SIGN LANGUAGE CHARACTER RECOGNITION

A PROJECT REPORT

Submitted in the partial fulfilment of the requirements
for the award of the degree of

BACHELOR OF TECHNOLOGY
IN
COMPUTER ENGINEERING

Under the Supervision of
Mr. Sarfaraz Masood
Assistant Professor,
Dept. of Computer Engg.
Jamia Millia Islamia

Submitted by
Harish Chandra Thuwal
(13-BCS-0027)
Adhyan Srivastava
(13-BCS-0007)

DEPARTMENT OF COMPUTER ENGINEERING
FACULTY OF ENGINEERING AND TECHNOLOGY
JAMIA MILLIA ISLAMIA, NEW DELHI - 110017
(Year-2016)

DECLARATION

We, **Harish Chandra Thuwal** and **Adhyan Srivastava**, students of Bachelor of Technology, Computer Engineering hereby declare that the project entitled "Sign Language Character Recognition" which is submitted by us to the Department of Computer Engineering, Faculty of Engineering and Technology, Jamia Millia Islamia, New Delhi in partial fulfilment of requirements for the award of the degree of Bachelor of Technology in Computer Engineering, has not been formed the basis for the award of any degree, diploma or other similar title or recognition.

Place: New Delhi

Date:

Harish Chandra Thuwal
(13BCS0027)

Adhyan Srivastava
(13BCS0007)

CERTIFICATE

This is to certify that the dissertation/project report(Course Code) entitled "Sign Language Character Recognition", is an authentic work carried out by **Harish Chandra Thuwal** and **Adhyan Srivastava.**

The work is submitted in partial fulfilment of the requirements for the award of the degree of Bachelor of Technology in Computer Engineering under my guidance. The matter embodied in this project work has not been submitted earlier for the award of any degree or diploma to the best of my knowledge and belief.

Date: 20/12/2016

Mr. Sarfaraz Masood,
Assistant Professor,
Dept. of Computer Engg.
Jamia Millia Islamia

ACKNOWLEDGMENT

We would like to thank our mentor **Mr. Sarfaraz Masood** (Assistant Professor, Department of Computer Engineering) for giving us the opportunity to undertake the project. We thank them for their immense guidance, and appreciate their timely engagement.

We would like to extend special gratitude to the the assistants and lab coordinators of the Department for providing us the infrastructural facilities necessary to sustain the project.

\

ABSTRACT

Inability to speak is considered to be true disability. People with this disability use different modes to communicate with others, there are number of methods available for their communication one such common method of communication is sign language.

Developing sign language application for deaf people can be very important, as they'll be able to communicate easily with even those who don't understand sign language. Our project aims at taking the basic step in bridging the communication gap between normal people and deaf and dumb people using sign language.

The main focus of this work is to create a vision based system to identify Finger spelled letters of ASL. The reason for choosing a system based on vision relates to the fact that it provides a simpler and more intuitive way of communication between a human and a computer. In this report, 36 different categories have been considered: 26 categories for English Alphabets (a-z) and 10 categories for Numerals (0-9).

We used two approaches for the classification of sign language :-

1. In the first approach, features were extracted from the images using SIFT(scale invariant vector transform) which were then plotted into histograms and used for training hierarchical SVM.The accuracy of the model obtained using this approach was 44.259%.

2. In the second approach, Convolutional neural network was used. The accuracy of the model obtained using Convolutional Neural Network was 95.50%.

TABLE OF CONTENTS

LIST OF FIGURES

1. INTRODUCTION

Motion of any body part like face, hand is a form of gesture. Here for gesture recognition we are using image processing and computer vision. Gesture recognition enables computer to understand human actions and also acts as an interpreter between computer and human. This could provide potential to human to interact naturally with the computers without any physical contact of the mechanical devices. Gestures are performed by deaf and dumb community to perform sign language. This community used sign language 3 for their communication when broadcasting audio is impossible, or typing and writing is difficult, but there is the vision possibility. At that time sign language is the only way for exchanging information between people. Normally sign language is used by everyone when they do not want to speak, but this is the only way of communication for deaf and dumb community. Sign language is also serving the same meaning as spoken language does. This is used by deaf and dumb community all over the world but in their regional form like ISL, ASL. Sign language can be performed by using Hand gesture either by one hand or two hands. It is of two type Isolated sign language and continuous sign language. Isolated sign language consists of single gesture having single word while continuous ISL or Continuous Sign language is a sequence of gestures that generate a meaningful sentence. In this report we performed isolated ASL gesture recognition technique.

1.1 Sign Language

Deaf people around the world communicate using sign language as distinct from spoken language in their every day a visual language that uses a system of manual, facial and body movements as the means of communication. Sign language is not an universal language, and different sign languages are used in different countries, like the many spoken languages all over the world. Some countries such as Belgium, the UK, the USA or India may have more than one sign language. Hundreds of sign languages are in used around the world, for instance, Japanese Sign Language, British Sign Language (BSL), Spanish Sign Language, Turkish Sign Language.

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.

We have used American Sign Language(ASL) as a base in our project. American Sign Language is a complex visual-spatial language that is used by the deaf community in the United States and English-speaking parts of Canada. It is linguistically complete, natural language. It is native language of many deaf men and women, as well as some hearing children born into deaf families.

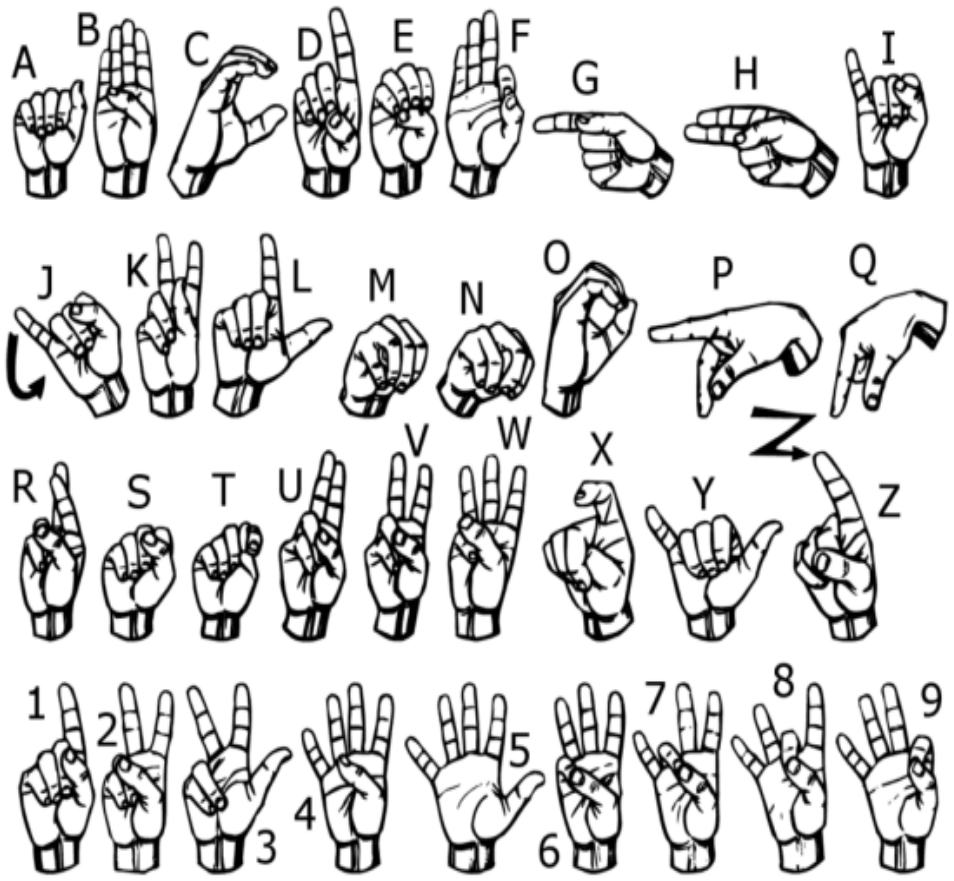


Fig 1: American Sign Language[18]

2. LITERATURE SURVEY

In the recent years, there has been tremendous research on the hand sign recognition. The technology of gesture recognition is divided into two categories

2.1 Vision-based

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. This poses a challenging problem as these systems need to be background invariant, lighting insensitive, person and camera independent to achieve real time performance. Moreover, such systems must be optimized to meet the requirements, including accuracy and robustness.

The vision based hand gesture recognition system is shown in fig.--:

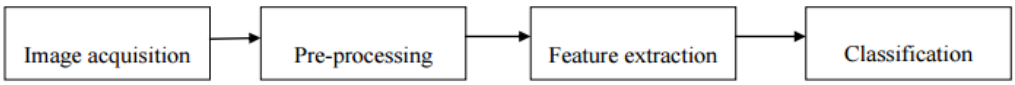


Fig 2: Block Diagram of vision based recognition system

Vision based analysis, is based on the way human beings perceive information about their surroundings, yet it is probably the most difficult to implement in a satisfactory way. Several different approaches have been tested so far.

1. One is to build a three-dimensional model of the human hand. The model is matched to images of the hand by one or more cameras, and parameters corresponding to palm orientation and joint angles are estimated. These parameters are then used to perform gesture classification.

2. Second one to capture the image using a camera then extract some feature and those features are used as input in a classification algorithm for classification.

2.1.1 Indian Sign Language Recognition Based on Histogram Of Oriented Gradient and NN [16]

In this paper, a method for hand gesture recognition of Indian sign language is proposed. The accurate classification of hand gestures plays a vital role to develop an efficient hand gesture recognition system. To implement this approach they have utilized a simple web camera to capture hand gesture images. They have attempted to propose a system to recognize alphabets characters (A-Z) and numerals (0-9) using Histograms of Oriented Gradients (HOG) features. Their purpose is to implement an algorithm to extract Histogram of Gradient Orientation (HOG) features and these features are used to pass in neural network training for the gesture recognition purpose.

