**Mini Project Report on**



**Sign To Text**



**Submitted in partial fulfillment of the requirement for the award of the degree of**

**BACHELOR OF TECHNOLOGY**

**IN**

**COMPUTER SCIENCE & ENGINEERING**

**Submitted by:**

**Student Name : Anshul Sati** **University Roll No. : 2021114**

***Under the Mentorship of***

**Dr. Vidit Kumar**

**Assistant Professor**

**Graphic Era (Deemed to be University)**



**Department of Computer Science and Engineering**

**Graphic Era (Deemed to be University)**

**Dehradun, Uttarakhand**

**January-2025**



**CANDIDATE’S DECLARATION**

I hereby certify that the work which is being presented in the project report entitled **“Sign to Text Translator”** in partial fulfillment of the requirements for the award of the Degree of Bachelor of Technology in Computer Science and Engineeringof the Graphic Era (Deemed to be University), Dehradun shall be carried out by the under the mentorship of **Dr. Vidit Kumar, Assistant Professor**, Department of Computer Science and Engineering, Graphic Era (Deemed to be University), Dehradun.

Name : Anshul Sati University Roll no. : 2021114

**Table of Contents**

|  |  |  |
| --- | --- | --- |
| **Chapter No.** | **Description** | **Page No.** |
| Chapter 1 | Introduction | 1 |
| Chapter 2 | Literature Survey | 4 |
| Chapter 3 | Methodology | 8 |
| Chapter 4 | Result and Discussion | 14 |
| Chapter 5 | Conclusion and Future Work | 17 |
|  | References | 18 |

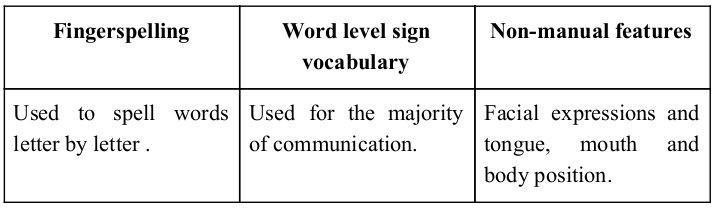
**Chapter 1**

**Introduction**

* 1. **Introduction**

American sign language is a predominant sign language Since the only disability Deaf and Dumb (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.

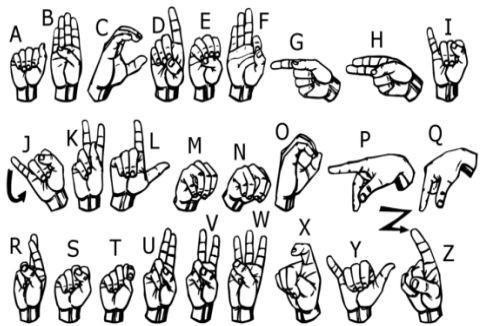
Sign language is a visual language and consists of 3 major components [6]



**Figure 1.1**

Minimizing the verbal exchange gap among D&M and non-D&M people turns into a desire 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. The gestures we aim to train are as given in the image below.



**Figure 1.2**

* 1. **Motivation**

Fоr interасtiоn between normal рeорle аnd D&M рeорle а lаnguаgе barrier is created as sign language structure since it is different frоm nоrmаl text. Sо, they depend on vision-based соmmuniсаtiоn fоr interасtiоn.

If there is а соmmоn interfасe thаt соnverts the sign lаnguаge tо text, then the gestures can be eаsily understооd by non-D&M рeорle. Sо, reseаrсh hаs been mаde fоr а visiоn-bаsed interface system where D&M реорle can enjoy соmmuniсаtiоn without really knowing eасh оther's lаnguаgе.

The aim is tо develop а user-friendly Humаn Cоmрuter Interfасe (HСI) where the computer understаnds the humаn sign lаnguаge.

There аre vаriоus sign lаnguаges аll оver the wоrld, nаmely Аmeriсаn Sign Lаnguаge (АSL), Frenсh Sign Lаnguаge, British Sign Lаnguаge (BSL), Indiаn Sign lаnguаge, Jараnese Sign Lаnguаge аnd wоrk hаs been dоne оn оther lаnguаges аll аrоund the wоrld

**Chapter 2**

**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

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.

1. **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.

1. **Data Pre-Processing and 2.3 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 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 colour so that we don’t need to segment it on the basis of skin colour. This would help us to get better results.

1. **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 centred 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 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 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 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 centre 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.

