel as depth (1 in case of Grayscale and 3 in case of RGB). Further, these pixel values are used for extracting useful features using CNN.

Artificial Neural Networks

Artificial Neural Network is a connections of neurons, replicating the structure of human brain. Each connection of neuron transfers information to another neuron. Inputs are fed into first layer of neurons which processes it and transfers to another layer of neurons called as hidden layers. After processing of information through multiple layers of hidden layers, information is passed to final output layer.

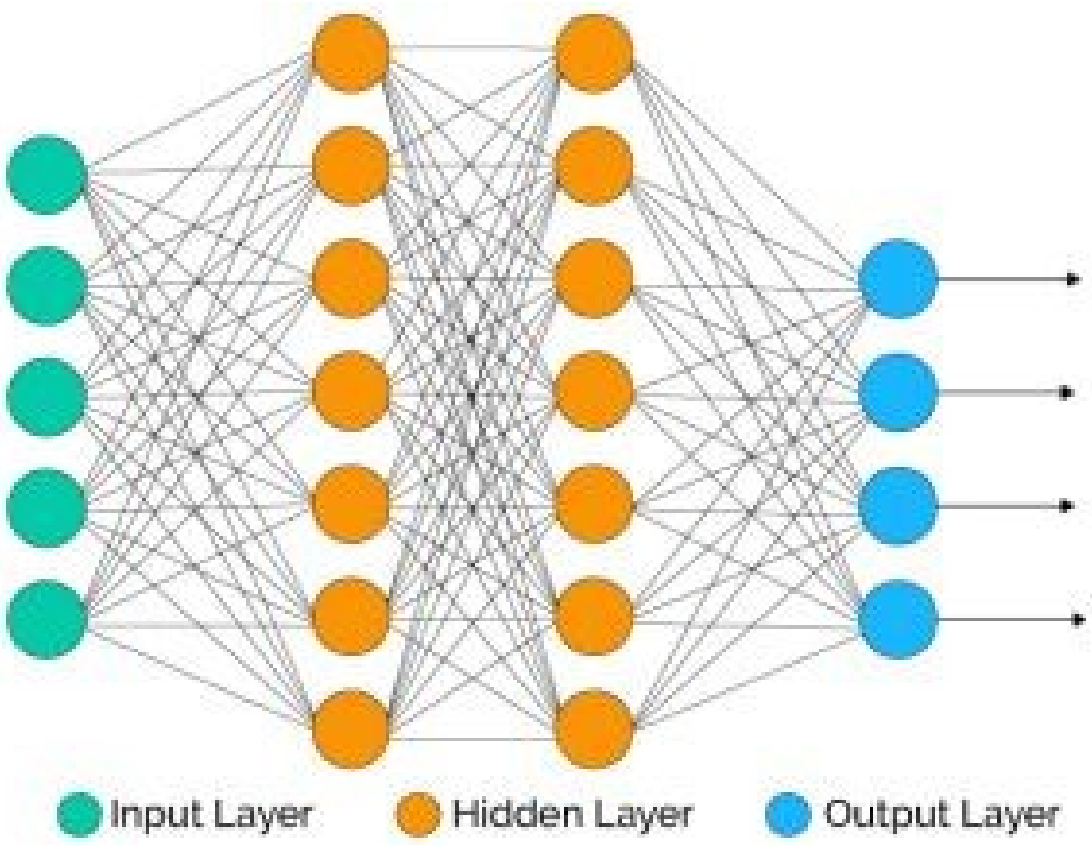


Figure 5.1: Artificial neural networks

They can learn and must be trained. There are different learning strategies:

a. Unsupervised Learning
b. Supervised Learning
c. Reinforcement Learning

Convolution Neural Network

Unlike regular Neural Networks, in the layers of CNN, the neuronsare arranged in 3 dimensions: width, height, depth. The neurons in a layer will only be connected to a small region of the layer (window size) before it, instead of all of the neurons in a fully-connected manner. Moreover, the final output layer would have dimensions (number of classes), because by the end of the CNN architecture we will reduce the full image into a single vector of class scores.

