G H Patel College of Engineering & Technology
(A Constituent College of CVM University)

Action Recognition System

A Mini Project (102040601) Report submitted

In partial fulfilment of the requirements for the Degree of Bachelor of Engineering

In
Information Technology
B. E. , Semester – VI

By

Mr.Armin Vaghasiya(12002080501005)
Ms.Ayushi Patel(12002080501006)

Faculty Guide
Prof. Rahul Patel
Assistant Professor

Academic Year 2022-23 (Even)

Department of Information Technology
G H Patel College of Engineering & Technology
Bakrol Road, Vallabh Vidyanagar

CERTIFICATE

This is to certify that *Mini Project (102040601)* work embodied in this report entitled, "**Action Recognition System**" was carried out by Armin Vaghasiya(12002080501005), Ayushi Patel(12002080501006), at G H Patel College of Engineering & Technology for partial fulfilment of B.E. degree to be awarded by Charutar Vidya Mandal University. This seminar work has been carried out under my supervision and is to the satisfaction of the department.

Date:

Place:

Guide	**Head of Department**
Prof. Rahul Patel	Dr. Nikhil Gondaliya
Assistant Professor	Professor & Head

Acknowledgement

The completion of any project work depends upon cooperation, coordination, and combined efforts of several sources of knowledge. We would like to express our deepest thanks to Prof. Rahul Patel, for their valuable inputs, guidance, encouragement, wholehearted cooperation and constructive criticism throughout the duration of our project.

We hope that this project work report will provide all necessary information required to readers to fulfil their aspiration. Man's quest for knowledge never ends. Theory and practices are essential and complementary to each other. We would like to express our sincere thanks to Dr. Nikhil Gondaliya (Head of Department) for wholehearted support.

Abstract

Sign language is one of the oldest and most natural forms 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 on 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. Also we have a solution with neural networks for action recognition. In this model, the Primary step is to extract key-points of hand, face and pose passed through the classifier which predicts the action while providing 34% accuracy for 3 actions.

Table Of Content

CHAPTER – 1

Introduction to Domain

1.1 Introduction

American Sign Language (ASL) is a natural and visual language used by deaf and hard-of-hearing individuals in the United States and some parts of Canada. ASL is a complex language with its own grammar and syntax, and fingerspelling is an important component of ASL that involves spelling out words and names using hand shapes that represent individual letters of the alphabet.

Fingerspelling is often used in situations where there is no specific sign for a particular word, or when spelling out a word is more efficient than using a series of signs. However, fingerspelling can be difficult for those who are not familiar with ASL or who have limited exposure to sign language.

Neural networks are a type of machine learning that can be trained to recognize patterns and make predictions based on data input. In the case of fingerspelling, neural networks can be trained to recognize the hand shapes associated with each letter of the alphabet and provide real-time feedback to users.

Action recognition involves identifying and classifying different movements and gestures made by individuals using ASL. This can be challenging due to the variability in signing style and the complexity of the language. Neural networks can be used to extract key features from video footage of sign language users and make predictions about the intended action being performed.

Overall, the use of neural networks in sign language recognition has the potential to greatly enhance communication and accessibility for the deaf and hard-of-hearing communities. With continued research and development, these technologies can help bridge the gap between those who use sign language and those who do not.