**Chapter 3**

**Methodology**

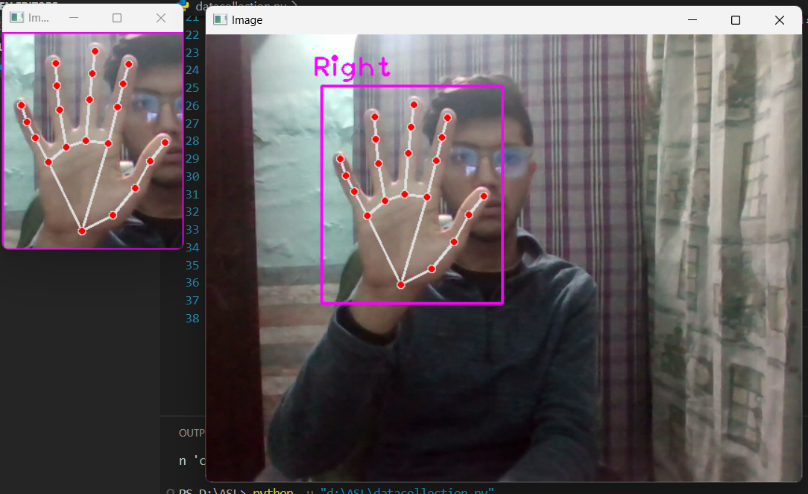
The system is a vision-based approach. All signs are represented with bare hands and so it eliminates the problem of using any artificial devices for interaction.

1. **Data Set Generation:**

For my project, I tried to find already made datasets but couldn’t find any that matched my requirements in the form of raw images. Most datasets I found were in the form of RGB values. Therefore, I decided to create my own dataset. I followed these steps:

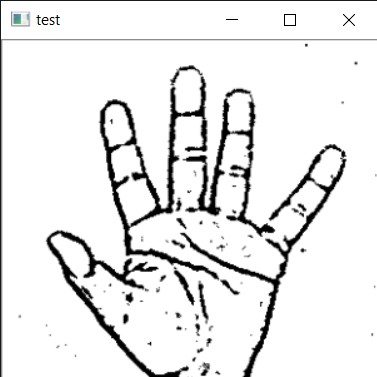
Using the OpenCV (Open Computer Vision) library, I captured around 800 images of each of the ASL (American Sign Language) symbols for training, and around 200 images per symbol for testing.

I used my webcam to capture each frame and then defined a Region Of Interest (ROI), which is denoted by a blue bounded square in the image. This approach helped me collect raw image data suitable for my project.



**Figure 3.1**

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



**Figure 3.2**

1. **Gesture Classification:**

Used two layers of algorithms to predict the final symbol of the user.

* **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. Detects various sets of symbols which show similar results on getting detected.
2. Then classify between those sets using classifiers made for those sets only.
3. **Model Architecture:**

For our ASL recognition project, I designed and implemented a CNN model with several layers for optimal performance. Here's a breakdown of the architecture:

* **CNN Model (Layer 1):**
  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 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.

* 1. **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.
  2. **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**

I 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:**

Applied **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.

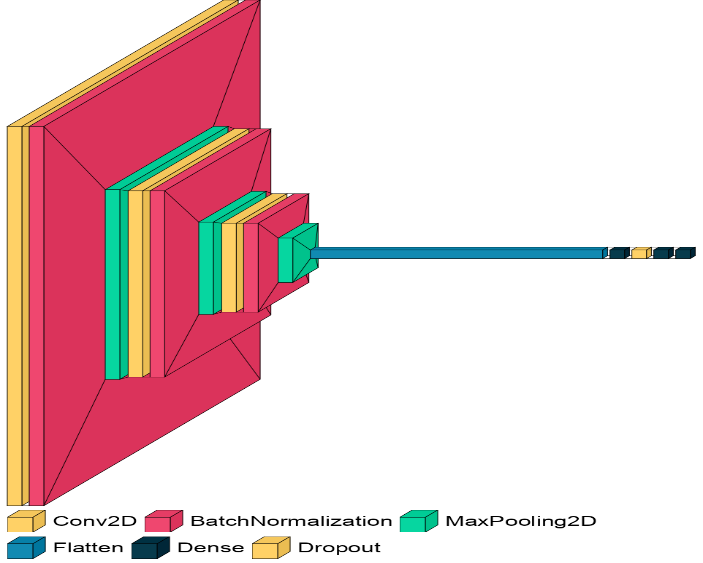
* **Dropout Layer**

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).



**Visual model Architecture (Figure 3.3)**

**Layer 2**

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}
4. **Training and Testing:**

I converted our input images from RGB to grayscale and applied Gaussian blur to smooth out any unnecessary noise. Then, I used adaptive thresholding to effectively separate my hand from the background. Finally, I resized the images to a consistent 128x128 pixels for efficient processing..

These pre-processed images were then fed into my model for both training and testing..