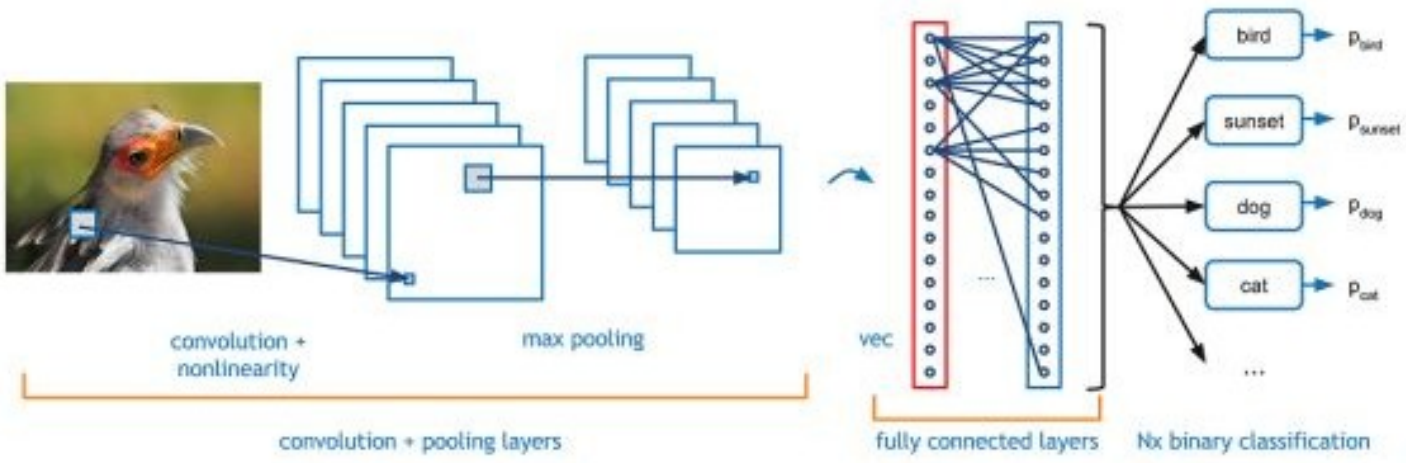


Figure 5.2: Convolution neural networks

1. **Convolution Layer:** In convolution layer we have taken a tiny window size [typically of length 5*5] that extends upto the depth of input matrix. Layers consist of trainable window size filters. During each iteration, we moved the window by a step size [usually 1] and computed the dot product of the filter item and the input value at the given location. Continuing this process results in a two-dimensional activation matrix giving the response of this matrix at each spatial location. In other words, the network learns a filter that activates when it sees some sort of visual element, such as an edge in a certain direction or a speck of a certain color.

2. **Pooling Layer:** We use a pooling layer to reduce the size of the activation matrix and ultimately reduce the trainable parameters. There are two types of pooling:

a) **Max Pooling:** In max pooling we take a window size [for example window of size 2*2], and only take the maximum of 4 values. Well lid this window and continue this process, so well finally get a activation matrix half of its original Size.

b) **Average Pooling:** In average pooling we take average of all values in a window.