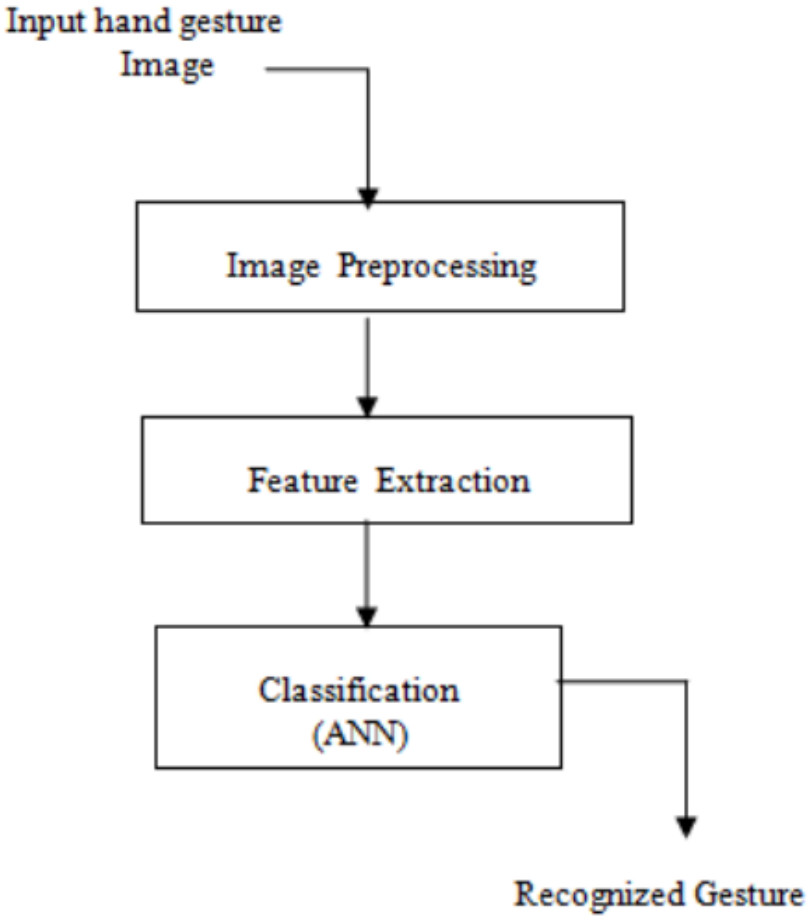


Fig 3: Block Diagram of Hand Gesture Recognition System

2.1.2 Automatic Indian Sign Language Recognition for Continuous Video Sequence [15]

The proposed system comprises of four major modules: Data Acquisition, Pre-processing, Feature Extraction and Classification. Pre-processing stage involves Skin Filtering and histogram matching after which Eigen-vector based Feature Extraction and Eigen value weighted Euclidean distance based Classification Technique was used. 24 different alphabets were considered in this paper where 96% recognition rate was obtained.

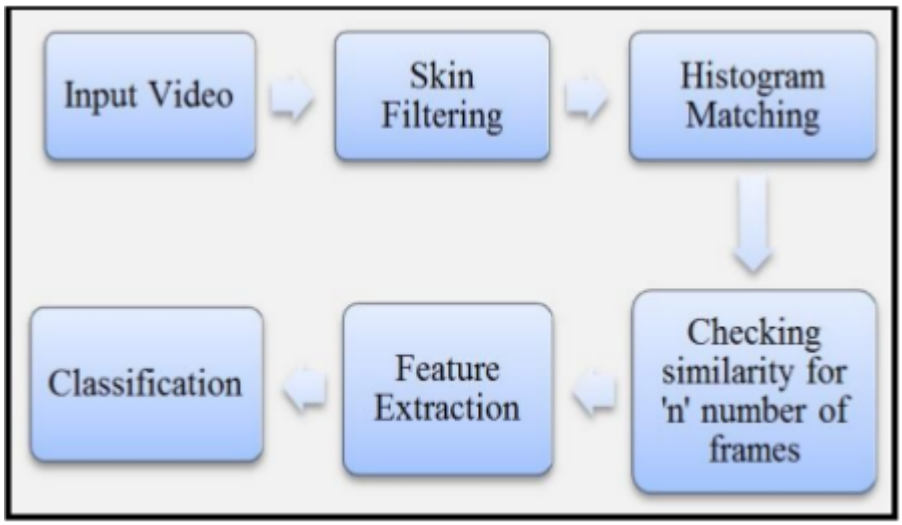


Fig 4: System Overview

2.2 Glove-Based

In glove based systems, data gloves are used which can achieve the accurate positions of hand gestures as its positions are directly measured. The Data-Glove based methods use sensor devices for digitizing hand and finger motions into multi-parametric data. The extra sensors make it easy to collect hand configuration and movement. However, the devices are quite expensive and bring much cumbersome experience to the users some of the earlier gesture recognition systems attempted to identify gestures using glove-based devices that would measure the position and joint angles of the hand. However, these devices are very cumbersome and usually have many cables connected to a computer. This has brought forth the motivation of using non-intrusive, vision-based approaches for recognizing gestures Also the sensors used for the detection of the sign language and the gesture recognition in the system that are available in the market are quite costly. In computer recognition of spoken language, speech data is captured using a microphone connected to an ADC.

Similarly a data-capturing device is also required in order to recognize sign language; in this case measuring the position and movement of the signer's hands.

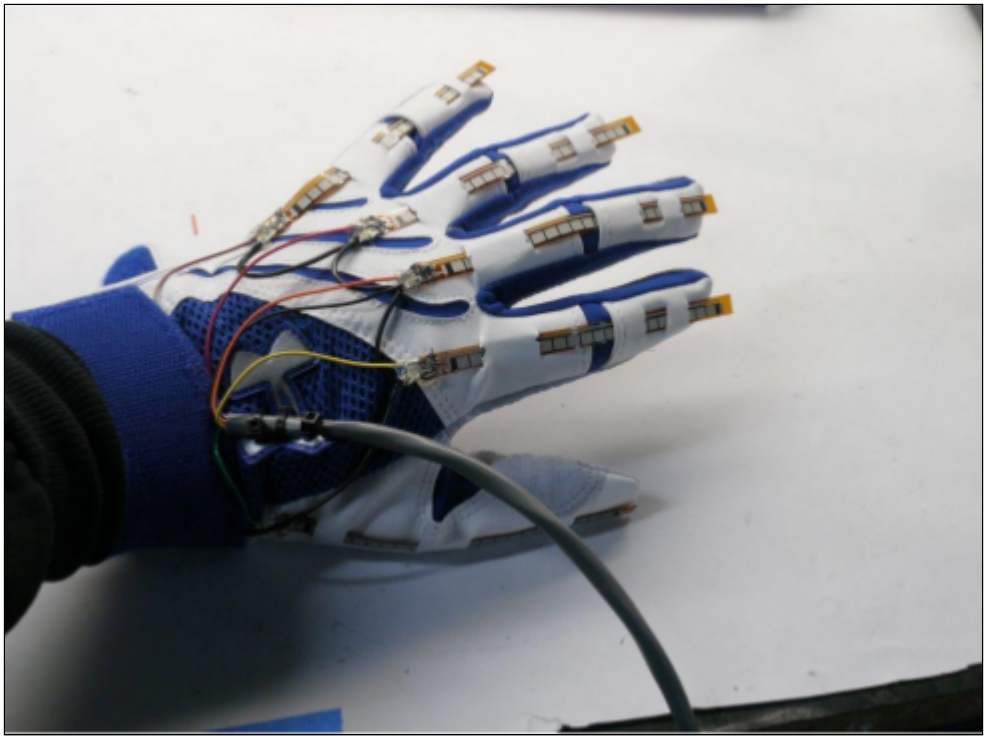


Fig 5: Gloves with flex sensor mounted on it

2.2.1 Hand Gesture Recognition Using Flex Sensors [17]

In this system an electro-mechanical robot is designed and controlled using hand gesture in real time. The system is designed on microcontroller platform using Keil and MPLAB tools. Hand gesture recognition is done on the principle of resistance change sensed through flex sensor. These sensors are integrated in a hand gloves from which input to the system is given. The designed system is divided into two sections as transmitter and receiver. The transmitter section will be in hand gloves from which the data is sensed and processed through PIC16F7487 and send serially to the receiver section. RF technology is used to transmit the data at the receiver section at the frequency of 2.4 GHz. ARM 7 (LPC2148) processor is used to receive the data. Here from the received data, the character is predicted and matched with the closest character from which the character is identified and displayed on LCD. The various case studies is prepared for the designed system and tested in real time. The proposed system can be used for the various applications such as in unmanned machines, industries, handicapped personnel etc.

