MINOR PROJECT REPORT

TOPIC : INTEGRATED SUPPORT SYSTEM FOR THE DEAF AND DUMB

MADE BY:

1.Devanshi Kapla 20103176 B6

2. Tamanna Madan 2013196 B7

3. Shreya Khosla 20103203 B7

SUBMITTED TO:

Dr. Bharat Gupta

Dr. Chetna Dabas

Under the supervision of:
Dr. Niyati Aggarwal

DECLARATION

I/We hereby declare that this submission is my/our own work and that, to the best of my knowledge and belief, it contains no material previously published or written by another person nor material which has been accepted for the award of any other degree or diploma of the university or other institute of higher learning, except where due acknowledgment has been made in the text.

Place: Noida Signature:

Date: 03/12/2022 Name: Shreya Khosla 20103203
Tamanna Madan 20103196
Devanshi Kapla 20103176

Signature of the Guide

Place:

Date:

CERTIFICATE

This is to certify that the work titled **"Integrated Support Software For Deaf And Dumb"** submitted by **Shreya Khosla Tammana Madan and Devanshi Kapla** in partial fulfillment for the award of degree of B.Tech of Jaypee Institute of Information Technology, Noida has been carried out under my supervision. This work has not been submitted partially or wholly to any other University or Institute for the award of this or any other degree or diploma.

Signature of Supervisor	……..
Name of Supervisor	Dr. Niyati Aggarwal
Designation	Assistant Professor (Senior Grade)
Date	03/12/2022

ACKNOWLEDGEMENT

I would like to thank the Computer Science Department for giving me the opportunity to work on this project. I would like to express my special thanks to our mentor **Dr. Niyati Aggarwal** for his/her time and efforts he/she provided throughout the year. Your useful advice and suggestions were really helpful to me during the project's completion.

This endeavor would not have been possible without her help and supervision. We could not have asked for a finer mentor in our studies.

This initiative would not have been a success without the contributions of each and every individual in our group . We were always there to cheer each other on, and that is what kept us together until the end.

This was quite a great experience and I learned a lot from it. It helped me to explore my skills and increased my interest in this project.

SUMMARY

Sign language is one of the oldest and most natural forms of communication, but most people do not know it and finding interpreters is very difficult, so we developed a real-time method using neural networks for finger spelling.

A language barrier is created in the interaction between normal people and D&M people as a sign language structure that differs from normal text.

As such, they rely on vision-based communication for interaction. Having a common sign language-to-text interface makes it easier for others to understand your gestures. Therefore, a visual-based interface system was researched that would allow people at D&M to enjoy communicating without really knowing each other's languages.

The goal is to develop an easy-to-use human-computer interface (HCI) that allows computers to understand human sign language. There are many different sign languages around the world, or American Sign Language (ASL). French Sign Language, British Sign Language (BSL), Indian Sign Language, Japanese Sign Language, etc. are active all over the world. [8]

LIST OF ABBREVIATIONS

CNN: Convolutional Neural Network

ASL: American Sign Language

HCI: Human Computer Interface

HMM: Hidden Markov Model

D&M: Deaf and Mute

Relu: Rectified linear network

ANN : Artificial neural network

KNNDW: K nearest neighbor and distance weighted algorithm

ROI : Region of interest

TABLE OF CONTENTS

LIST OF FIGURES

ABSTRACT

Sign language is one of the oldest and most natural forms of language for communication. The research of sign language recognition has been a topic of interest in recent years in the fields of computing vision, artificial intelligence and human computer interaction. 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 84.4 % accuracy for the 26 letters of the alphabet.

While this percentage can be increased with a variety of techniques, the test shows that users can create the dataset using this software and that those values can be used in a classification algorithm with acceptable results.

Sign language recognition has at least three steps such as data acquisition, data classification and results. This paper introduces a software capable of facilitating the data acquisition step with the Logitech C270 HD webcam. [8]

CHAPTER 1

INTRODUCTION

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

Sign language is a visual language and consists of 3 major components

Fingerspelling	Word level sign vocabulary	Non-manual features
Used to spell words letter by letter .	Used for the majority of communication.	Facial expressions and tongue, mouth and body position.

Figure 1.1 Major Components of Sign Language

Minimizing the verbal exchange gap among D&M and non-D&M people turns into a want to make certain effective conversation among all. Sign language translation is among one of the most growing lines of research and it enables the maximum natural manner of communication for those with hearing impairments. A hand gesture recognition system offers an opportunity for deaf people to talk with vocal humans without the need of an interpreter. The system is built for the automated conversion of ASL into textual content and speech.

In our project we primarily focus on producing a model which can recognize Fingerspelling based hand gestures in order to form a complete word by combining each gesture. Gestures are recognized using following image:

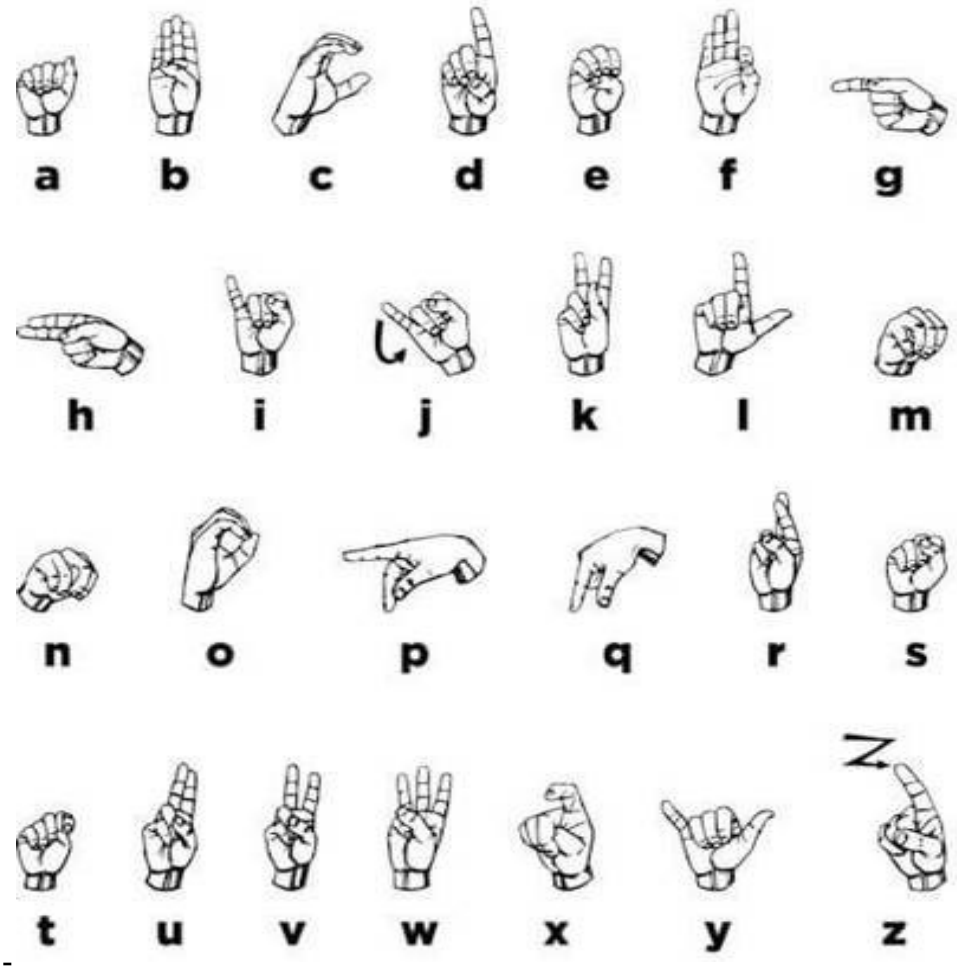

Figure 1.2 American Sign Language Symbols

1.1 PROBLEM STATEMENT

Making communication easier for deaf and dumb by converting sign language to text.

1.2 DETAILS OF PROBLEM STATEMENT

We trying to make the interview process more inclusive by making an software application that will help deaf and dumb people.

Our software can detect sign language and convert it into text .We will be using AZURE API to test the model, along with that we will be using several layers of other training models using Machine Learning in order to increase the accuracy.

This will help communication easier between a person who can communicate using sign language and the person who cannot understand the sign language. This will motivate the people to not only get affected by the disabilities but it will help them to focus on their abilities.

CHAPTER 2

MOTIVATION

For interaction between normal people and Deaf & Mute people a language barrier is created as sign language structure since it 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, then the gestures can be easily understood by non-Deaf & Dumb people. So, research has been made for a vision-based interface system where Deaf & Dumb 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.

We have made this project on a real life scenario where we have seen someone very near and dear to us face this language barrier issue. This barrier became a hurdle in the path of the said person’s path to growth, he was not allowed to sit in the interviews mainly because there was no provision for this particular section of people.These people are indeed very talented and also are a deserving candidates for several positions in different companies but this language barrier stops them.

This project will not only help them in the interview process but also in their regular routines, it will help them to express themselves freely without any need of hiring a sign language [8] translator to interpret what they are saying.

CHAPTER 3

LITERATURE SURVEY

In recent years there has been tremendous research done on hand gesture recognition.

With the help of literature survey, we realized that the basic steps in hand gesture recognition are:

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

3.1 DATA ACQUISITION

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

1. **USE OF SENSORY DEVICES**

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

2. **VISION BASED APPROACH**

   In vision-based methods, the computer webcam is the input device for observing the information of hands and/or fingers. The Vision Based methods require only a camera, thus realizing a natural interaction between humans and computers without the use of any extra devices, thereby reducing cost. These systems tend to complement biological vision by describing artificial vision systems that are implemented in software and/or hardware. The main challenge of vision-based hand detection ranges from coping with the large

variability of the human hand's appearance due to a huge number of hand movements, to different skin-color possibilities as well as to the variations in viewpoints, scales, and speed of the camera capturing the scene.

3.2 **DATA PRE-PROCESSING FEATURE EXTRACTION FOR VISION-BASED APPROACH**

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

- We can also extract the necessary image which is to be trained by applying a filter called Gaussian Blur (also known as Gaussian smoothing). The filter can be easily applied using open computer vision (also known as OpenCV) and is described in [3].

- For extracting the necessary image which is to be trained we can use instrumented gloves as mentioned in [4]. This helps reduce computation time for Pre-Processing and gives us more concise and accurate data compared to applying filters on data received from video extraction.

- We tried doing the hand segmentation of an image using color segmentation techniques but skin color and tone is highly dependent on the lighting conditions due to which output we got for the segmentation we tried to do was 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.

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-color 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 color pixels are gathered into blobs. Blobs are statistical objects based on the location (x, y) and the colorimetry (Y, U, V) of the skin color 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 color. 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 the 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 have constructed a skin model to extract the hands 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 the axis. 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 this model they produced an accuracy of around 95% for those 7 gestures. [1]

CHAPTER 4

METHODOLOGY

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

4.1 **DATA ACQUISITION**

As mentioned in the introduction the data acquisition is one step of the sign language recognition, this step consists of digitize the information from the user to make it usable to classification algorithms, the information from the user can be acquired in different ways depending on the tools used, such as visual based like cameras or wearable like the use of electromyogram, sensors in a glove, etc. Those tools have different advantages such as low-cost, hardware and software compatibility, user detection, etc., in the same way those tools have disadvantages such as occlusion, cost, trouble with the users, etc.