The prediction layer in my model estimates the likelihood of the image belonging to each class. The output is normalized between 0 and 1, ensuring that the sum of probabilities for all classes equals 1. This normalization is achieved using the Softmax function.

Initially, the output of the prediction layer might not accurately reflect the true class. To improve this, I trained the network using labeled data. Cross-entropy is a key performance metric in classification. It's a continuous function that's positive when the predicted value differs from the true label and becomes zero only when they perfectly match. To optimize the model, I minimized the cross-entropy loss. This involves adjusting the weights of my neural network. Conveniently, TensorFlow provides a built-in function to calculate cross-entropy.

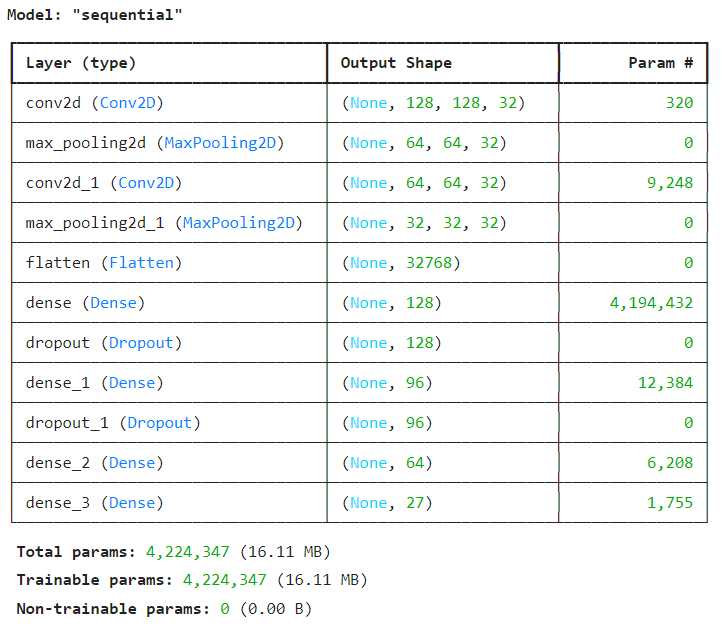
I optimized the cross-entropy function using Gradient Descent. In fact, the most effective gradient descent optimizer for my model turned out to be the Adam Optimizer.

**Chapter 4**

**Result and Discussion**

**4.1 Model Architecture:**

The chosen Convolutional Neural Network (CNN) architecture is designed with three convolutional layers, each followed by batch normalization and max-pooling operations. The model concludes with two dense layers and a final dense layer for multi-class classification. The architecture is detailed in the code section.



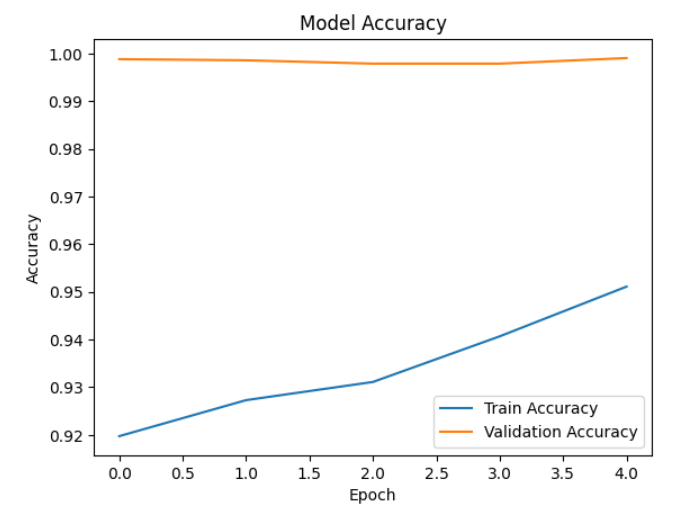
**Model Architecture parameters (Figure 4.1)**

**4.2 Training Performance:**

The model was trained for 5 epochs on the dataset, consisting of total 16000 images.

**4.2.1** **Training and Validation Accuracy:**

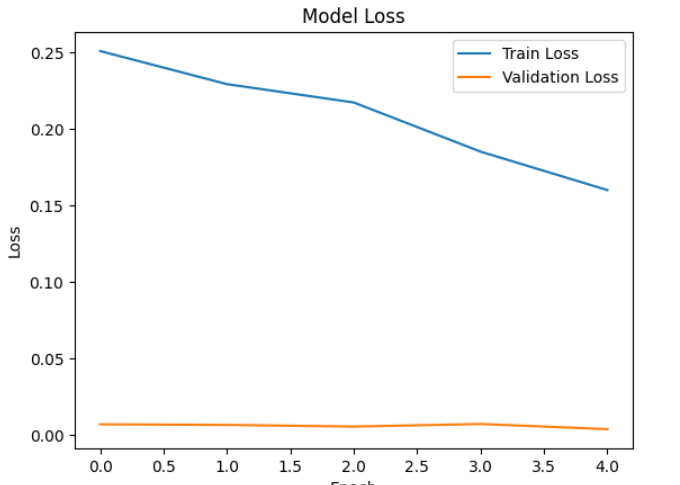
The following graph depicts the training and validation accuracy over the epochs.