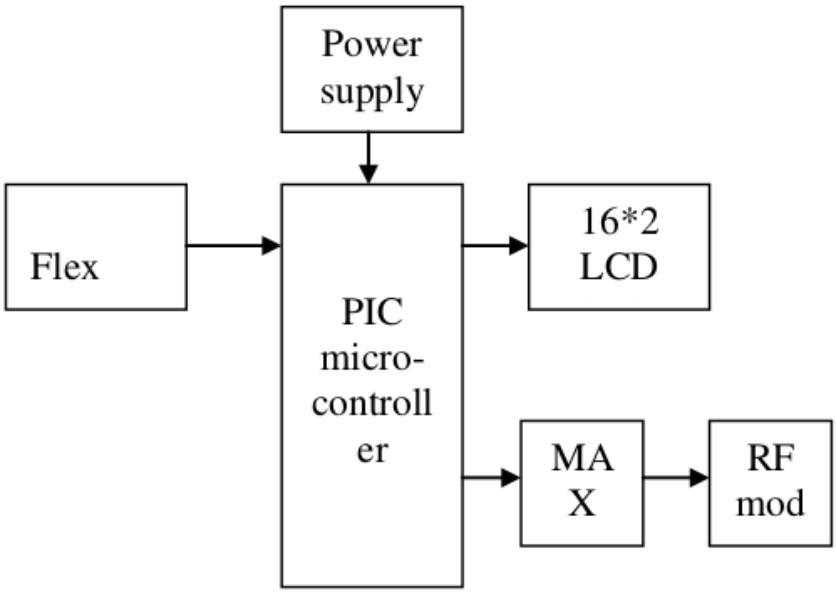


Fig 6:
Transmitter section for hand gesture recognition

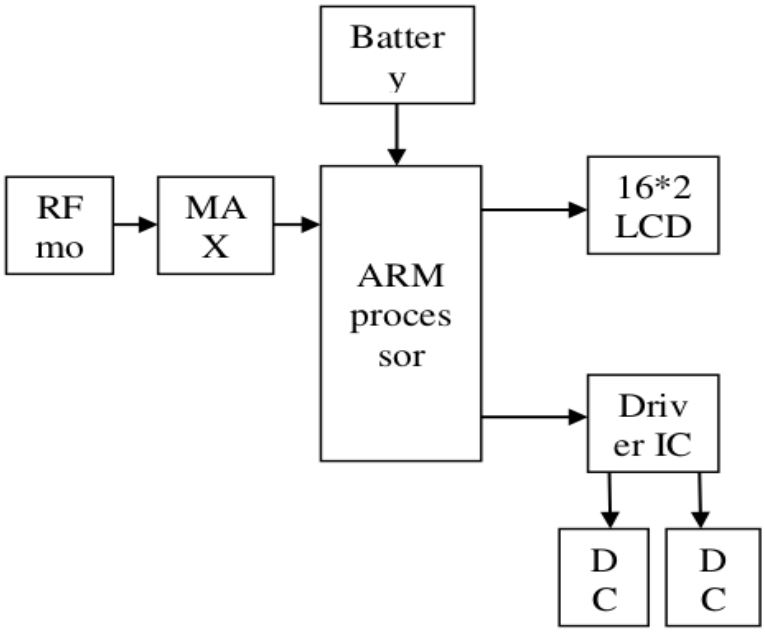


Fig 7:
Receiver section for hand gesture recognition

3. ALGORITHMS

3.1 Scale Invariant Feature Transform(SIFT)

Scale-invariant feature transform (SIFT) is an algorithm in computer vision to detect and describe local features in images. The algorithm was patented in the US by the University of British Columbia and published by David Lowe in 1999.

Applications include object recognition, robotic mapping and navigation, image stitching, 3D modeling, gesture recognition, video tracking, individual identification of wildlife and match moving.

SIFT can robustly identify objects even among clutter and under partial occlusion, because the SIFT feature descriptor is invariant to uniform scaling, orientation, and partially invariant to affine distortion and illumination changes.

The Scale-Invariant Feature Transform (SIFT) bundles a feature detector and a feature descriptor. The detector extracts from an image a number of frames (attributed regions) in a way which is consistent with (some) variations of the illumination, viewpoint and other viewing conditions. The descriptor associates to the regions a signature which identifies their appearance compactly and robustly.

Fig 8: Some of the detected SIFT frames [12]

The SIFT detector is controlled mainly by two parameters: **the peak threshold** and **the (non) edge threshold.**

The peak threshold filters peaks of the DoG scale space that are too small (in absolute value). For instance, consider a test image of 2D Gaussian blobs:

Fig 9: Test Image For Peak Threshold [12]

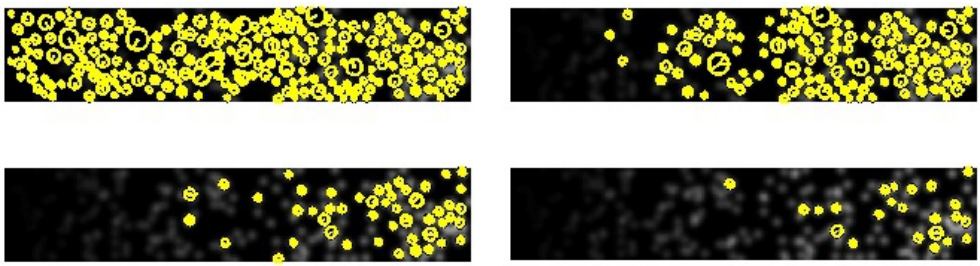


Fig 10: Detected frames for increasing peak threshold.
From top: peak_thresh = {0,10,20,30} [12]

The **edge threshold** eliminates peaks of the DoG scale space whose curvature is too small (such peaks yield badly localized frames).

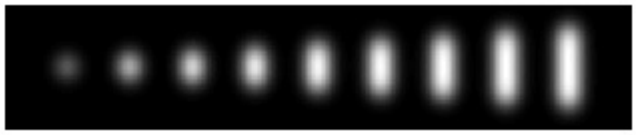


Fig 11: A test image for the edge threshold parameter. [12]

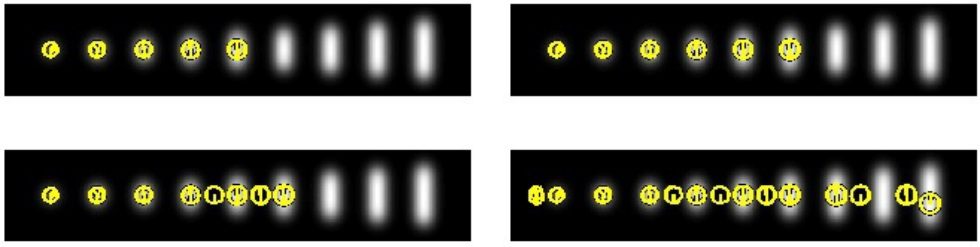
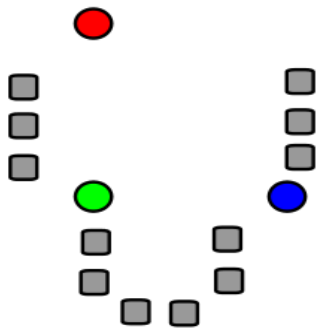


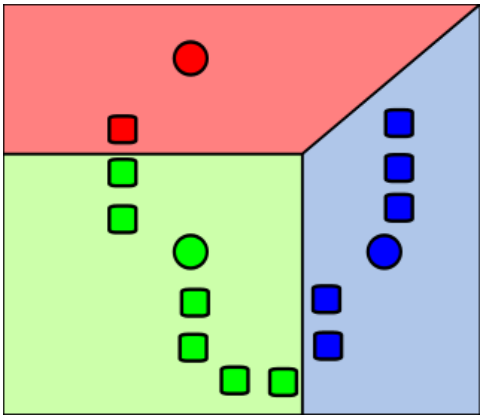
Fig 12: Detected frames for increasing edge threshold.
From top: edge_thresh = {3.5, 5, 7.5, 10} [12]

3.2 K MEANS CLUSTERING

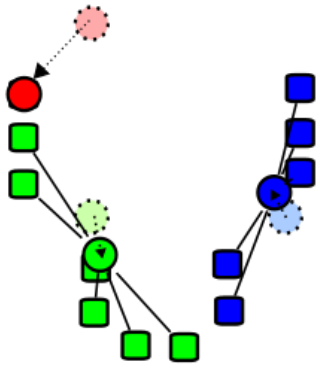
The k-means algorithm takes as input the number of clusters to generate, k, and a set of observation vectors to cluster. It returns a set of centroids, one for each of the k clusters. An observation vector is classified with the cluster number or centroid index of the centroid closest to it.



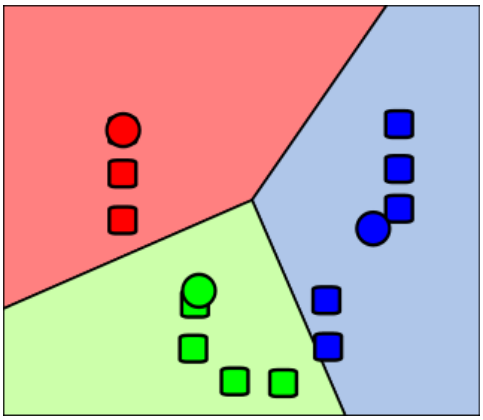
1. K initial "means" (in this case k = 3) Randomly generated within the data domain (shown in color)



2. K clusters are created by associating every observation with the nearest mean



3. The centroid of each of the k clusters becomes the new mean.



4. Steps 2 and 3 are repeated until convergence has been reached.

Fig 13: K Means Clustering [13]

A vector v belongs to cluster i if it is closer to centroid i than any other centroids. If v belongs to i, we say centroid i is the dominating centroid of v. The k-means algorithm tries to minimize distortion, which is defined as the sum of the squared distances between each observation vector and its dominating centroid. Each step of the k-means algorithm refines the choices of

centroids to reduce distortion. The change in distortion is used as a stopping criterion: when the change is lower than a threshold, the k-means algorithm is not making sufficient progress and terminates. One can also define a maximum number of iterations.

The centroid index or cluster index is also referred to as a “code” and the table mapping codes to centroids and vice versa is often referred as a “code book”. The result of k-means, a set of centroids, can be used to quantize vectors. Quantization aims to find an encoding of vectors that reduces the expected distortion.

As an example, suppose we wish to compress a 24-bit color image (each pixel is represented by one byte for red, one for blue, and one for green) before sending it over the web. By using a smaller 8-bit encoding, we can reduce the amount of data by two thirds. Ideally, the colors for each of the 256 possible 8-bit encoding values should be chosen to minimize distortion of the color. Running k-means with k=256 generates a codebook of 256 codes, which fills up all possible 8-bit sequences. Instead of sending a 3-byte value for each pixel, the 8-bit centroid index (or code word) of the dominating centroid is transmitted. The code book is also sent over the wire so each 8-bit code can be translated back to a 24-bit pixel value representation.