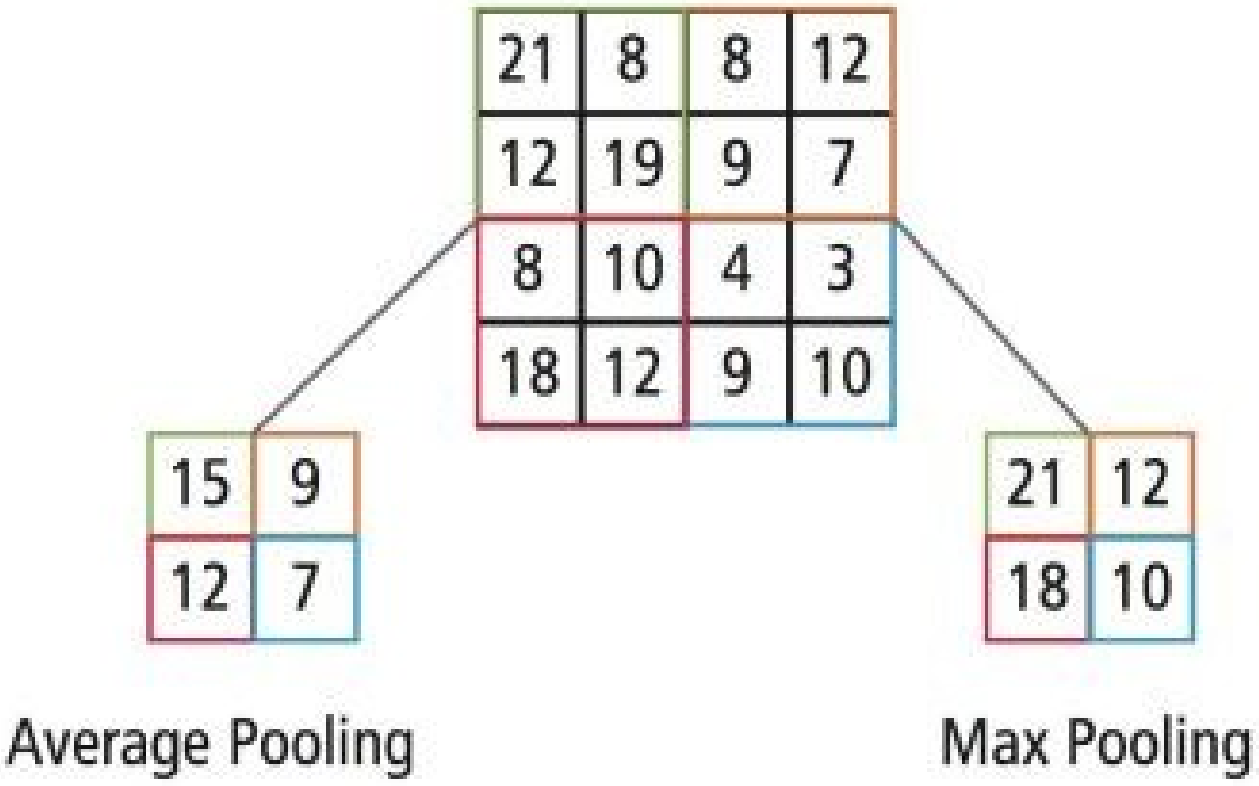


Figure 5.3: Types of pooling

3. Fully Connected Layer:

In convolution layer neurons are connected only to a local region, while in a fully connected region, will connect the all the inputs to neurons.

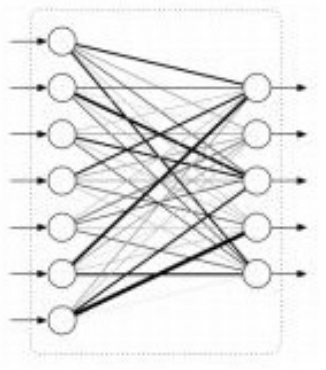


Figure 5.4: Fully Connected Layer

4. Final Output Layer:

After getting values from fully connected layer, will connect them to final layer of neurons [having count equal to total number of classes], that will predict the probability of each image to be in different classes.

TensorFlow

Tensorflow is an open source, freely available software library for numerical computations.First we define the computation graph node and then the actual computation takes place within the session. TensorFlow is widely used in machine learning.

Keras

Keras is a high-level neural networks library written in python that works as a wrapper to TensorFlow. It is used in cases where we want to quickly build and test the neural network with minimal lines of code. It contains implementations of commonly used neural network elements like layers, objective, activation functions, optimizers, and tools to make working with images and text data easier.

OpenCV

OpenCV(Open Source Computer Vision) is an open source library of programming functions used for real-time computer-vision. It is mainly used for image processing, video capture and analysis for features like face and object recognition. It is written in C++ which is its primary interface, however bindings are available for Python, Java, and MATLAB/OCTAVE.

Methodology:

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

Data Set Generation

For the project we tried to find already made datasets but we couldn’t find dataset in the form of raw images that matched our requirements. All we could find were the datasets in the form of RGB values. Hence we decidedto create our own data set. Steps we followed to create our data set are as follows:

We used Open computer vision (OpenCV) library in order to produce our dataset. Firstly we captured around 800 images of each of the symbol in ASL for training purposes and around 200 images per symbol for testing purpose.