4.2 **DATA PRE-PROCESSING**

Significance of text preprocessing in the performance of models. Data preprocessing is an essential step in building a Machine Learning model and depending on how well the data has been preprocessed; the results are seen.In NLP, text preprocessing is the first step in the process of building a model.The various text preprocessing steps are:

1. Tokenization
2. Lower casing
3. Stop words removal
4. Stemming
5. Lemmatization

These various text preprocessing steps are widely used for dimensionality reduction.In the vector space model, each word/term is an axis/dimension. The text/document is represented as a vector in the multi-dimensional space.The number of unique words means the number of dimensions

4.3 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) [12][4] library in order to produce our dataset.
- Firstly, we captured around 800 images of each of the symbols in ASL (American Sign Language) for training purposes and around 200 images per symbol for testing purposes.
- Then, 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 as shown in the image below:

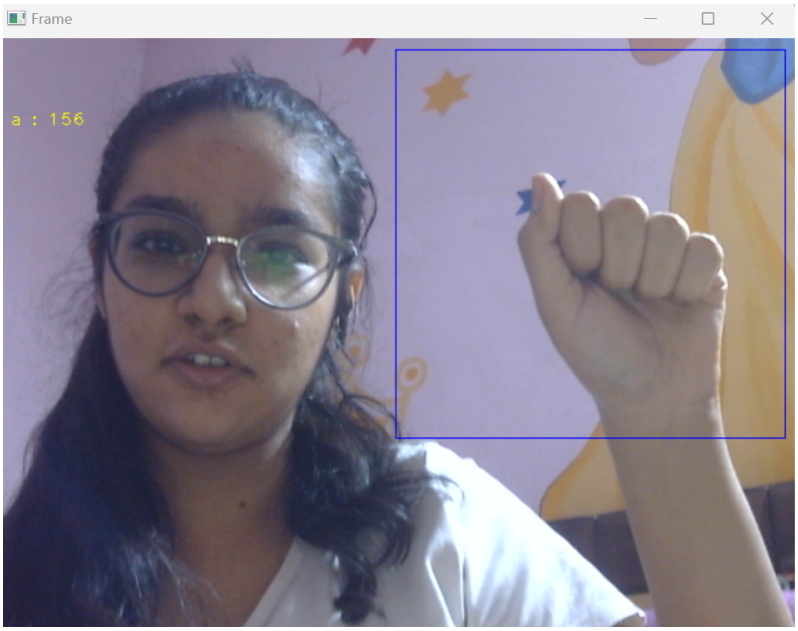

Figure 4.1 Collecting Data

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

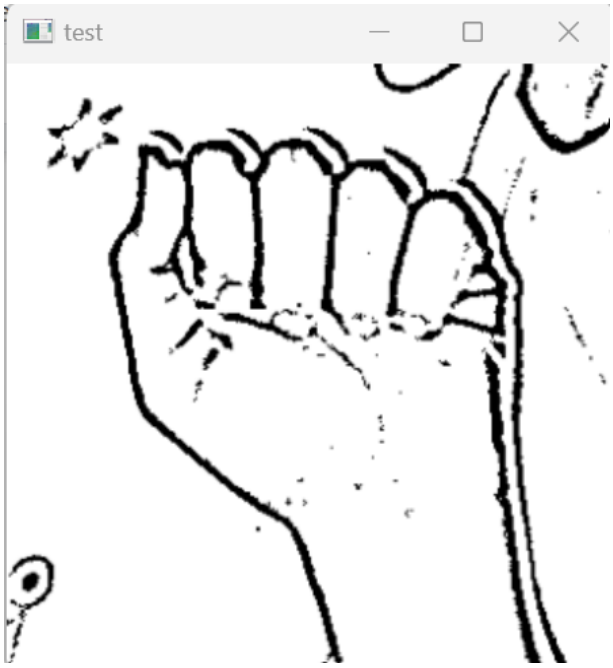

Figure 4.2 Gaussian Blur

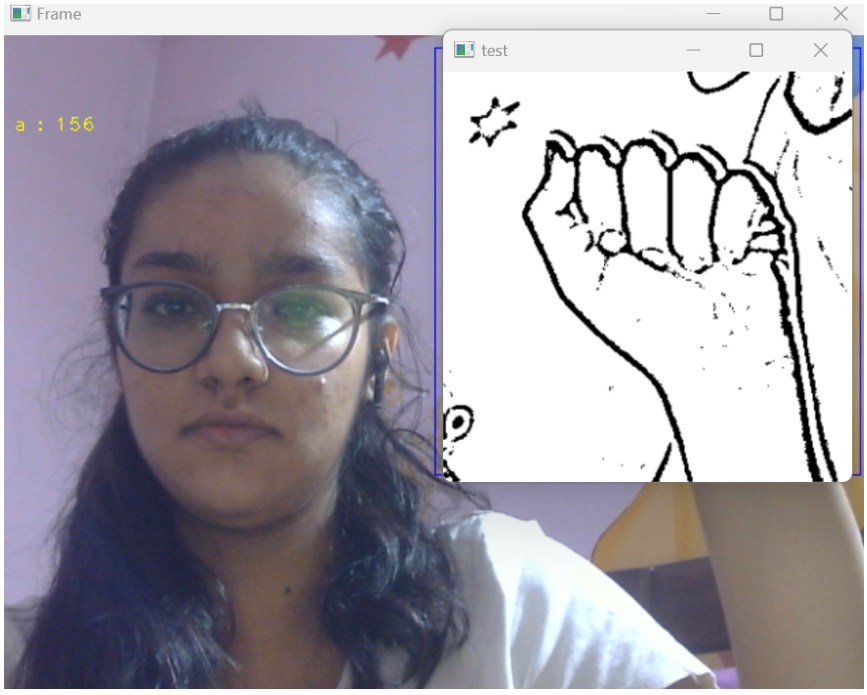

Figure 4.3 Collecting Training and Testing Data

4.4 GESTURE CLASSIFICATION

Real-time recognition of dynamic hand gestures from video streams is a challenging task since:

1. there is no indication when a gesture starts and ends in the video
2. performed gestures should only be recognized once
3. the entire architecture should be designed considering the memory and power budget.