Three key features of *k*-means which make it efficient are often regarded as its biggest drawbacks:

1. Euclidean distance is used as a metric and variance is used as a measure of cluster scatter.

2. The number of clusters *k* is an input parameter: an inappropriate choice of *k* may yield poor results. That is why, when performing k-means, it is important to run diagnostic checks for determining the number of clusters in the data set.

3. Convergence to a local minimum may produce counterintuitive ("wrong") results.

3.3 Support Vector Machines (SVM)

Support Vector Machine (SVM) is a supervised machine learning algorithm which can be used for both classification or regression challenges. However, it is mostly used in classification problems. In this algorithm, we plot each data item as a point in n-dimensional space (where n is number of features you have) with the value of each feature being the value of a particular coordinate. Then, we perform classification by finding the hyperplane that differentiate the two classes very well. Given a set of training examples, each marked as belonging to one or the other of two categories, an SVM training algorithm builds a model that assigns new examples to one category or the other, making it a non-probabilistic binary linear classifier.

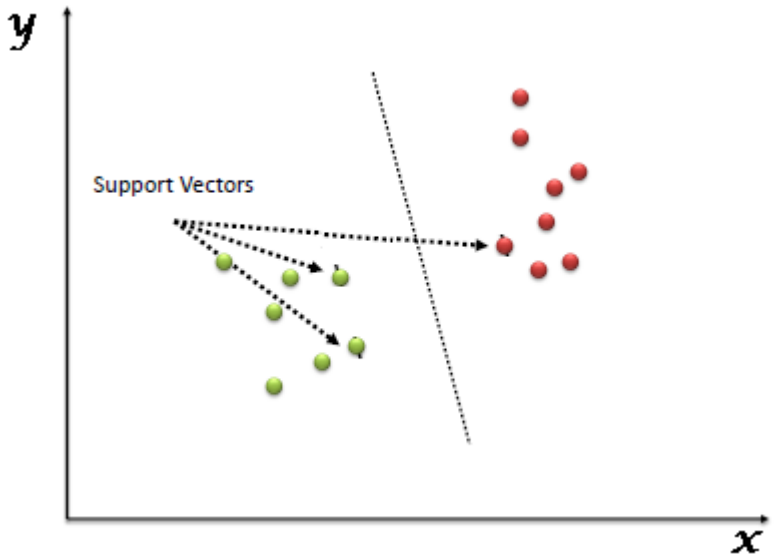


Fig 14 [11]

There are many hyperplanes that might classify the data. One reasonable choice as the best hyperplane is the one that represents the largest separation, or margin, between the two classes. So we choose the hyperplane so that the distance from it to the nearest data point on each side is maximized. If such a hyperplane exists, it is known as the maximum-margin hyperplane and the linear classifier it defines is known as a maximum margin classifier; or equivalently, the perceptron of optimal stability.

Identifying the right HyperPlane

Scenario 1:

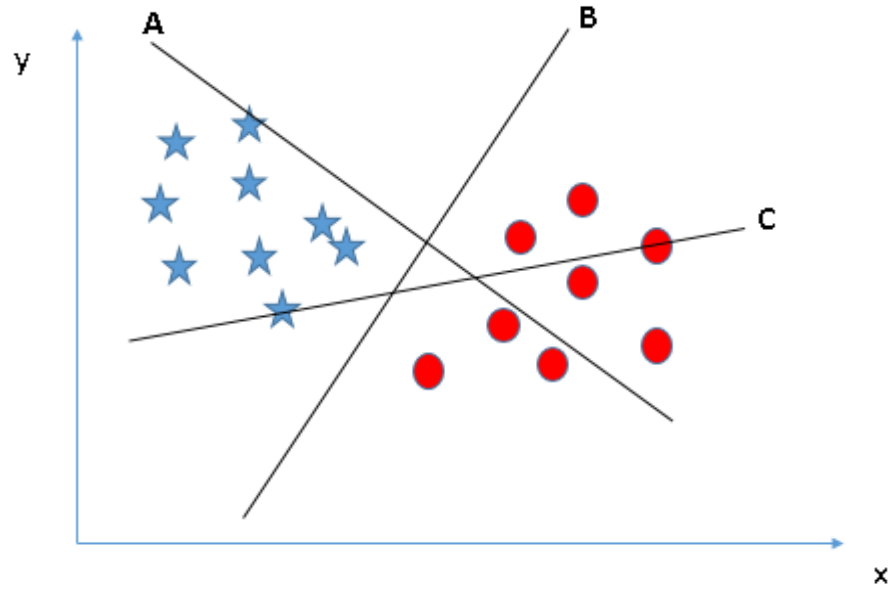


Fig 14.1 [11]

Select the hyper-plane which segregates the two classes better. In this scenario, hyperplane "B" has excellently performed this job.

Scenario 2:

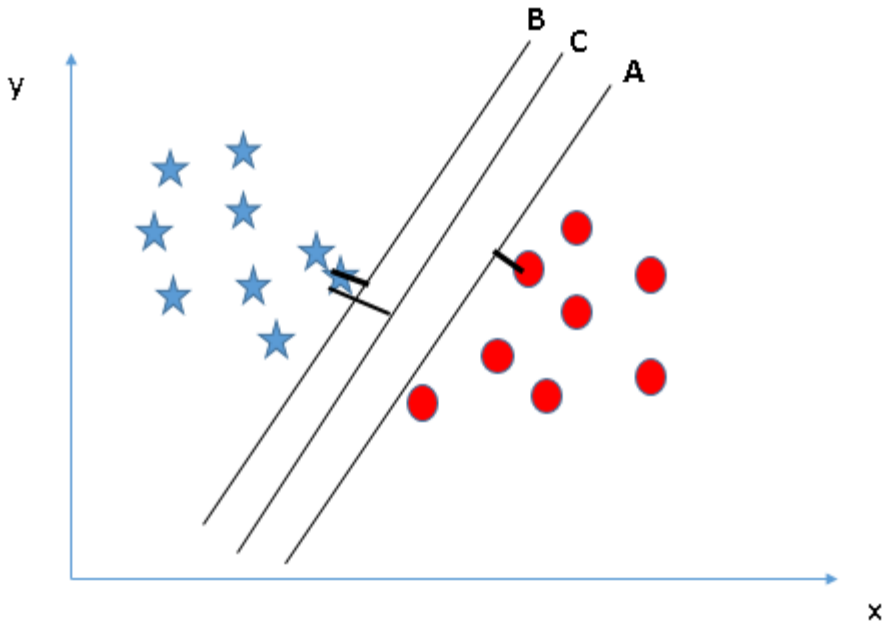


Fig 14.2 [11]

All three hyperplanes are segregating the classes well. Here maximize the distances between nearest data point(either class) and hyperplane. The distance is called as Margin.The margin for hyperplane C is high as compared to both A and B. Hence, we name the right hyperplane as C. Selecting the hyperplane having high margin then the chances of misclassification are less and the model becomes more robust.

Scenario 3:

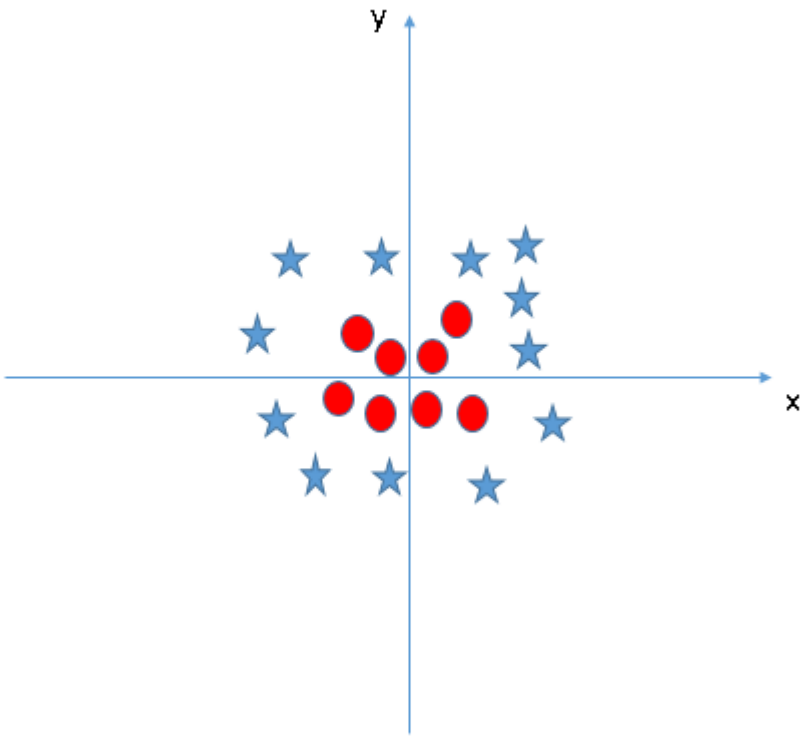


Fig 14.3 [11]

In the scenario below, we can't have linear hyper-plane between the two classes. SVM has a technique called the **kernel trick**. These are functions which takes low dimensional input space and transform it to a higher dimensional space i.e. it converts not separable problem to separable problem, these functions are called kernels. It is mostly useful in non-linear separation problem.

SVM solves the above problem by introducing addition feature $z=x^2+y^2$ using kernel trick. Now plot the data points on axis x and z.