First, we capture each frame shown by the webcam of our machine. In the each frame we define a region of interest (ROI) which is denoted by a blue bounded square.

From this whole image we extracted our ROI which is RGB and convert it into gray scale Image as shown below.

Finally we apply our gaussian blur filter to our image which helps us extracting various features of our image. The image after applying gaussian blur looks like below.

GESTURE CLASSIFICATION

The approach which we used for this project is:

Our approach uses two layers of algorithm to forecast the final gesture of theuser.

Algorithm Layer - 1:

1. Apply Gaussian blur filter and threshold to frames captured with OPENCV to obtain processed images after feature extraction.
2. This processed image is passed to the CNN model

for prediction, and if a character is found in more than 50 frames, it is printed and taken into account when constructing a word.

3. Spaces in-between words are to be considered by using the blank symbol.

Algorithm Layer - 2:

1. We detect various sets of symbols which show similar results on getting detected.
2. We then classify between those sets using classifiers made for those sets only.

Layer 1:

CNN Model:

1. **1st Convolution Layer:**

   The resolution of the original image is 128x128 pixels. It is processed first in the first convolutional layer with 32 filter weights (3x3 pixels each). The result is a 126X126 pixel image, one for each filter weight.

2. **1st Pooling Layer:**

Images are down sampled using 2x2 max pooling. That is, it stores the largest value in the square of a 2x2 array. So the image is reduced to 63x63 pixels.

3. **2nd Convolution Layer :**

Now these 63 x 63 from the output of the first full layer are used as the input of the second convolutional layer. It is processed in the second convolutional layer with 32 filter weights (3x3 pixels each). The result is a 60 x 60 pixel image.

4. **2nd Pooling Layer:**

The resulting image is down sampled again using up to a 2x2 pool and down sampled to a 30x30 image resolution.

5. **1st Densely Connected Layer:**

Now this images are used as input to a fully connected layer with 128 neurons and the output of the second convolutional layer is transformed into 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 a second tightly coupled layer. We have used a dropout layer of 0.5 to avoid over fitting.

6. **2nd Densely Connected Layer:**

Now, the output of the 1st Densely Connected Layer are to be used as input to a fully connected layer along with 96 neurons.

7. **Final layer:**

The output of the tightly coupled second layer is used as the input of the last layer, which has the same number of neurons as the number of classes we have classified (alphabets + blank symbol).

Activation Function:

We have used ReLu (Rectified Linear Unit) in each of the layers (convolutional as well as fully connected neurons). ReLu calculates max(x, 0) for each input pixel. This adds nonlinearity to the formula and helps to learn more complicated features. It helps in removing the vanishing gradient problem andspeeding up the training by reducing the computation time.

Pooling Layer:

We apply **Max** pooling to the input image with a pool size of (2, 2) with reLU activation function. This reduces the amount of parameters thus lessening the computation cost and reduces over fitting.

Dropout Layers:

The problem of over fitting, where after training, the weights of the network are so tuned to the training examples they are given that the network doesn't perform well when given new examples. This layer "drops out" a random set of activations in that layer by setting them to zero. The network should be able to provide the right classification or output for a specific example even if some of the activations are dropped out [5].

Optimizer:

We have used Adam optimizer for updating the model in response to the output of the loss function. Adam combines the advantages of two extensions of two stochastic gradient descent algorithms namely adaptive gradient algorithm (ADA GRAD) and root mean square propagation (RMSProp)

Layer 2:

We are hereby using two layers of algorithms to verify and predict symbols which are more similar to each other so that we can get us close as we can get to detect the symbol shown. In our testing we found that following symbols were creating ambiguity even not showing properly and were giving other symbols also:

1. For D : R and U
2. For U : D and R
3. For I : T, D, K and I
4. For S : M and N

So, to handle above occurring cases we are making 3 different classifiers to classify these sets:

1. [D, R, U]
2. [T, K, D, I]
3.] S, M, N]

Finger spelling sentence formation Implementation:

1. Whenever the number of detected characters exceeds a certain value and there are no other characters close to the threshold, we print the character and append it to the current line (in our code we stored the value as 50 and the difference threshold as 20).