In this work, we address these challenges by proposing a hierarchical structure enabling offline-working convolutional neural network (CNN) architectures to operate online efficiently by using a sliding window approach. [14][6][7]

Our approach uses two layers of algorithms to predict the final symbol of the user:

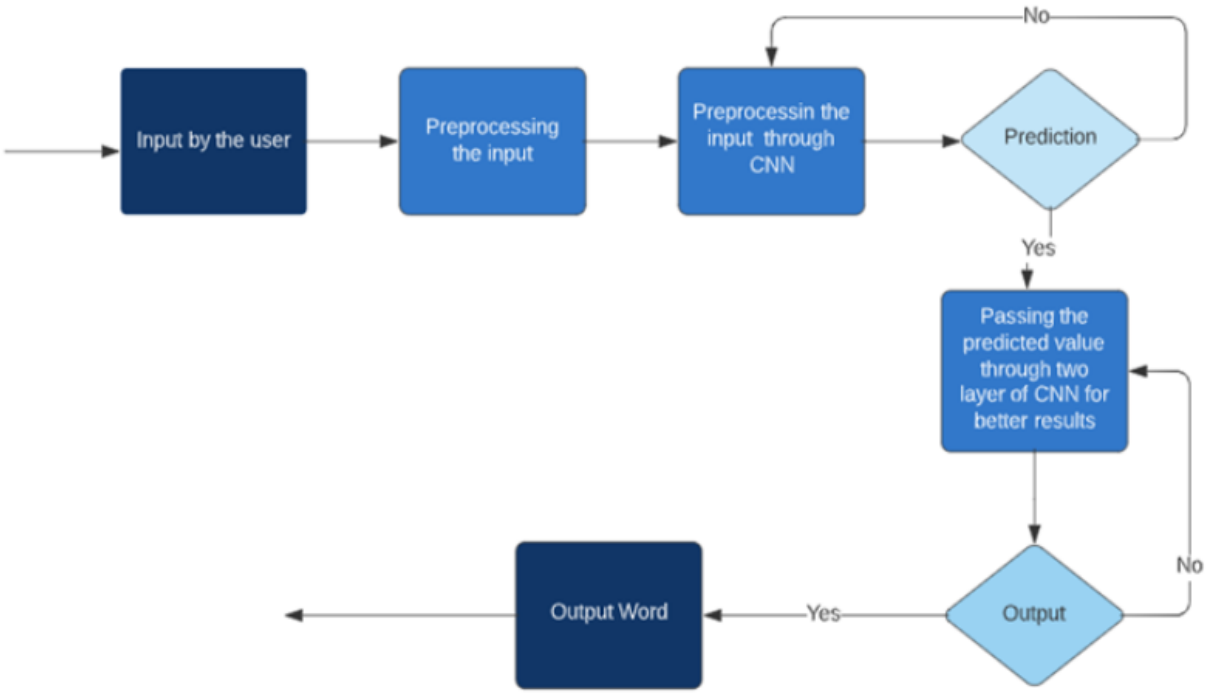

Figure 4.4 Working Model OF Project

Algorithm Layer 1:

1. Apply Gaussian Blur filter and threshold to the frame taken with openCV to get the processed image after feature extraction.
2. This processed image is passed to the CNN model for prediction and if a letter is detected for more than 50 frames then the letter is printed and taken into consideration for forming the word.
3. Space between the words is considered 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 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 down sampled using max pooling of 2x2 i.e we keep the highest value in the 2x2 square of array. Therefore, our picture is down sampled to 63x63 pixels.

3. **2nd Convolution Layer**

Now, these 63 x 63 from the output of the first pooling layer serve as an input to the second convolutional layer. It is processed in the second convolutional layer using 32 filter weights (3x3 pixels each). This will result in a 60 x 60 pixel image.

4. **2nd Pooling Layer**

The resulting images are down sampled again using a max pool of 2x2 and 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 these layers 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).

- **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.

- **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.

- **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].

- **Optimizer:**

We have used Adam optimizer for updating the model in response to the output of the loss function.

Adam optimizer 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 using two layers of algorithms to verify and predict symbols which are more similar to each other so that we can get as close as we can get to detect the symbol shown. In our testing we found that following symbols were 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 cases we made three different classifiers for classifying these sets:

1. **{D, R, U}**
2. **{T, K, D, I}**
3. **{S, M, N} [14][7]**

4.5 FINGER SPELLING SENTENCE FORMATION IMPLEMENTATION

1. Whenever the count of a letter detected exceeds a specific value and no other letter is close to it by a threshold, we print the letter and add it to the current string (In our code we kept the value as 50 and difference threshold as 20).
2. Otherwise, we clear the current dictionary which has the count of detections of the present symbol to avoid the probability of a wrong letter getting predicted.
3. Whenever the count of a blank (plain background) detected exceeds a specific value and if the current buffer is empty no spaces are detected.
4. In other cases it predicts the end of a word by printing a space and the current gets appended to the sentence below.