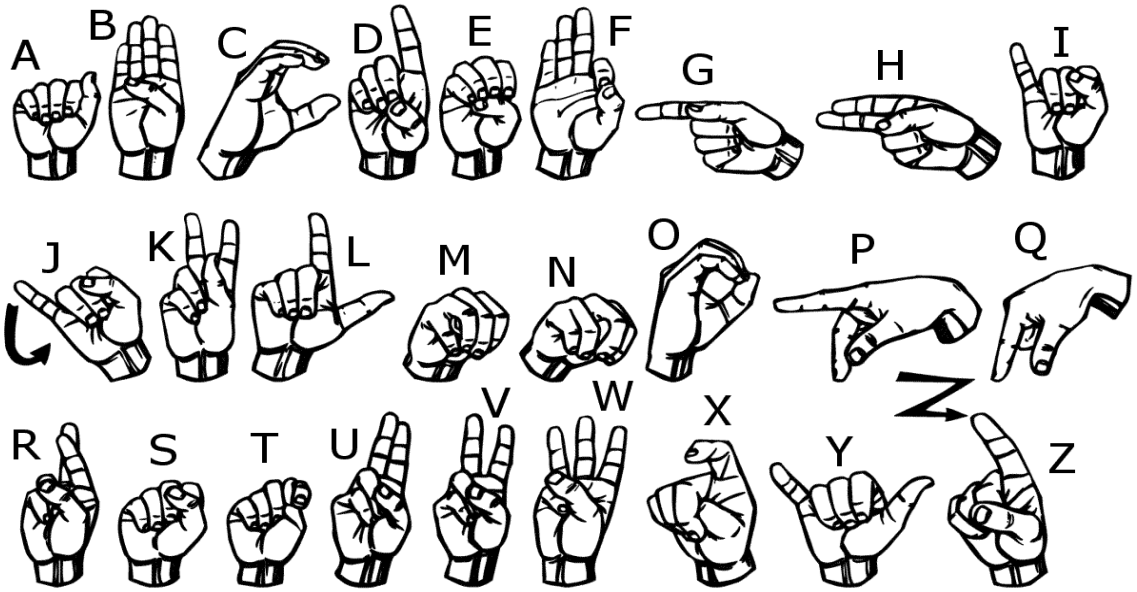


Figure 1: A-Z symbols in ASL

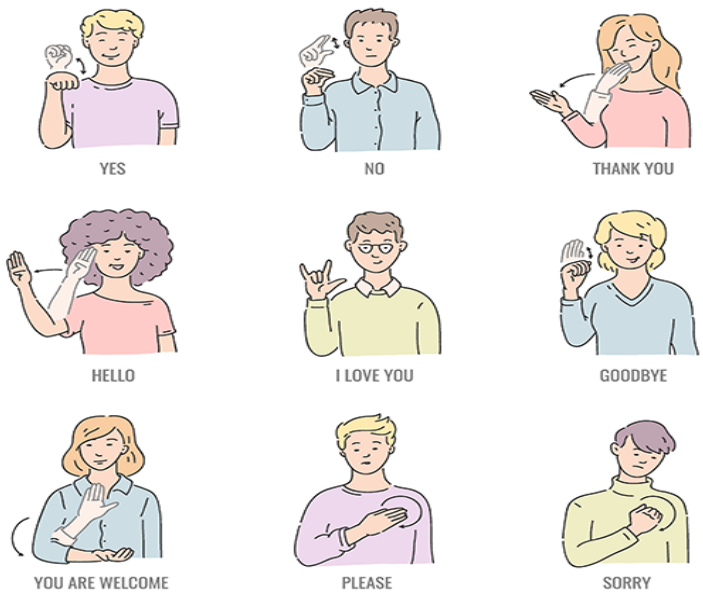


Figure 2: Example of some actions

1.2 Motivation

For interaction between normal people and deaf and dumb(D&M) people a language barrier is created as a 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 interface (HCI) where the computer understands the human sign language. There are various sign 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 world.

1.3 Literature survey

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

Data acquisition
Data preprocessing
Feature extraction
Gesture classification

1.3.1 Data acquisition

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

A. **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.
B. **Vision based approach:** In vision based methods computer camera is the input device for observing the information of hands 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. These systems tend to complement biological vision 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 of hand movements, to different skin-colour possibilities as well as to the variations in view points, scales, and speed of the camera capturing the scene.

1.3.2 Data preprocessing and Feature extraction:

In [1] the approach for hand detection combines threshold-based color detection with background subtraction.We can use Adaboost face detector to differentiate between faces and hands as both involve similar skin-color.

We can also extract the necessary image which is to be trained by applying a filter called Gaussian blur. The filter can be easily applied using open computer vision also known as OpenCV and is described in [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 preprocessing and can give us more concise and accurate data compared to applying filters on data received from video extraction.