2. Otherwise, clear the current

dictionary with the number of occurrences of the current character to avoid the possibility of predicting an invalid character.

3. Whenever the number of detected blanks (normal background) exceeds a certain value, no blanks are detected if the current buffer is empty.

4. Otherwise, it will print a space to predict the end of a word, and the current word will be added to the sentence below.

Autocorrect Feature:

A python library **Hunspell suggest** is used to suggest correct alternatives for each (incorrect) input word and we display a set of words matching the current word in which the user can select a word to append it to the current sentence. This helps in reducing mistakes committed in spellings and assistsin predicting complex words.

Training and Testing:

Convert the input image (RGB) to grayscale and apply a Gaussian blur to remove unwanted noise. Apply an adaptive threshold to extract the hand from the background and resize the image to 128 x 128.

After applying all the operations mentioned above, we pass the pre-processed input images to the model for training and testing

.

The prediction level estimates the likelihood that an image belongs to one of the classes. So the output is normalized between 0 and 1 so that all values in each class sum to 1. We achieved this with the softmax function.

Initially, the output of the prediction layer will be slightly different from the actual value. To improve this, we trained a network using labeled data. Cross entropy is a

performance measure used for classification. A continuo us function that is positive for values that do not match the labeled value and is exactly zero when the label is equal to the specified value. Therefore, we optimized the cross entropy by minimizing it as close to zero as possible. To do this, we adjust the weights

of the neural network at the network layer. TensorFlow has a built-in function to compute the cross entropy.

Since discovering the cross entropy function, we have optimized it using gradient descent. In fact, the best gradient descent optimizer is called Adam Optimizer.

Challenges Faced:

There were many challenges faced by us during the project. The very first issue we faced was of dataset. We wanted to deal with raw images and that too square images as CNN in Keras as it was a lot more convenient working with only square images. We couldn't find any existing dataset for that hence we decided to make our own dataset. Second issue was to select afilter which we could apply on our images so that proper features of the images could be obtained and hence then we could provide that image as input for CNN model. We tried various filter including binary threshold, canny edge detection, gaussian blur etc. but finally we settled with gaussian blur filter. More issues were faced relating to the accuracy of the model we trained in earlier phases which we eventually improved by increasing the input image size and also by improving the dataset.

Results:

We have achieved an accuracy of 95.8% in our model using only layer 1 of our algorithm, and using the combination of layer 1 and layer 2 we achieve an accuracy of 98.0%, which is a better accuracy then 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 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 uses any background subtraction algorithm whiles some of the models present above 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 audience to buy and our model uses a normal webcam of the laptop hence it is great plus point.

Below are the confusion matrices for our results:

		Predicted Values																							
		A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y
Correct Values	A	147	0	0	0	0	0	0	0	0	0	0	0	1	2	0	0	0	0	0	0	0	0	2	0	0
	B	0	139	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	11	0	0	0
	C	0	0	152	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	D	0	0	0	145	0	0	0	0	0	0	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	E	0	0	0	0	152	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	F	0	0	0	0	0	135	0	0	0	0	0	4	0	0	0	0	0	1	0	0	2	10	0	0	0
	G	0	0	0	0	0	0	150	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
	H	1	0	0	0	0	0	7	143	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	1
	I	0	0	0	33	0	0	0	0	108	0	2	0	0	0	0	0	0	0	0	7	1	0	0	0	0
	J	0	0	0	0	0	0	0	0	0	153	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	K	0	0	0	0	0	0	0	0	0	0	153	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	L	0	0	0	0	0	0	0	0	0	0	0	153	0	0	0	0	0	0	0	0	0	0	0	0	0
	M	0	0	0	0	0	0	0	0	0	0	2	0	152	0	0	0	0	0	0	0	0	0	0	0	0
	N	0	0	0	0	0	0	0	0	0	0	0	0	0	152	0	0	0	0	0	0	0	0	0	0	0
	O	0	0	0	0	0	0	0	0	0	0	0	0	0	0	154	0	0	0	0	0	0	0	0	0	0
	P	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	153	0	0	0	0	0	0	0	0	0
	Q	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	2	147	1	0	0	0	0	0	0	0
	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	150	0	0	0	0	0	0	0
	S	0	0	0	0	1	0	0	0	0	0	0	0	0	1	10	0	0	0	132	0	0	0	0	8	0
	T	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	151	0	0	0	0	0
	U	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	35	0	0	115	0	0	0	0
	V	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	151	1	0	0
	W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	149	0	0
	X	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	0	0	0	0	0	0	0	0	148	0
	Y	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	151
	Z	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Algo 1