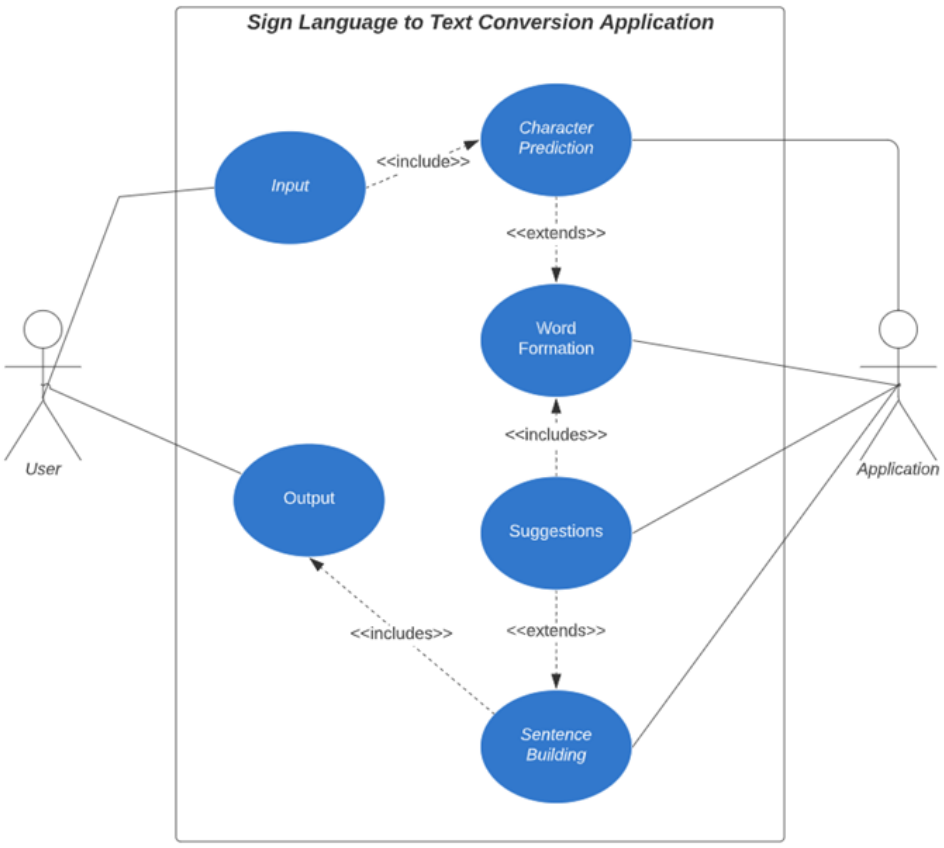

Figure 4.5 Conversion

4.6 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 assists in predicting complex words.[15]

4.7 TRAINING AND TESTING

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

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

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

4.8 FEATURE EXTRACTION

Feature extraction is part of the data reduction process and is followed by feature analysis. One of the important aspects of feature analysis is determining exactly which features are important .

Feature extraction is a complex problem in which the whole image or the transformed image is often taken as the input. The goal of feature extraction is to nd the most discriminating information in the recorded images. Feature extraction operates on two-dimensional image arrays but produces a list of descriptions or a feature vector.

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.

4.9 ARTIFICIAL NEURAL NETWORK (ANN)

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.

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

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

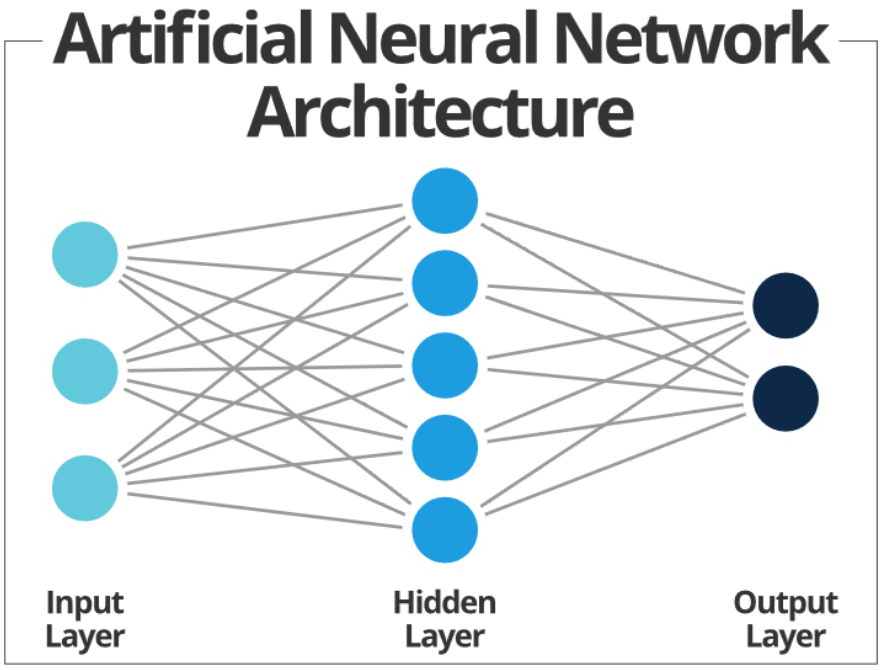

Figure 4.6 Artificial Neural Network

4.10 CONVOLUTION NEURAL NETWORK (CNN)

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.