We tried doing the hand segmentation of an image using color segmentation techniques but as mentioned in the research paper skin color and tone is highly dependent on the lighting conditions due to which output we got for the segmentation we tried to do were not so great. Moreover we have a huge number of symbols to be trained for our project many of which look similar to each other like the gesture for symbol 'V' and digit '2', hence we decided that in order to produce better accuracies for our 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.

1.3.3 Gesture classification

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-colour blobs corresponding to the hand into a body–face space centered on the face of the user. The goal is to recognize two classes of gestures: deictic and symbolic.The image is filtered using a fast look–up indexing table. After filtering, skin colour pixels are gathered into blobs. Blobs are statistical objects based on the location (x,y) and the colourimetry (Y,U,V) of the skin colour pixels in order to determine homogeneous areas.

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.Thus,unlike many other recognition methods, this method is not dependent on skin colour. The gestures are extracted from each frame of the video,with a static background. The first step is to segment and label the objects of interest and to extract geometric invariants from them. Next step is the classification of gestures by using a K nearest neighbor algorithm aided with distance weighting algorithm (KNNDW) to provide suitable data for a locally weighted Naïve Bayes classifier.

According to paper on “Human Hand Gesture Recognition Using a Convolution Neural Network” by Hsien-I Lin , Ming-Hsiang Hsu, and Wei-Kai Chen graduates of Institute of Automation Technology National Taipei University of Technology Taipei, Taiwan, they construct a skin model to extract the hand out of an image and then apply binary threshold to the whole image. After obtaining the threshold image they calibrate it about the principal axis in order to center the image about it. They input this image to a convolutional neural network model in order to train and predict the outputs. They have trained their model over 7 hand gestures and using their model they produce an accuracy of around 95% for those 7 gestures.

CHAPTER – 2
Detail Analysis of Domain

2.1 Feature Extraction and Representation:

The representation of an image as a 3D matrix having dimension as of height 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.

2.2 Artificial Neural Networks:

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

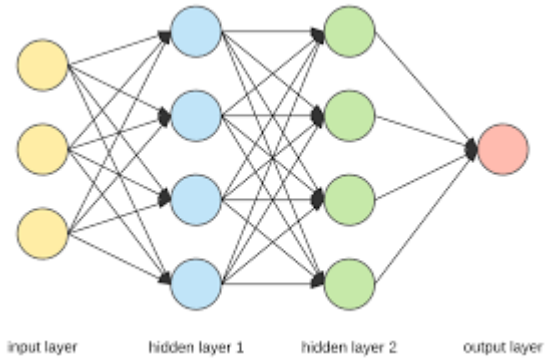


Figure 3: ANN Architecture

They are capable of learning and they have to be trained. There are different learning strategies :

1. Unsupervised Learning
2. Supervised Learning
3. Reinforcement Learning

2.3 Convolution Neural Network:

Unlike regular Neural Networks, in the layers of CNN, the neurons are 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.