						Predicted Values																			
		A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y
	A	147	0	0	0	0	0	0	0	0	0	0	0	1	2	0	0	0	0	0	0	0	0	2	0	0
	B	0	139	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	11	0	0	0
	C	0	0	152	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	D	0	0	0	153	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	E	0	0	0	0	152	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	F	0	0	0	0	0	135	0	0	0	0	0	4	0	0	0	0	0	0	0	0	3	10	0	0	0
Correct Values	G	0	0	0	0	0	0	150	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
	H	1	0	0	0	0	0	7	143	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	1
	I	0	0	0	0	0	0	0	0	150	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
	J	0	0	0	0	0	0	0	0	0	153	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	K	0	0	0	0	0	0	0	0	0	0	153	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	L	0	0	0	0	0	0	0	0	0	0	0	153	0	0	0	0	0	0	0	0	0	0	0	0	0
	M	0	0	0	0	0	0	0	0	0	0	2	0	152	0	0	0	0	0	0	0	0	0	0	0	0
	N	0	0	0	0	0	0	0	0	0	0	0	0	0	152	0	0	0	0	0	0	0	0	0	0	0
	O	0	0	0	0	0	0	0	0	0	0	0	0	0	0	154	0	0	0	0	0	0	0	0	0	0
	P	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	153	0	0	0	0	0	0	0	0	0
	Q	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	2	147	1	0	0	0	0	0	0	0
	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	150	0	0	0	0	0	0	0
	S	0	0	0	0	1	0	0	0	0	0	0	0	0	0	10	0	0	0	133	0	0	0	0	8	0
	T	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	151	0	0	0	0	0
	U	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	150	0	0	0	0
	V	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	151	1	0	0
	W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	149	0	0
	X	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	0	0	0	0	0	0	0	0	148	0
	Y	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	151
	Z	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Algo 1 + Algo 2

Conclusion:

In this report, a functional real time vision based American Sign Language recognition for D&M people have been developed for ASL alphabets. We achieved final accuracy of 98.0% on our dataset. We are able to improve ourprediction 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.

Future Scope:

We are planning to achieve higher accuracy even in case of complex backgrounds by trying out various background subtraction algorithms. We are also thinking of improving the preprocessing to predict gestures in low light conditions with a higher accuracy.

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 ¨ıve 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 FingerspellingRecognition 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]https://opencv.org/
[12]https://en.wikipedia.org/wiki/TensorFlow
[13]https://en.wikipedia.org/wiki/Convolutional_neural_network

APPENDIX:

OpenCV

OpenCV (Open Source Computer Vision Library) is released under a BSDlicense and hence it's free for both academic and commercial use. It has C++, Python and Java interfaces and supports Windows, Linux, Mac OS, iOS and Android. OpenCV was designed for computational efficiency andwith a strong focus on real-time applications. Written in optimized C/C++,the library can take advantage of multi-core processing. Enabled with OpenCL, it can take advantage of the hardware acceleration of the underlying heterogeneous compute platform.

Adopted all around the world, OpenCV has more than 47 thousand people ofuser community and estimated number of downloads exceeding 14 million. Usage ranges from interactive art, to mines inspection, stitching maps on theweb or through advanced robotics.

Convolution Neural Network

CNNs use a variation of multilayer perceptron's designed to require minimalpreprocessing. They are also known as shift invariant or space invariant artificial neural networks (SIANN), based on their shared-weights architecture and translation invariance characteristics.