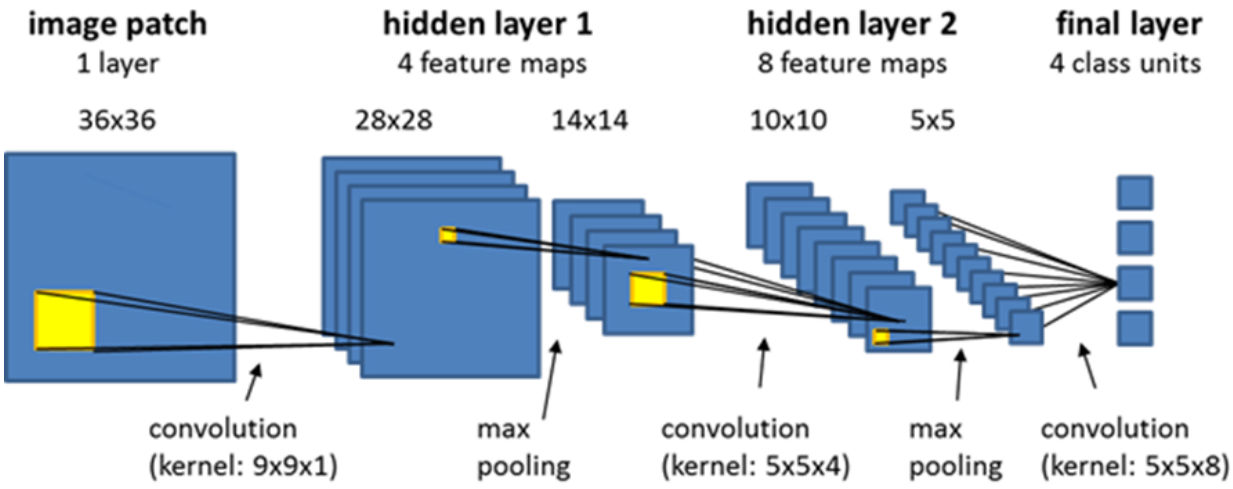

Figure 4.7 CNN

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 types of pooling:

a. <u>Max Pooling</u>

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. <u>Average Pooling</u>

In average pooling, we take advantage of all Values in a window.

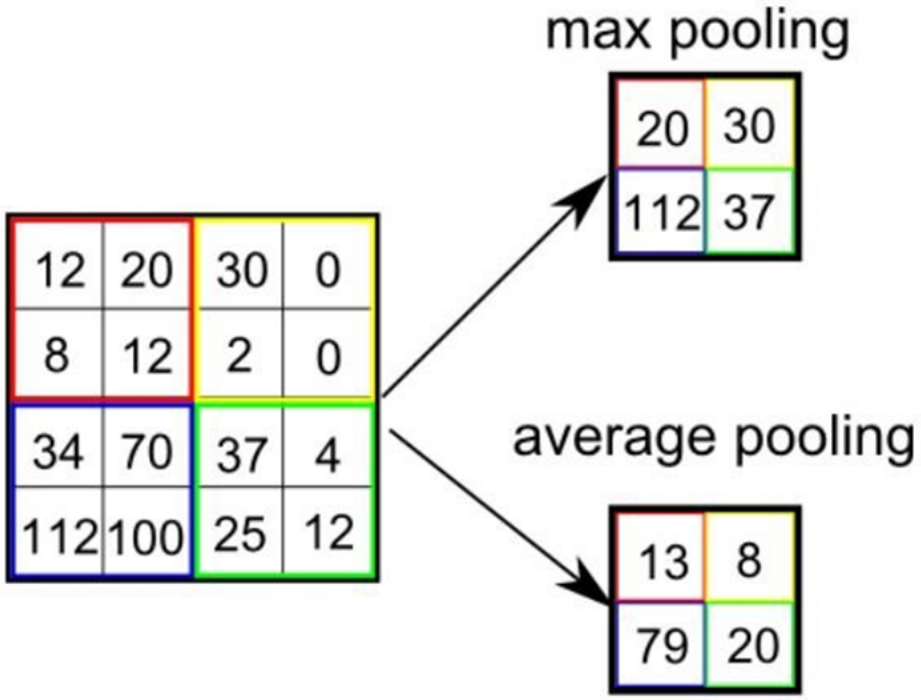

Figure 4.8 Pooling

3. FULLY CONNECTED LAYER

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

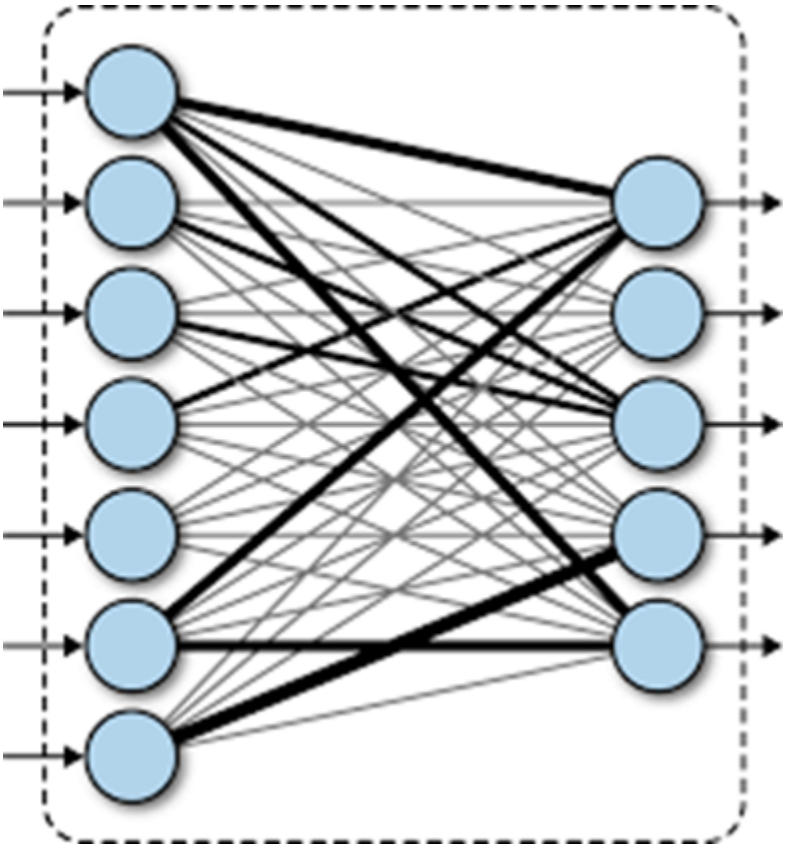

Figure 4.9 Fully Connected

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. [14][7]

CHAPTER 5
IMPLEMENTATION

5.1 TECHNOLOGY USED

5.1.1 TENSORFLOW

TensorFlow is an end-to-end open-source platform for Machine Learning. It has a comprehensive, flexible ecosystem of tools, libraries and community resources that lets researchers push the state-of-the-art in Machine Learning and developers easily build and deploy Machine Learning powered applications.