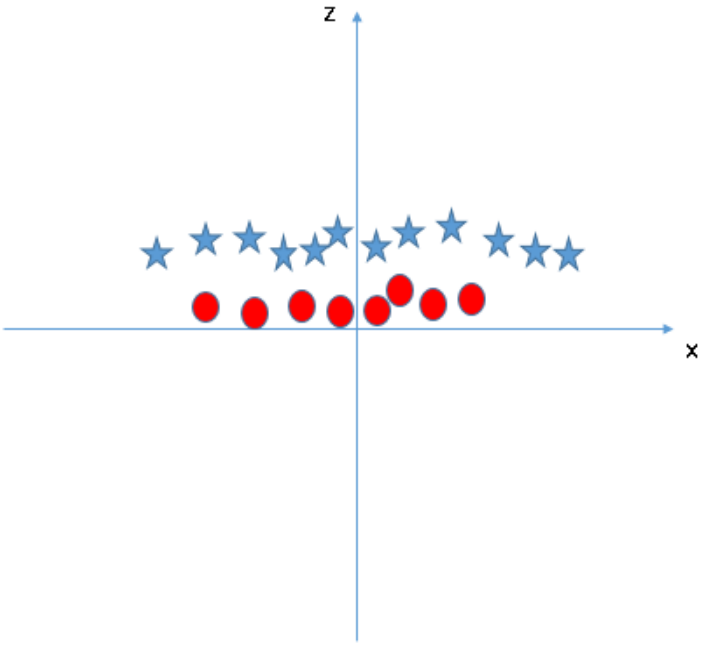


Fig 14.4

In the original plot, red circles appear close to the origin of x and y axes, leading to lower value of z and star relatively away from the origin result to higher value of z.

Simply put, kernel trick does some extremely complex data transformations, then find out the process to separate the data based on the labels or outputs defined.

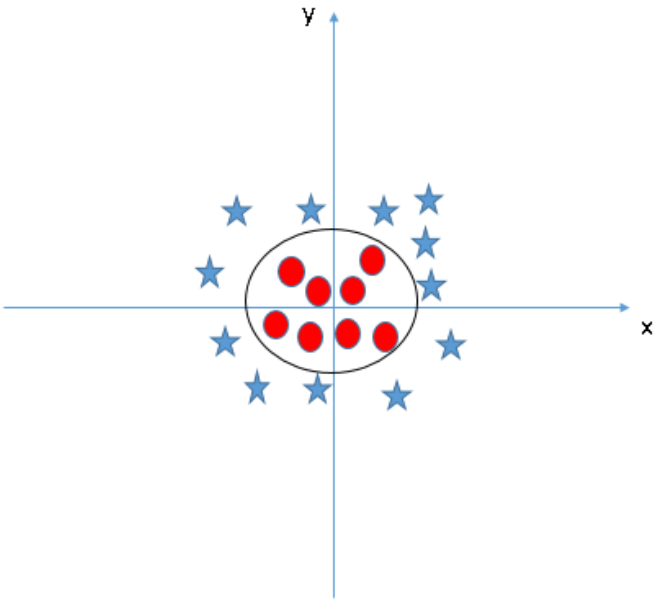


Fig 14.5 [11]

When we look at the hyperplane in original input space it looks like a circle.

3.4 CONVOLUTIONAL NEURAL NETWORK(CNN)

Neural networks, as its name suggests, is a machine learning technique which is modeled after the brain structure. It comprises of a network of learning units called neurons. These neurons learn how to convert **input signals** (e.g. picture of a cat) into corresponding **output signals** (e.g. the label “cat”), forming the basis of automated recognition.

A convolutional neural network (CNN, or ConvNet) is a type of feed-forward artificial neural network in which the connectivity pattern between its neurons is inspired by the organization of the animal visual cortex.

CNNs have repetitive blocks of neurons that are applied across space (for images) or time (for audio signals etc). For images, these blocks of neurons can be interpreted as 2D convolutional kernels, repeatedly applied over each patch of the image. For speech, they can be seen as the 1D convolutional kernels applied across time-windows. At training time, the weights for these repeated blocks are 'shared', i.e. the weight gradients learned over various image patches are averaged.

3.4.1 CNN Summarized in 4 Steps

There are four main steps in CNN: convolution, subsampling, activation and full connectedness.

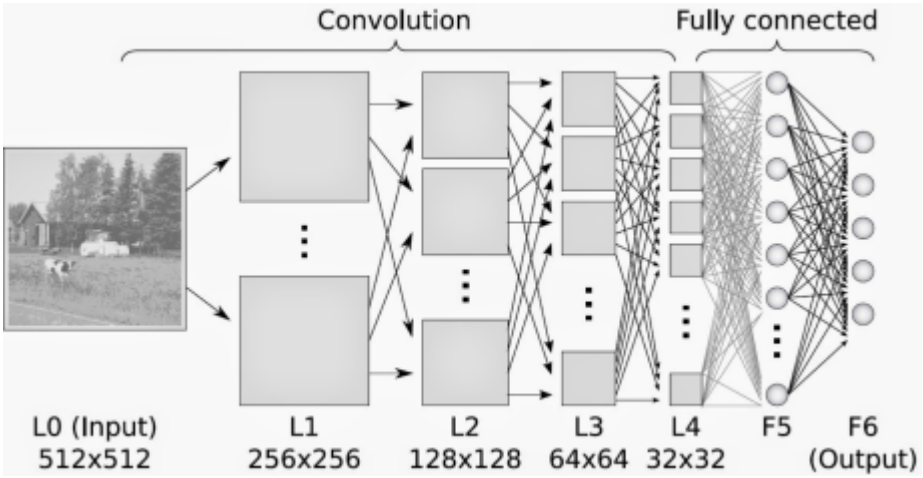


Fig 15: Convolutional neural network [14]

3.4.1.1 Convolution

The first layers that receive an input signal are called convolution filters. Convolution is a process where the network tries to label the input signal by

referring to what it has learned in the past. If the input signal looks like previous cat images it has seen before, the "cat" reference signal will be mixed into, or convolved with, the input signal. The resulting output signal is then passed on to the next layer.

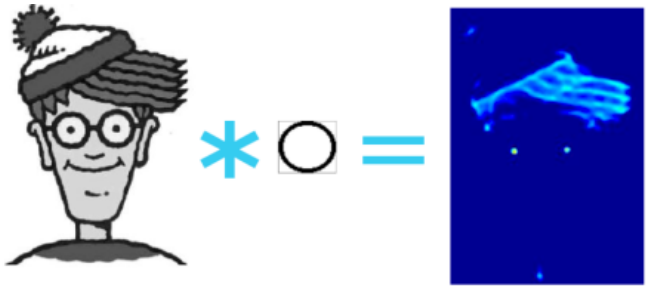


Fig 16: Convolving Wally with a circle filter. The circle filter responds strongly to the eyes.

Convolution has the nice property of being **translational invariant**. Intuitively, this means that each convolution filter represents a feature of interest (e.g whiskers, fur), and the CNN algorithm learns which features comprise the resulting reference (i.e. cat). The output signal strength is not dependent on where the features are located, but simply whether the features are present. Hence, a cat could be sitting in different positions, and the CNN algorithm would still be able to recognize it.

3.4.1.2 Subsampling

Inputs from the convolution layer can be "smoothened" to reduce the sensitivity of the filters to noise and variations. This smoothing process is called **subsampling**, and can be achieved by taking averages or taking the maximum over a sample of the signal. Examples of subsampling methods (for image signals) include reducing the size of the image, or reducing the color contrast across red, green, blue (RGB) channels.

Fig 17: Sub sampling Wally by 10 times. This creates a lower resolution image.

3.4.1.3 Activation

The activation layer controls how the signal flows from one layer to the next, emulating how neurons are fired in our brain. Output signals which are strongly associated with past references would activate more neurons, enabling signals to be propagated more efficiently for identification.

CNN is compatible with a wide variety of complex activation functions to model signal propagation, the most common function being the Rectified Linear Unit (ReLU), which is favored for its faster training speed.

3.4.1.4 Fully Connected

The last layers in the network are fully connected, meaning that neurons of preceding layers are connected to every neuron in subsequent layers. This mimics high level reasoning where all possible pathways from the input to output are considered.

3.4.1.5 (During Training) Loss

When training the neural network, there is additional layer called the loss layer. This layer provides feedback to the neural network on whether it identified inputs correctly, and if not, how far off its guesses were. This helps to guide the neural network to reinforce the right concepts as it trains. This is always the last layer during training.

3.4.2 Implementation

Algorithms used in training CNN are analogous to studying for exams using flash cards. First, you draw several flashcards and check if you have mastered the concepts on each card. For cards with concepts that you already know, discard them. For those cards with concepts that you are unsure of, put them back into the pile. Repeat this process until you are fairly certain that you know enough concepts to do well in the exam. This method allows you to focus on less familiar concepts by revisiting them often. Formally, these algorithms are called gradient descent algorithms for forward pass learning. Modern deep learning algorithm uses a variation called stochastic gradient descent, where instead of drawing the flashcards sequentially, you draw them at random. If similar topics are drawn in sequence, the learners might over-estimate how well they know the topic. The random approach helps to minimize any form of bias in the learning of topics.

Learning algorithms require feedback. This is done using a **validation set** where the CNN would make predictions and compare them with the true labels or ground truth. The predictions which errors are made are then fed backwards to the CNN to refine the weights learned, in a so called backwards pass. Formally, this algorithm is called **backpropagation of errors**, and it requires functions in the CNN to be differentiable (almost).

CNNs are too complex to implement from scratch. Today, machine learning practitioners often utilize toolboxes developed such as Caffe, Torch, MatConvNet and Tensor flow for their work.

4. EXPERIMENTAL DESIGN

4.1 FIRST APPROACH (SIFT AND SVM)

4.1.1 Data Set Used

The data set used for this approach consists of 2524 American Sign Language(ASL) Gestures, with 70 gestures belonging to each of the 36 categories : 26 categories for English Alphabets (a-z) and 10 categories for Numerals (0-9). 1975 (nearly 55 images per category) images were used for training and remaining 540 were used for training.

4.1.2 Methodology

The **first stage** was to segment the skin part from the image, as the remaining part can be regarded as noise with respect to the character classification problem.

The **second stage** was to extract relevant features from the skin segmented images which can prove significant for the next stage i.e. learning and classification.

The **third stage** was to use the extracted features as input into supervised learning models for training and then finally use the trained models for classification.