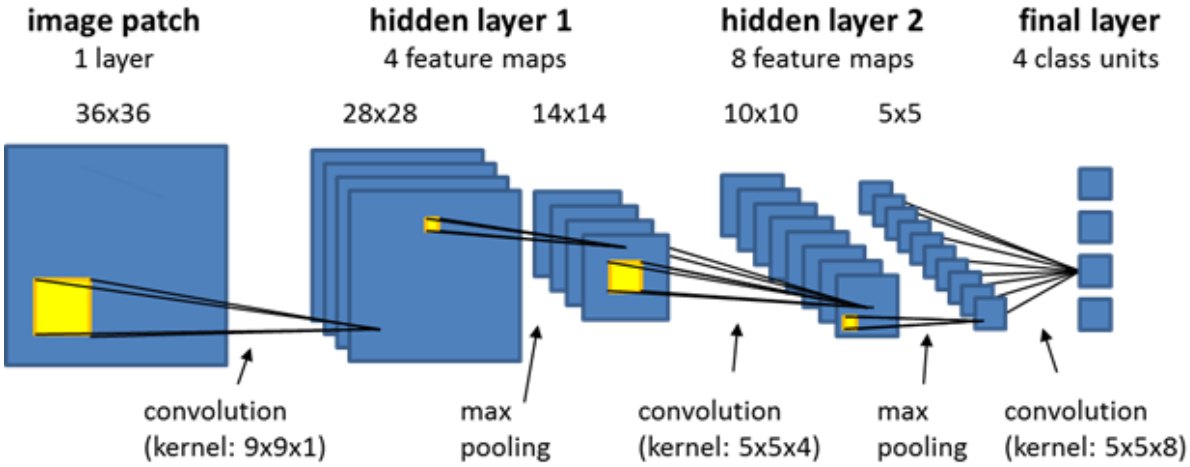


Figure 4: CNN Architecture

1) **Convolution Layer:** In convolution layer we take a small window size [typically of length 5*5] that extends to the depth of the input matrix. The layer consists of learnable filters of window size. During every iteration we slide the window by stride size [typically 1], and compute the dot product of filter entries and input values at a given position. As we continue this process we will create a 2-Dimensional activation matrix that gives the response of that matrix at every spatial position. That is, the network will learn filters that activate when they see some type of visual feature such as an edge of some orientation or a blotch of some color.
2) **Pooling Layer:** We use a pooling layer to decrease the size of the activation matrix and ultimately reduce the learnable parameters. There are two type 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 we'll finally get an activation matrix half of its original Size.

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

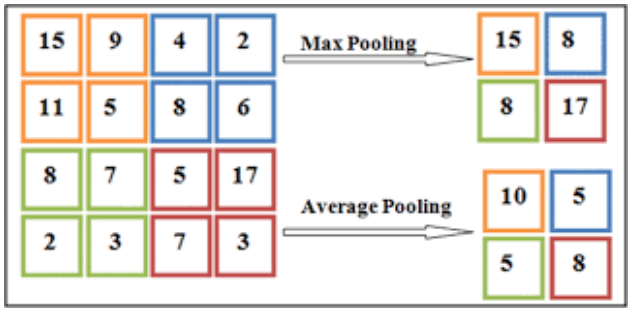


Figure 5: Max and Average Pooling

3) **Fully Connected Layer:** In convolution layer neurons are connected only to a local region, while in a fully connected region, we connect all the inputs to neurons.
4) **Final Output Layer:** After getting values from a fully connected layer, we will connect them to the 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.

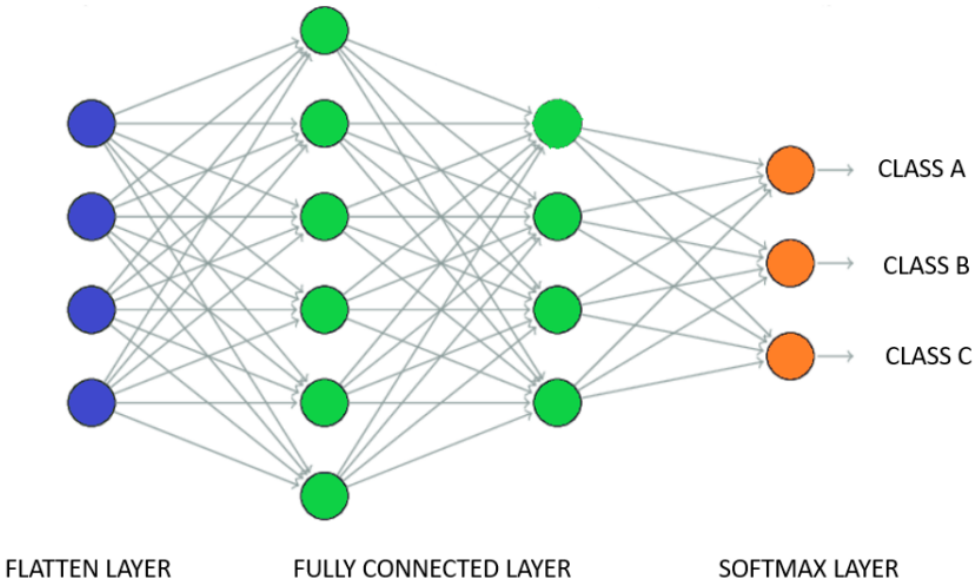


Figure 6: Classification

2.4 Long Short-Term Memory:

LSTMs are a type of RNN that have the ability to retain information over longer periods of time than traditional RNNs. This is achieved through the use of "memory cells" that allow information to be stored and retrieved over time.