TensorFlow offers multiple levels of abstraction so you can choose the right one for your needs. Build and train models by using the high-level Keras API, which makes getting started with TensorFlow and machine learning easy.

If you need more flexibility, eager execution allows for immediate iteration and intuitive debugging. For large ML training tasks, use the Distribution Strategy API for distributed training on different hardware configurations without changing the model definition. [13]

5.1.2 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. [9][10]

5.1.3 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. [12][4]

5.1.4 MICROSOFT AZURE

Microsoft Azure, often referred to as Azure is a cloud computing platform operated by Microsoft for application management via Microsoft-managed data centers. Microsoft Azure has multiple capabilities such as software as a service (SaaS), platform as a service (PaaS) and infrastructure as a service (IaaS) and supports many different programming languages, tools, and frameworks, including both Microsoft-specific and third-party software and systems.

5.1.5 MACHINE LEARNING

Machine learning (ML) is a field of inquiry devoted to understanding and building methods that 'learn', that is, methods that leverage data to improve performance on some set of tasks. Machine learning algorithms build a model based on sample data, known as training data, in order to make predictions or decisions without being explicitly programmed to do so. Machine learning algorithms are used in a wide variety of applications, such as in medicine, email filtering, speech recognition, agriculture, and computer vision, where it is difficult or unfeasible to develop conventional algorithms to perform the needed tasks.

5.2 PROGRAMMING LANGUAGE USED

PYTHON

We have used python because Benefits that make Python the best fit for machine learning and AI-based projects include simplicity and consistency, access to great libraries and frameworks for AI and machine learning (ML), flexibility, platform independence, and a wide community. These add to the overall popularity of the language.

Python is a high-level, general-purpose programming language. Its design philosophy emphasizes code readability with the use of significant indentation.

Python is dynamically-typed and garbage-collected. It supports multiple programming paradigms, including structured (particularly procedural), object-oriented and functional programming. It is often described as a "batteries included" language due to its comprehensive standard library.

5.3 DATA PRE-PROCESSING

The initial set of data set was downloaded from kaggle and then trained of the microsoft azure platform

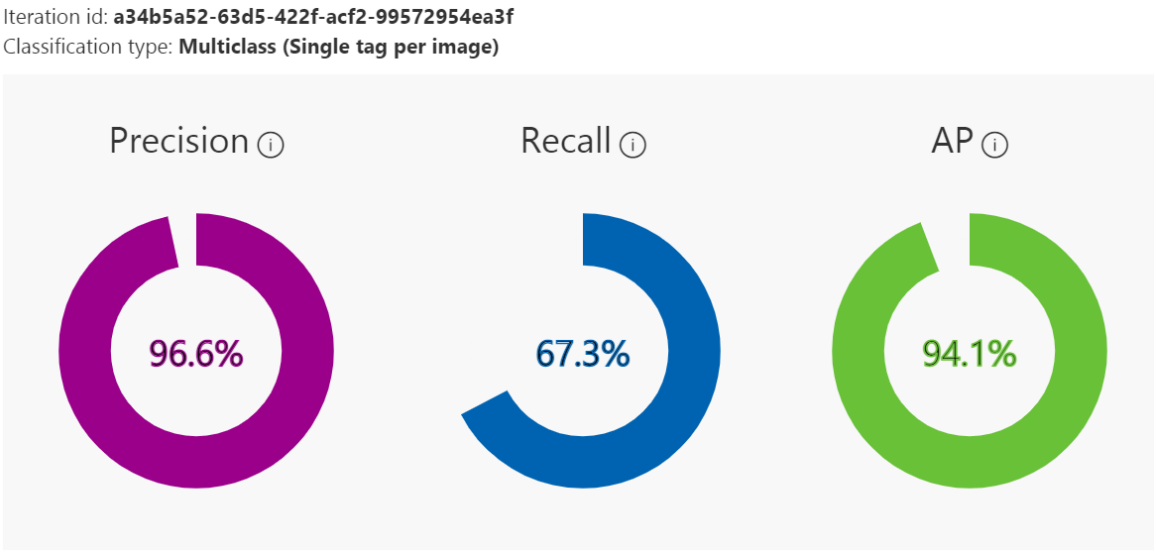

Figure 5.1 Training of data on Microsoft Azure

Significance of text preprocessing in the performance of models. Data preprocessing is an essential step in building a Machine Learning model and depending on how well the data has been preprocessed; the results are seen.In NLP, text preprocessing is the first step in the process of building a model.

5.4 GENERATION OF DATA

But yet for the project the dataset did not have enough data to suit 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.

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

5.5 CNN

Real-time recognition of dynamic hand gestures from video streams is a challenging task since:

there is no indication when a gesture starts and ends in the video

performed gestures should only be recognized once

the entire architecture should be designed considering the memory and power budget.

In this work, we address these challenges by proposing a hierarchical structure enabling offline-working convolutional neural network (CNN) architectures to operate online efficiently by using a sliding window approach. Dense Layer is a simple layer of neurons in which each neuron receives input from all the neurons of the previous layer, thus called dense. Dense Layer is used to classify images based on output from convolutional layers.

5.6 RELU (Rectified linear network)

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 stops the exponential growth of the function.