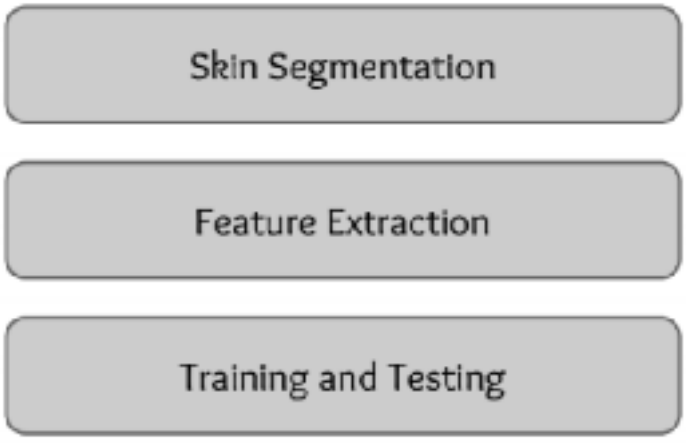


Fig 18

4.1.2.1 Image Segmentation

Transform the image from RGB space to YIQ and YUV space. Retain pixels with 30<I<100 and 105<theta<150 ,theta = tanh(U/V).

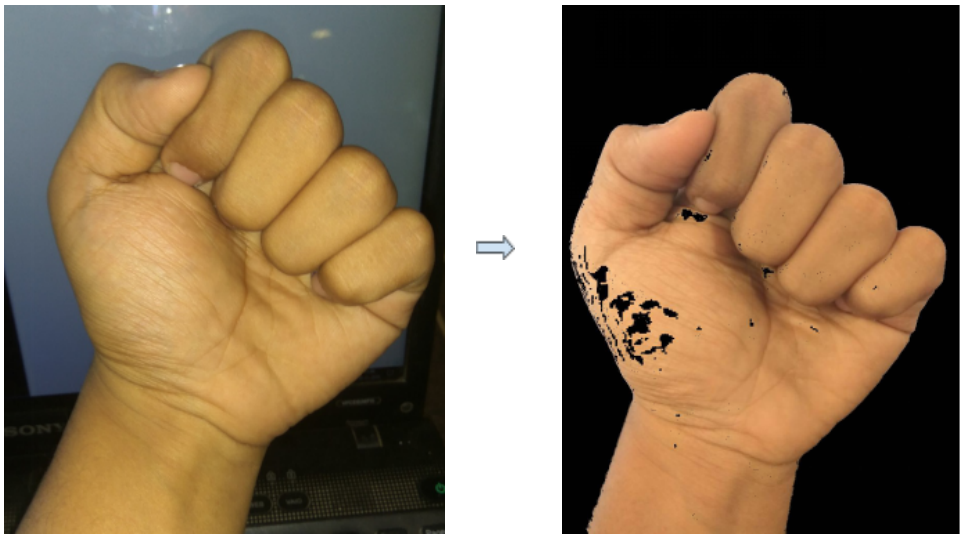


Fig 19

4.1.2.2 Feature Extraction

1. SIFT extracts features vectors (128 dimensional) from image such that they are invariant to any scaling, rotation or translation of the images.

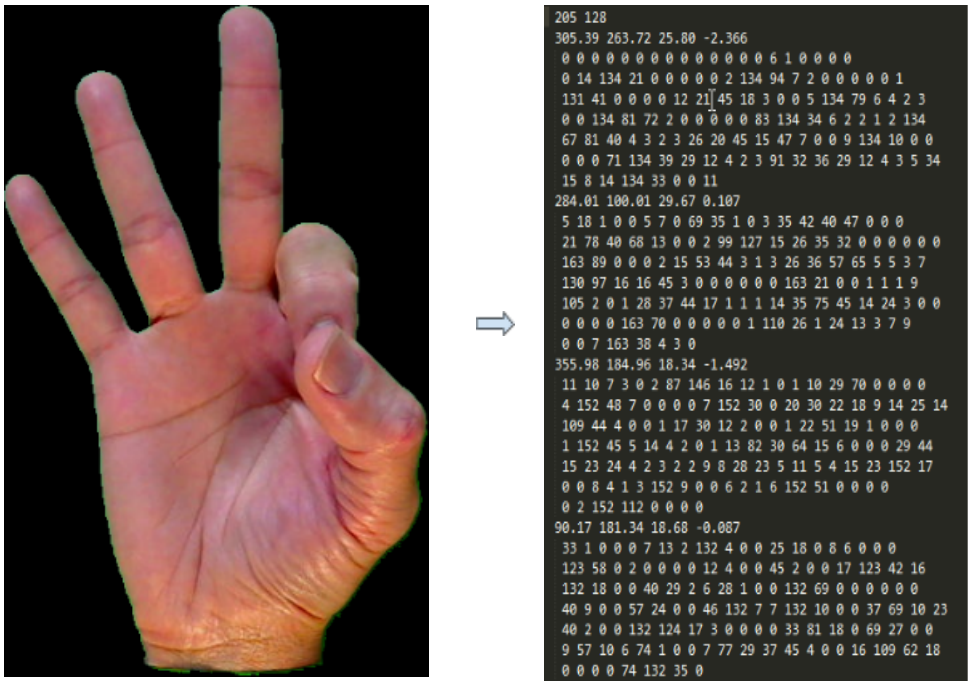


Fig 20

2. K means Clustering of features of all the images is done.

This returns a codebook which contains K centroids of K clusters represented as 128 dimensional vectors.

```
[[ 0.01167863  0.00195208  0.00624475 ...,  0.00681792  0.00374425
   0.00341412]
 [ 0.02729902  0.00406347  0.00081515 ...,  0.00643028  0.00609109
   0.01316141]
 [ 0.03044893  0.05518203  0.1126434  ...,  0.00104061  0.00764782
   0.0232054 ]
 ...,
 [ 0.01610837  0.00126622  0.00042605 ...,  0.01014167  0.00691381
   0.01618478]
 [ 0.02139577  0.01609286  0.03594243 ...,  0.00095354  0.00351789
   0.01098772]
 [ 0.01880423  0.02156151  0.03303165 ...,  0.0066296   0.00827671
   0.01067307]]
```

Fig 21

3. Assign a code word to each image.

```
[548 374 420  31 369 364 225 180  74  35  86 571 464  74 528 134 295 387
 230 145 571 412 571  92 537 410 539 172 412 170 313 101 124 348 346 129
 282  86 251 443 437 581 121 279  42 204  98 162 330 424 302 581 576 279
 104 191 146  76  72 525 254  20 443  49 209 450 166 179  95 100 100  12
 458  79 177 292 280  95 279  30 220  65 200 533 243 143 284 539 319 388
 282 279 279  42  51  98 410 108  12 458 472 249 185 249 185 223 514 185
 444 397 133 390  12 349 514 185 106 524 452 442 352 583 321 333 377 377
 172 260 533 469 507 315 102  25 498  80  41 263 397 397 166 476  26  51
 204 457  49 477 388 145  58 308 114 366 357 440 531  29 391 531  81 501
 266 472 444 249 299 228  30 208 154 249  55 525 396 499 584 102 507 493
 353 501  25 513 114 587 469 439 137 474 500 287  14  61 145 433 293 337
 190  92 390 305 142 193 510]
```