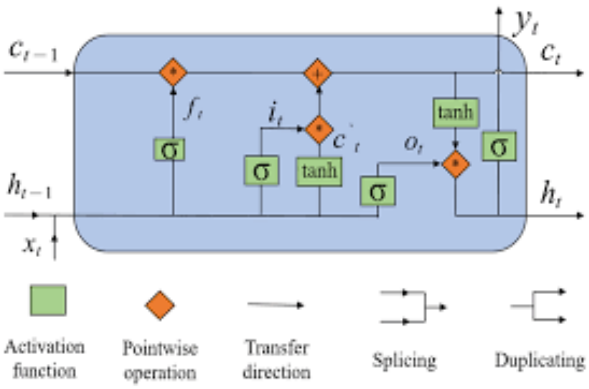


Figure 7: LSTM Architecture

Memory cells in LSTMs are controlled by gates, which are a set of neural network layers that determine which information is kept or discarded at each time step. The three types of gates used in LSTMs are the input gate, the forget gate, and the output gate.

The input gate controls how much of the new input should be stored in the memory cell. The forget gate determines how much of the previous memory cell state should be retained, and the output gate controls how much of the current memory cell state should be output.

LSTMs can be used for a variety of tasks, including natural language processing, speech recognition, and time series prediction. They have been shown to outperform traditional RNNs on many sequence processing tasks.

Training LSTMs involves minimizing a loss function that measures the difference between the predicted output and the true output. This is typically done using backpropagation through time, which involves calculating gradients and adjusting weights at each time step.

2.5 TensorFlow:

Tensorflow is an open source software library for numerical computation. First we define the nodes of the computation graph, then inside a session, the actual computation takes place. TensorFlow is widely used in Machine Learning.

2.6 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.

2.7 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, MATLAB/OCTAVE.

CHAPTER – 3
Project Implementation

3.1 Data Set Generation:

For the project we tried to find already made datasets but we couldn't find datasets 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 decided to create our own data set. Steps we followed to create our data set are as follows.

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

3.1.1 For Image Classification:

Firstly we captured around 800 images of each of the symbols in ASL 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 blue bounded square.

From this whole image we extract our ROI which is RGB and convert it into gray scale Image. Finally we apply our gaussian blur filter to our image which helps us extract various features of our image. The image after applying gaussian blur.