Convolutional networks were inspired by biological processes in that the connectivity pattern between neurons resembles the organization of the animal visual cortex. Individual cortical neurons respond to stimuli only in a restricted region of the visual field known as the receptive field. The receptive fields of different neurons partially overlap such that they cover the entire visual field.

CNNs use relatively little pre-processing compared to other image classification algorithms. This means that the network learns the filters thatin traditional algorithms were hand-engineered. This independence from prior knowledge and human effort in feature design is a major advantage.

They have applications in image and video recognition, recommender systems, image classification, medical image analysis, and natural languageprocessing.

Tensorflow

TensorFlow is an open-source software library for dataflow programmingacross a range of tasks. It is a symbolic math library, and is also used for machine learning applications such as neural networks. It is used for bothresearch and production at Google.

TensorFlow was developed by the Google brain team for internal Google use. It was released under the Apache 2.0 open source library on November9, 2015.

TensorFlow is Google Brain's second-generation system. Version 1.0.0 was released on February 11, 2017. While the reference implementation runs on single devices, TensorFlow can run on multiple CPUs and GPUs (with optional CUDA and SYCL extensions for general-purpose computing on graphics processing units). TensorFlow is available on 64-bit Linux, macOS,Windows, and mobile computing platforms including Android and iOS.

Its flexible architecture allows for the easy deployment of computation across a variety of platforms (CPUs, GPUs, TPUs), and from desktops toclusters of servers to mobile and edge devices.

Thanking you,

AVTAR CHANDRA

Acknowledgement

No project reports ever grows full boom from one's personal head. This report owes its existence to help and guidance of many people and I am humbly thankful to all of them.

First and forever, I would like to express deepest sense of gratitude to project mentor, **Prof. SHIV PRAKASH**, **Department of Electronics and Communication Engineering, JK Institute of Applied Physics and Technology, University of Allahabad**. I was privileged to experience a sustained enthusiastic and involved interest from his side. He has a source of constant inspiration throughout the period of this world. His inspiring guidance and innovative ideas were invaluable towards the completion of this project.

I am extremely grateful to **Prof. R.S. Yadav, Head of Department of Electronics and Communication Engineering, JK Institute of Applied Physics and Technology, University of Allahabad**, for providing necessary facilities in order to complete project work.

Last but not the least, I would like to thanks JK Institute staff members and institute, in general, for extending a helping hand at every junction of need.

AVTAR CHANDRA

Department of Electronics and Communication Engineering
JK Institute of Applied Physics and Technology, Allahabad

Date: _____

CANDIDATE'S DECLARATION

I hereby declare that the work presented in this report titled " **Sign Language to Text Conversion for Deaf and Dumb** " is an authentic record of my own work carried out at **Department Of Electronics And Communication Engineering ,JK Institute Of Applied Physics And Technology, University Of Allahabad** as required for the award of degree of **Master of Technology** in Electronics Engineering, submitted in the JK Institute of Applied Physics an Technology, under the supervision of **Prof. Shiv Prakash**, Department Of Electronics And Communication Engineering, JK Institute Of Applied Physics And Technology, University Of Allahabad.

It does not contain any part of work, which has been submitted for the award of any degree either in this Institute or in other University/Deemed University without proper citation.

AVTAR CHANDRA PANDEY

Department of Electronics and Communication Engineering
JK Institute of Applied Physics and Technology, Allahabad

Ref. Date: _____

CERTIFICATE

This is to certify that the project "Sign **Language to Text Conversion for Deaf and Dumb**" submitted by **Avtar Chandra** (Enroll. no.: **2104754**), to the Department Of Electronics And Communication Engineering, JK Institute Of Applied Physics And Technology, University Of Allahabad, in partial fulfillment of the requirements for the award of the degree of **Master of Technology** in **Computer Science and Engineering** is an authentic work carried out at Department Of Electronics And Communication Engineering, JK Institute Of Applied Physics And Technology, University Of Allahabad by him under my supervision and guidance.

(Prof. R S Yadav)
Head of Department

(Prof. Shiv Prakash)
Supervisor