5.7 SAMPLE OUTPUT

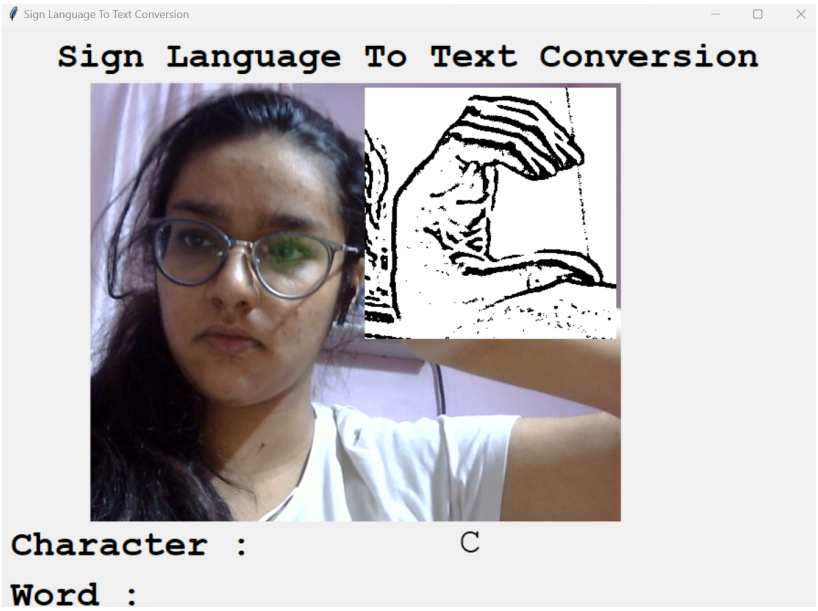

Figure 5.2 Identifying C

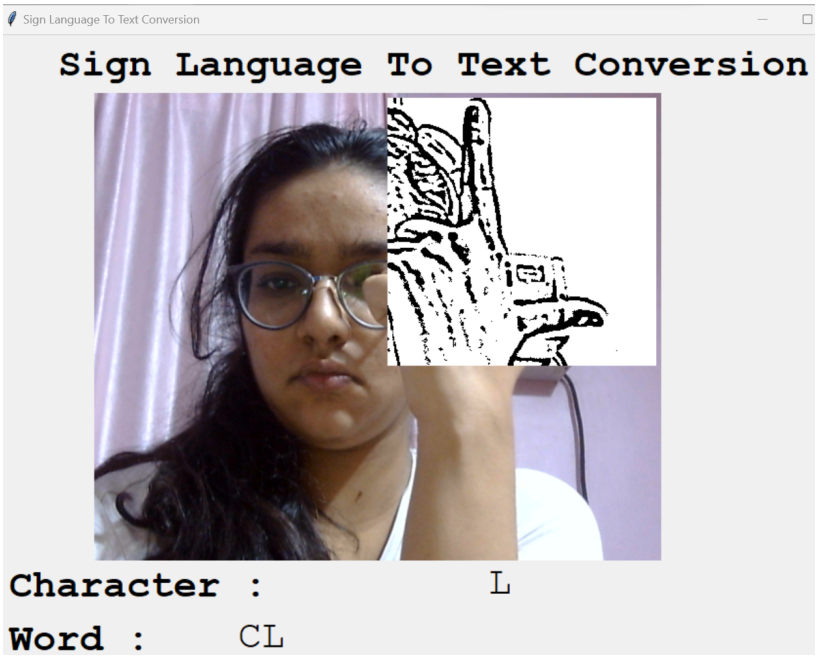

Figure 5.3 Identify L and creating a word with L and previously detected C

CHAPTER 6

CHALLENGES FACED

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

We couldn't find any existing data set as per our requirements and hence we decided to make our own data set. 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.

We tried various cloud and its features and tried AWS, GCP, IBM and finally shifted to Azure .

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

CHAPTER 7

RESULT

We have achieved an accuracy of around 80.2**%** 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 83.2%, 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 a hidden Markov model classifier and a vocabulary of 30 words and they achieve an error rate of **15.6%**.

In [9] they achieve an average accuracy of **84%** for 41 static gestures in Japanese sign language.

Using depth sensors map [10] achieved an accuracy of **84.4%** for observed signers and **83.58%** and **85.49%** for new signers.

They also used CNN for their recognition system. One thing should be noted that our model doesn't use any background subtraction algorithm while 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 the audience to buy and our model uses a normal webcam of the laptop hence it is a great plus point.

CHAPTER 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 Pre Processing to predict gestures in low light conditions with a higher accuracy.

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

CHAPTER 9

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 **84.4%** on our data set(having 12,000 elements) . We have improved our prediction after implementing two layers of algorithms wherein we have verified and predicted symbols which are more similar to each other.

This gives us the ability to detect almost all the symbols provided that they are shown properly, there is no noise in the background and lighting is adequate.

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 Muller, Mary Ellen Foster, and Alois Knoll “A Na¨ive Bayes Munich, Dept. of Informatics VI, Robotics and Embedded Systems, Boltzmannstr. 3, DE-85748 Garching, Germany.

[3]https://docs.opencv.org/2.4/doc/tutorials/imgproc/gausian_median_blur_bilateral_filter/gausian_median_blur_bilateral_filter.html

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

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

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

[7] Pigou L., Dieleman S., Kindermans PJ., Schrauwen B. (2015) Sign Language Recognition Using Convolutional Neural Networks. In: Agapito L., Bronstein M., Rother C. (eds) Computer Vision - ECCV 2014 Workshops. ECCV 2014. Lecture Notes in Computer Science, vol 8925. Springer, Cham

[8] Zaki, M.M., Shaheen, S.I.: Sign language recognition using a combination of new vision-based features. Pattern Recognition Letters 32(4), 572–577 (2011).

[9] https://keras.io/

[10] https://github.com/keras-team/keras

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

[12] https://opencv.org/

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

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

[15] http://hunspell.github.io/