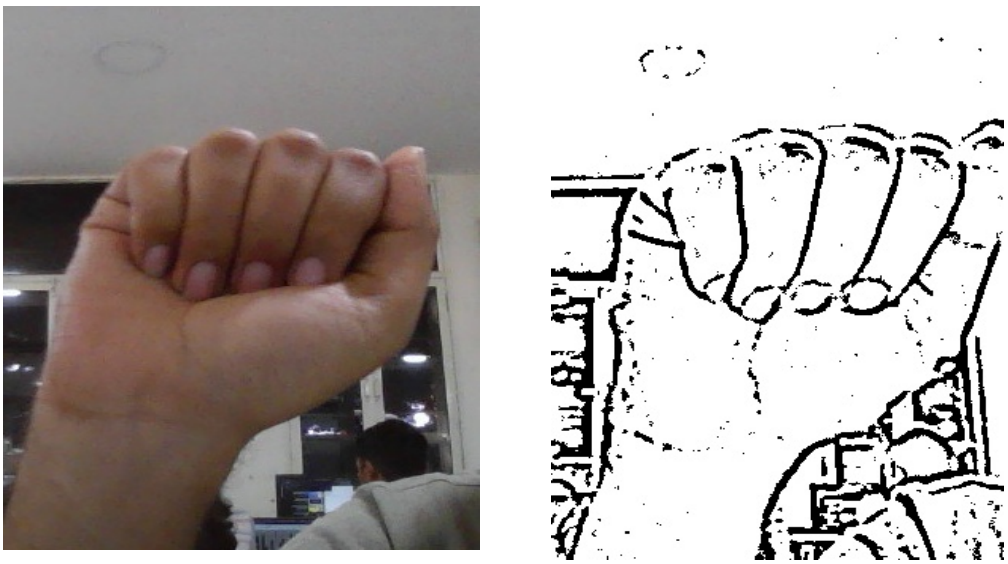


Figure 8: Image Dataset

3.1.2 For Video Classification:

Firstly we captured around 50 videos of each of the classes for training purposes and around 60 frames per video.

First we capture each video frame shown by the webcam of our machine. In which we make one folder for each video containing extracted features of hand,face and pose stored in the form of a numpy array.

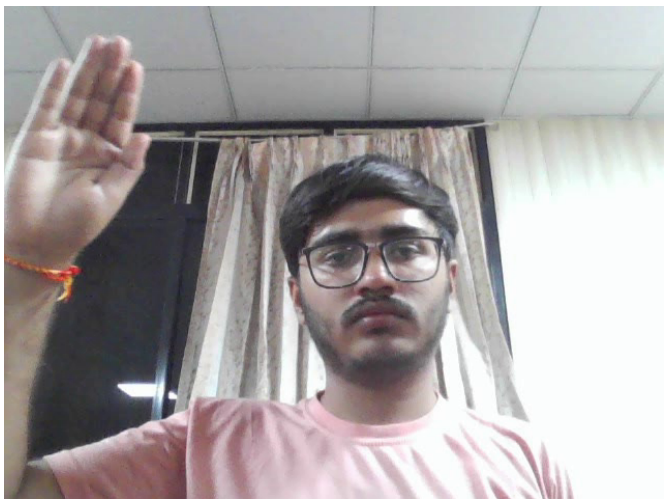


Figure 9: Video Dataset

3.2 Image classification:

3.2.1 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 array. Therefore, our picture is downsampled 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 downsampled again using max pool of 2x2 and is 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).

```
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 126, 126, 32)      320

 max_pooling2d (MaxPooling2D  (None, 63, 63, 32)       0
 )

 conv2d_1 (Conv2D)           (None, 61, 61, 32)        9248

 max_pooling2d_1 (MaxPooling  (None, 30, 30, 32)       0
 2D)

 flatten (Flatten)           (None, 28800)             0

 dense (Dense)               (None, 128)               3686528

 dropout (Dropout)           (None, 128)               0

 dense_1 (Dense)             (None, 96)                12384

 dropout_1 (Dropout)         (None, 96)                0

 dense_2 (Dense)             (None, 64)                6208
...
Total params: 3,716,378
Trainable params: 3,716,378
Non-trainable params: 0
```

Figure 10: CNN Model

3.2.2 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 and speeding up the training by reducing the computation time.

3.2.3 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 overfitting.

3.2.4 Dropout Layers:

The problem of overfitting, 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].

3.2.5 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).