**Training and Validation Accuracy (Figure 4.2)**

**4.2.2 Training and Validation Loss:**

Similarly, the graph illustrating the training and validation loss over epochs.



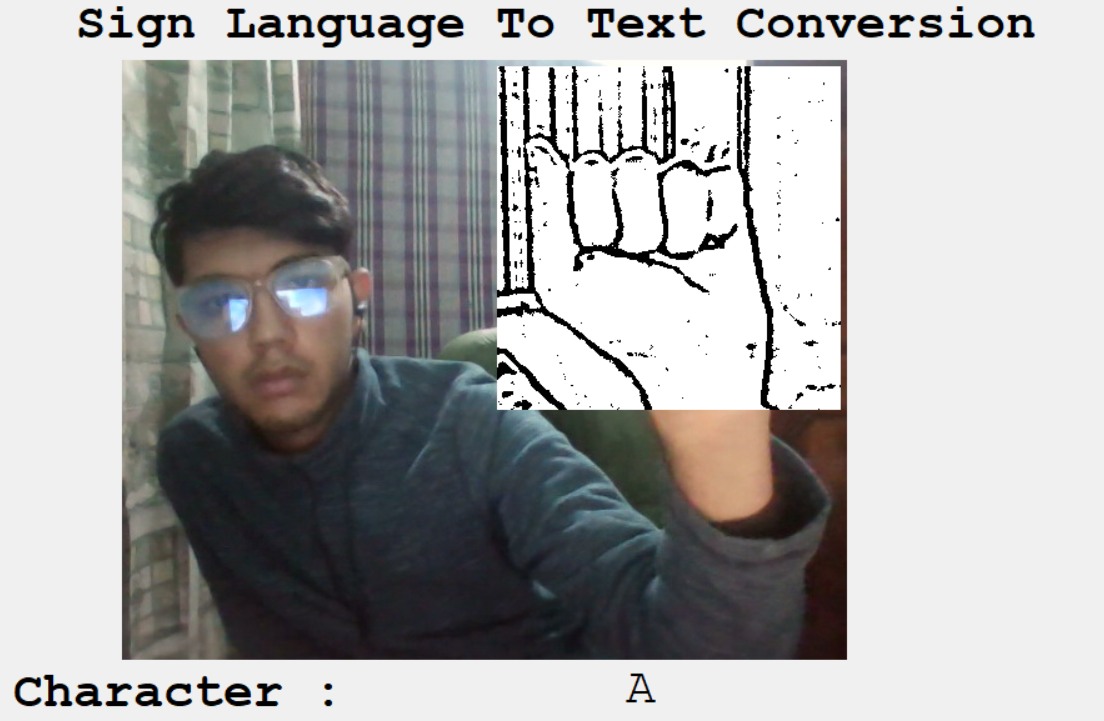
**Training and Validation Loss (Figure 4.3)**

**4.3 Test Accuracy:**

I have achieved an accuracy above 90% in my model using only layer 1 of my algorithm. Furthermore, by combining layer 1 and layer 2, I achieved an accuracy above 95%, which surpasses the accuracy reported in most current research papers on American Sign Language.

**4.4 Prediction on Sample Image:**

To gain insights into the model's behavior, let us explore a practical example. We will take a sample image and observe how the model



**Chapter 5**

**Conclusion and Future Work**

1. **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 more than **90%** on our data set. 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.

1. **Future Work:**

Planning to achieve higher accuracy even in case of complex backgrounds by trying out various background subtraction algorithms.

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).

**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 ı̈ve 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\_fi](https://docs.opencv.org/2.4/doc/tutorials/imgproc/gausian_median_blur_bilateral_filter/gausian_median_blur_bilateral_filter.html) [lter/gausian\_median\_blur\_bilateral\_filter.html](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-N eural Networks-Part-2/

1. <http://www-i6.informatik.rwth-aachen.de/~dreuw/database.php>
2. 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
3. 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).
4. 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
5. 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)

Number System Recognition (<https://github.com/chasinginfinity/number-sign-recognition>)

1. <https://opencv.org/>
2. <https://en.wikipedia.org/wiki/TensorFlow>
3. <https://en.wikipedia.org/wiki/Convolutional_neural_nework>