Fig 22

4. Create Histogram corresponding to codeword of Image

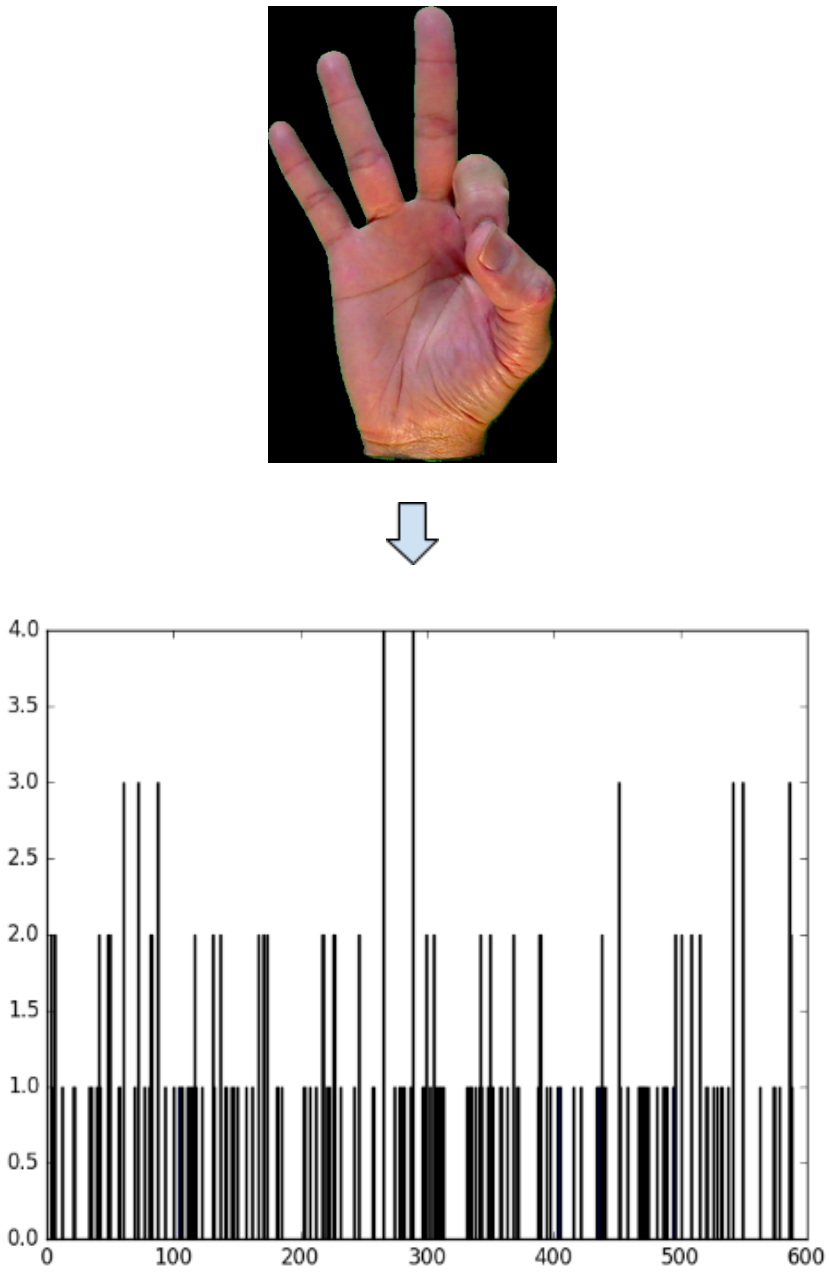


Fig 23

5. Dumping histogram along with its label into a file.

```
0   0:0.000000 1:0.000000 2:0.005495 3:0.000000 4:0.000000 5:0.000000 6:0.005495
7:0.000000 8:0.000000 9:0.000000 10:0.000000 11:0.005495 12:0.000000 13:0.000000
14:0.000000 15:0.000000 16:0.000000 17:0.000000 18:0.000000 19:0.000000 20:0.000000
21:0.005495 22:0.005495 23:0.000000 24:0.000000 25:0.000000 26:0.000000
27:0.000000 28:0.000000 29:0.000000 30:0.000000 31:0.005495 32:0.000000 33:0.000000
34:0.000000 35:0.000000 36:0.000000 37:0.000000 38:0.000000 39:0.000000
40:0.000000 41:0.000000 42:0.005495 43:0.000000 44:0.000000 45:0.005495 46:0.000000
47:0.000000 48:0.000000 49:0.010989 50:0.000000 51:0.000000 52:0.000000
53:0.005495 54:0.000000 55:0.000000 56:0.000000 57:0.005495 58:0.000000 59:0.000000
60:0.000000 61:0.000000 62:0.010989 63:0.000000 64:0.005495 65:0.000000
66:0.000000 67:0.000000 68:0.000000 69:0.000000 70:0.005495 71:0.005495 72:0.000000
73:0.010989 74:0.000000 75:0.000000 76:0.000000 77:0.000000 78:0.000000
79:0.000000 80:0.000000 81:0.000000 82:0.000000 83:0.000000 84:0.000000 85:0.005495
86:0.000000 87:0.010484 88:0.000000 89:0.000000 90:0.000000 91:0.000000
92:0.005495 93:0.000000 94:0.005495 95:0.005495 96:0.000000 97:0.005495 98:0.000000
99:0.000000 100:0.000000

1   0:0.000000 1:0.000000 2:0.000000 3:0.006757 4:0.000000 5:0.006757 6:0.000000
7:0.000000 8:0.000000 9:0.000000 10:0.000000 11:0.000000 12:0.000000 13:0.000000
14:0.000000 15:0.000000 16:0.000000 17:0.000000 18:0.000000 19:0.000000 20:0.000000
21:0.006757 22:0.000000 23:0.000000 24:0.000000 25:0.000000 26:0.000000
27:0.000000 28:0.000000 29:0.000000 30:0.000000 31:0.000000 32:0.000000 33:0.000000
34:0.000000 35:0.000000 36:0.000000 37:0.000000 38:0.000000 39:0.000000
40:0.000000 41:0.000000 42:0.000000 43:0.000000 44:0.000000 45:0.000000 46:0.000000
47:0.006757 48:0.000000 49:0.000000 50:0.000000 51:0.000000 52:0.000000
53:0.006757 54:0.013514 55:0.000000 56:0.000000 57:0.000000 58:0.006757 59:0.000000
60:0.000000 61:0.000000 62:0.000000 63:0.000000 64:0.000000 65:0.000000
66:0.000000 67:0.013514 68:0.000000 69:0.000000 70:0.006757 71:0.000000 72:0.006757
73:0.000000 74:0.000000 75:0.006757 76:0.000000 77:0.000000 78:0.006757
79:0.000000 80:0.000000 81:0.013514 82:0.000000 83:0.000000 84:0.000000 85:0.000000
86:0.000000 87:0.000000 88:0.000000 89:0.000000 90:0.000000 91:0.000000
92:0.006757 93:0.000000 94:0.000000 95:0.006757 96:0.000000 97:0.006757 98:0.000000
99:0.000000 100:0.006757
```

Fig 24

This dump of histograms of all files is used for training the SVM.

4.1.2.3 Training and Testing

1. Finding Optimum Values of SVM parameters c and Gamma using Cross Validation

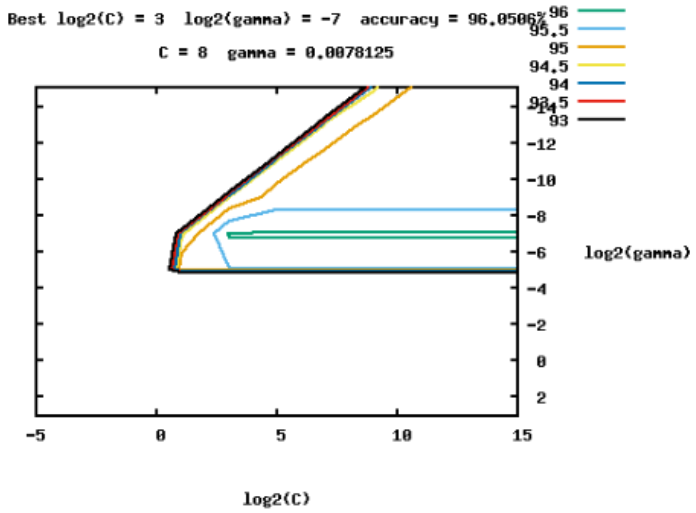


Fig 25

2. Using above c and Gamma values training was performed on 1975 samples and model file was generated. Remaining 540 samples were used for testing.

4.1.3 Results

Symbol	Accuracy
0	33.33
1	13.33
2	13.33
3	86.67
4	80.00
5	73.33
6	13.33
7	26.67
8	40.00
9	93.33
a	73.33
b	60.00
c	53.33
d	60.00
e	46.67
f	80.00
g	46.67
h	53.33

Symbol	Accuracy
i	53.33
j	46.67
k	46.67
l	40.00
m	20.00
n	40.00
o	46.67
p	20.00
q	53.33
r	46.67
s	53.33
t	6.67
u	40.00
v	33.33
w	20.00
x	46.67
y	33.33
z	0.00

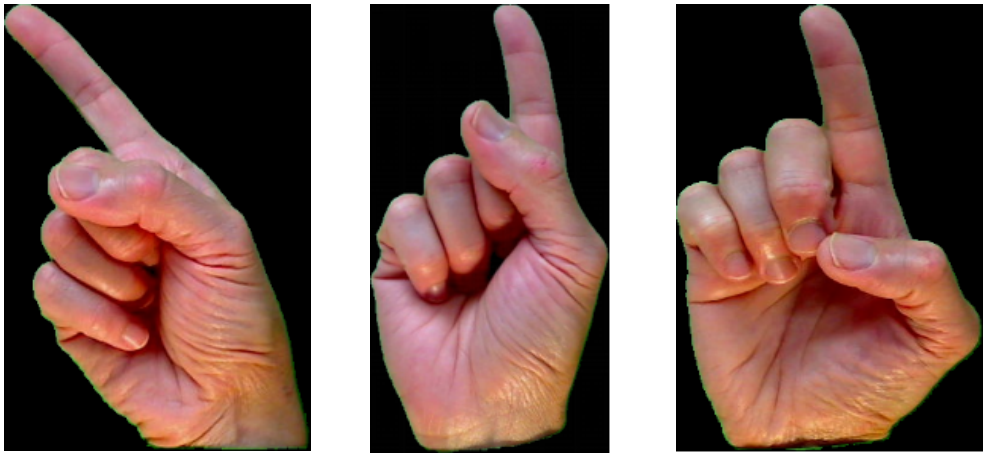
Fig 26

Average accuracy obtained using this approach is 44.259%.

4.1.4 Limitations of this approach

1. No correct classification for the symbol ‘Z’

SIFT extract features irrespective of orientation.Whereas the symbol ‘Z’ differs only in orientation from some other symbols.



z 1 D

Fig 27

2. Some Symbols are too similar to be differentiated using this methodology and SVM doesn’t performs well when target classes are overlapping.

D 1 A M

Fig 28

4.2 SECOND APPROACH
(CONVOLUTIONAL NEURAL NETWORK)

Since very less accuracy (44.259%) was obtained using the last approach.We used CNN.

4.2.1 Data Set Used

The dataset consists of 2524 images with 70 images per category. Each category represented a different character of ASL. This dataset was then augmented to create a dataset of 14781 images. Out of this dataset 75% i.e. 11085 images were used for training and remaining 25% i.e. 3696 images were used for testing.

4.2.2 Image Augmentation and Resize

The images in the data set were of a varying size and shape. Therefore the first step was to read and resize each of the image to the similar size of 224x224 pixel. Only images in the dataset, which are of the same size can be fed into a neural network for training.

To combat this challenge the dataset was augmented to produce several images from each image, thus increasing the size of the dataset and also tackling the problem of overfitting.

Maximum ranges or degrees for **shear, zoom, horizontal and vertical shifting** were specified. Then random values within the maximum range for each of the mentioned parameters were applied on the image, thus generating new images but conserving the class or the category of the image.

Image augmentation not only helped to provide a larger dataset or prevent the classifier from overfitting, but also helped in developing a more robust classifier which could classify much more efficiently even if some changes are brought in owing to the fact that different camera modules might be used and also there might be a shift in the position of the device.