3.3 Video classification:

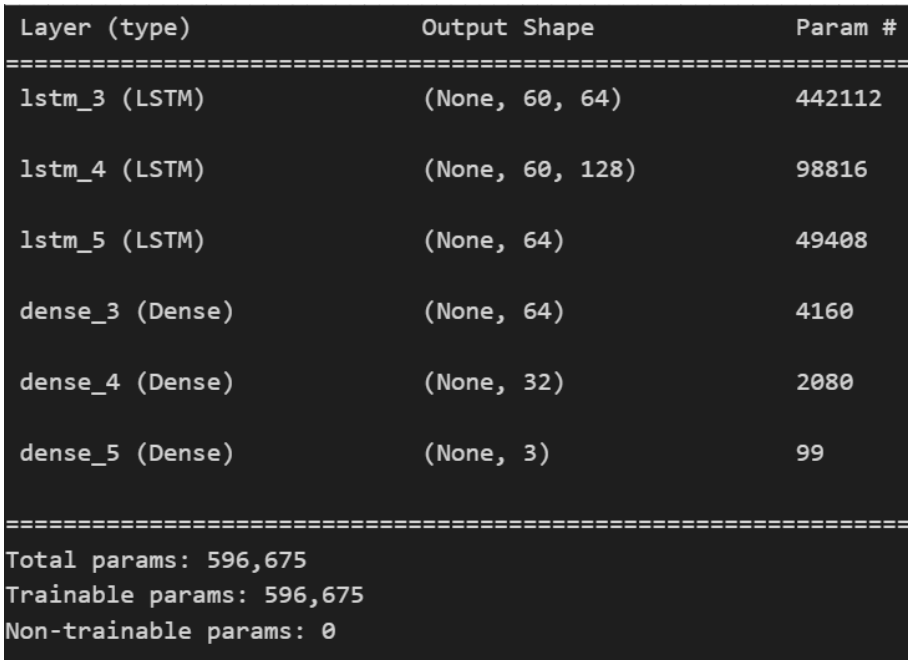
3.3.1 LSTM Model :

1)The model has three LSTM layers, which are specialized for processing sequential data. The output shape of the first two LSTM layers is (None, 60, 64) and (None, 60, 128), respectively, indicating that the model expects input sequences of length 60 and outputs sequences of the same length with 64 and 128 features, respectively. The output shape of the third LSTM layer is (None, 64), meaning that the model expects a single vector input and outputs a single vector with 64 features.

2)The model also includes three dense layers, which are fully connected layers that apply a linear transformation to the input data. The first dense layer has an output shape of (None, 64), meaning that it takes in the output of the third LSTM layer and produces a vector with 64 features. The second dense layer has an output shape of (None, 32), meaning that it takes in the output of the first dense layer and produces a vector with 32 features. The final dense layer has an output shape of (None, 3), indicating that it produces a vector of 3 values, which corresponds to the probabilities of each of the 3 possible actions.

3)The total number of parameters in the model is 596,675, which indicates the complexity of the model and the amount of information that it can store. The trainable parameters are those that the model learns during training, while the

non-trainable parameters are those that are fixed and cannot be updated during training.



Layer (type)	Output Shape	Param #
lstm_3 (LSTM)	(None, 60, 64)	442112
lstm_4 (LSTM)	(None, 60, 128)	98816
lstm_5 (LSTM)	(None, 64)	49408
dense_3 (Dense)	(None, 64)	4160
dense_4 (Dense)	(None, 32)	2080
dense_5 (Dense)	(None, 3)	99

Total params: 596,675
Trainable params: 596,675
Non-trainable params: 0

Figure 11: LSTM Model

3.3.2 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 and speeding up the training by reducing the computation time.

3.4 Training and Testing:

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

We feed the input images after preprocessing 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 and 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 which is positive at values which are not the same as 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.

As we have found out the cross entropy function, we have optimized it using Gradient Descent. In fact, the best gradient descent optimizer is called Adam Optimizer.

3.5 Challenges Faced:

There were many challenges faced by us during the project. The very first issue we faced was the dataset. Second issue was to select a filter 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. 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. On the other side we have no idea about video classification. Which we implement with help of youtuber and also facing problems with it's accuracy.

3.6 Limitations of our model:

The model works well only in good lighting conditions.
Plain background is needed for the model to detect with accuracy.
Videomodel take too much time to predict class.
Proper GUI is required.

3.7 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 95.0% and 34% on our respective image and video datasets. We are able to improve our prediction after implementing two layers of algorithms in which we verify and predict symbols which are more similar to each other. This way we are able to detect almost all the symbols provided that they are shown properly, there is no noise in the background and lighting is adequate.

3.8 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 class 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_b

ilateral_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]https://opencv.org/

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

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