4.2.3 Image Preprocessing

The mean value of RGB over all pixels was subtracted from each pixel value. Subtracting the mean value of the dataset serves to center the data. The reason

mean subtraction is done is that training our model involves multiplying weights and adding biases to the initial inputs in order to cause activations which are then back propagated with the gradients to train the model. It is important that each feature has a similar range, in order to prevent the gradients from getting out of control.Also CNN's involve sharing of parameters and if the inputs are not scaled to have similar ranged values sharing will not happen easily because one part of the image will have large value of weights while the other will end with smaller values.

4.2.4 VGG 16

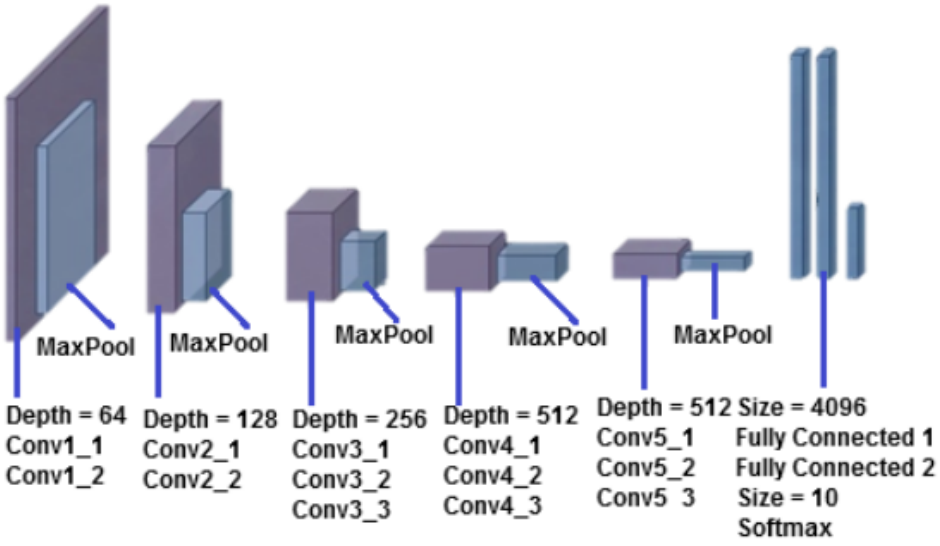


Fig 29: VGG 16 Architecture

VGG 16 as depicted in Figure -- is a deep convolutional neural network model which was proposed by K. Simonyan and A. Zisserman in the paper "Very Deep Convolutional Networks for Large-Scale Image Recognition" . The model was able to achieve 92.7% top-5 test accuracy in ImageNet. The macroarchitecture of VGG 16 can be seen in Fig. .

The input to the ConvNets is a RGB image of size 224 × 224. The image is then passed through a stack of convolutional layers, where filters with a very small receptive field of 3 × 3 are used. The convolution stride is fixed to 1 pixel. The spatial padding of convolution layer input is 1 pixel for 3 × 3 convolutional layers thus preserving the spatial resolution. Spatial pooling has been carried out using five max-pooling layers, which follow some of the convolutional layers but not all the convolutional layers. Max-pooling is performed with a window size of 2 × 2 pixel and stride-size of 2. The stack of convolutional layers is followed by three Fully-Connected layers: the first two

layers have 4096 channels each and the third layer performs 1000-way ILSVRC classification. The final layer is the softmax layer. All hidden layers are equipped with the rectification (ReLU (Krizhevsky et al., 2012)) nonlinearity.

4.2.5 Training

Pre-trained weights which were obtained by training the VGG 16 model on the ImageNet database were used to initial the weights of the model. However, the original model contained 1000 channels signifying the 1000 categories which it aimed to classify. But here only 36 classes are being targeted. Therefore, the final layer was popped and replaced with a fully connected softmax layer with 36 channels to perform the 36 way classifications. The remaining model was retained as it is.The training was carried out using stochastic gradient descent with momentum. Batch size of 128 and momentum of 0.9 was used. The learning rate was initialized with 0.001 and the decay rate was set to10^{-6} .It was observed that in spite of a having a very deep model, the model required very few epochs on the dataset to converge. This is due to the fact that pre-trained were used for weight initialization, which reduced the learning time by a great extent. The pre-trained were obtained on the ImageNet database, but still the model converged with very few epochs despite the fact that the project shares no category with ImageNet. ImageNet can be used because of the large size of the database which contains around 1.2 million images. Thus, the problem of having to converge the Neural Network on a small dataset was tackled .

4.2.6 Results

The samples were tested on VGG 16. The Testing and training accuracies are tabulated below for 4 epochs.

4.2.6.1 Training Result

Epoch	Validation Loss	Validation Accuracy
1	0.7627	71.06
2	0.2387	91.25
3	0.1634	93.96
4	0.1295	94.08

Fig 30

4.2.6.2 Testing Result

The loss obtained on testing set was 0.1182 with an accuracy of 95.50%.

5. CONCLUSION AND FUTURE WORK

Hand gestures are a powerful way for human communication, with lots of potential applications in the area of human computer interaction. Vision-based hand gesture recognition techniques have many proven advantages compared with traditional devices. However, hand gesture recognition is a difficult problem and the current work is only a small contribution towards achieving the results needed in the field of sign language recognition. This report presented a vision-based system able to interpret static hand gestures from the American Sign Language.

Experiments with two different approaches were done.

1. In the first approach SIFT and SVM were used for the sign language classification. SIFT was used to extract scale invariant features while SVM was used for training and testing. The accuracy obtained using this methodology was 44.259%. The low accuracy can be attributed to the fact that SIFT extracts scale invariant features which were beneficial for scaled and rotated images but proved counteractive for some symbols (z,1) as they differ only in orientation. Another reason for low accuracy can be due to the fact that some Symbols are too similar to be differentiated and SVM doesn't performs well when target classes are overlapping.

2. The second approach deals with the application of Convolutional Neural Network for recognizing the hand gestures.The accuracy of the model obtained using Convolutional Neural Network was 95.50%. From this result it is clear that Convolutional Neural Network provides a remarkable accuracy in identifying sign language characters.

We wish to extend our work further in real time with better accuracy. And moreover we have dealt with only alphabets of American Sign Language. We will try to extend it towards recognition of words and sentences.

6. REFERENCES

[1] A.L.C. Barczak, N.H. Reyes, M. Abastillas, A. Piccio and T. Susnjak : A New 2D Static Hand Gesture Colour Image Dataset for ASL Gestures. IIMS, Massey University, Auckland, New Zealand

[2] Neha V. Tavan, Prof. A.V. Deorankar : Indian Sign Language Recognition based on Histograms of Oriented Gradient. http://ijcsit.com/docs/Volume%205/vol5issue03/ijcsit20140503220.pdf

[3] Haitham Hasan · S. Abdul-Kareem : Static hand gesture recognition using neural networks.

[4] Lionel Pigou , Sander Dieleman, PieterJan Kindermans, Benjamin Schrauwen Sign Language Recognition Using Convolutional Neural Networks http://link.springercom/chapter/10.1007/978-3-319-16178-5_40

[5] Bhumika Gupta1, Pushkar Shukla, Ankush Mittal : K-Nearest Correlated Neighbor Classification for Indian Sign Language Gesture Recognition using Feature Fusion

[6] S.Nagarajan, S.Nagarajan : Static Hand Gesture Recognition for Sign Language Alphabets using Edge Oriented Histogram and Multi Class SVM http://research.ijcaonline.org/volume82/number4/pxc3892145.pdf

[7] LeCun, Yann, Léon Bottou, Yoshua Bengio, and Patrick Haffner. "Gradient-based learning applied to document recognition." Proceedings of the IEEE 86, no. 11 (1998): 2278-2324.

[8] Simonyan, Karen, and Andrew Zisserman. "Very deep convolutional networks for large-scale image recognition." arXiv preprint arXiv:1409.1556 (2014).

[9] Rumelhart, David E., Geoffrey E. Hinton, and Ronald J. Williams. "Learning representations by back-propagating errors." Cognitive modeling 5, no. 3 (1988): 1

[10] Hahnloser, Richard HR, Rahul Sarpeshkar, Misha A. Mahowald, Rodney J. Douglas, and H. Sebastian Seung. "Digital selection and analogue amplification coexist in a cortex-inspired silicon circuit." Nature 405, no. 6789 (2000): 947-951.12 Bottou, Léon. "Large-scale machine learning with stochastic gradient descent." In Proceedings of COMPSTAT'2010, pp. 177-186. Physica-Verlag HD, 2010.

[11] Understanding Support Vector Machine Algorithm https://www.analyticsvidhya.com/blog/2015/10/understaing-support-vector-machine-example-code/

[12] Understanding SIFT descriptor and detectors http://www.vlfeat.org/overview/sift.html

[13] https://en.wikipedia.org/wiki/K-means_clustering

[14] http://www.ais.uni-bonn.de/deep_learning/

[15] Automatic Indian Sign Language Recognition for Continuous Video Sequence Joyeeta Singha 1 , Karen Das 2 1 National Institute of Technology, Silchar, Silchar-788010, Assam, INDIA joyeeta_singha90[at]yahoo.com 2 School of Technology, Assam Don Bosco University Airport Road, Azara, Guwahati - 781017, Assam, INDIA karen.das[at]dbuniversity.ac.in

[16] Indian Sign Language Recognition based on Histograms of Oriented Gradient Neha V. Tavari #1 , Prof. A. V. Deorankar #2 1 M. Tech. Scholar Department of Computer Science and Engineering Government College of Engineering, Amravati, Maharashtra, India 2 Associate Professor Department of Computer Science and Engineering Government College of Engineering, Amravati, Maharashtra, India

[17] Hand Gesture Recognition Using Flex Sensors #1 D. K. Barbole, #2 Dr. D. V. Jadhav 1 barbole.dhanshree@gmail.com 2 dvjadhav@gmail.com #12 BSCOER, Narhe, Pune

[18] http://www.lifeprint.com/asl101/topics/wallpaper1.htm