A
Minor Project Report on
"AUTOMATIC SIGN LANGUAGE TRANSLATOR"

In partial fulfillment of requirements for the degree of

Bachelor of Technology (B. Tech.)

in

Computer Science and Engineering

MODY UNIVERSITY OF SCIENCE AND TECHNOLOGY
LAKSHMANGARH · RAJASTHAN
A Leading Women's University

Submitted by

Ms. Tanishka Kapoor (170391)

Under the Guidance of

Dr. Sunil Kumar Jangir

Computer Science Engineering

SCHOOL OF ENGINEERING AND TECHNOLOGY

Mody University and Science and Technology
Lakshmangarh, Distt. Sikar-332311

December 2020

A C K N O W L E D G E M E N T

The completion of any inter-disciplinary project depends upon cooperation, co-ordination and combined efforts of several sources of knowledge. It gives ME immense pleasure to take this opportunity to thank Dr. A. Senthil for giving us such a great opportunity to do our Minor project in the esteemed organization.I owe my profound gratitude to my project Mentor, Dr. Sunil Kumar Jangir, who took keen interest in the project work and guided me along, till the completion of my project work by providing all the necessary information for developing a good system. Finally, no word will be enough to express my deepest reverence to family and friends who have willingly helped me out with their abilities and without whose enthusiasm and support, I wouldn’t have been able to pursue my goals.

Tanishka Kapoor

CERTIFICATE

This is to certify that the minor project report entitled “Automatic Sign Language Translator ” submitted by Ms. Tanishka Kapoor, as a partial fulfillment for the requirement of B. Tech. VII Semester examination of the School of Engineering and Technology, Mody University of Science and Technology, Lakshmangarh for the academic session 2019-2020 is an original project work carried out under the supervision and guidance of Dr. Sunil Kumar Jangir has undergone the requisite duration as prescribed by the institution for the project work.

PROJECT GUIDE:	**HEAD OF DEPARTMENT**
Approval Code: AUT_20_CSE_F12_05	**Signature: Dr. A Senthil**
Name: Dr. Sunil Kumar Jangir	**Name: Dr. A Senthil**
Date: 27.12.2020	**Date: 27.12.2020**

EXAMINER-I:	**EXAMINER-II**
Name: Dr. Praneet Saurabh	**Name: Dr. Ajay Kumar**
Dept: Computer Science	**Dept: Computer Science**

ABSTRACT

Gesture based communication is one of the most seasoned and most characteristic type of language for correspondence, however since the vast majority don't know communication through signing and translators are hard to get a hold of we have thought of a continuous technique utilizing neural organizations for finger spelling based American gesture based communication. In our strategy, the hand is first gone through a channel and after the channel is applied the hand is gone through a classifier which predicts the class of the hand motions. n. It very well may be utilized by an individual who experiences issues in talking or by an individual who can hear yet can not talk and furthermore, by ordinary individuals to speak with hearing impaired individuals. Taking everything into account, approaching a communication via gestures is significant for their social, passionate and phonetic development. We can perceive 20 Italian signals with high exactness. The prescient model can sum up on clients and environmental factors not happening during preparing with a cross-approval exactness of 91.7%.A continuous gesture based communication interpreter is a significant achievement in encouraging correspondence between the hard of hearing network and the overall population. We thus present the turn of events and execution of an American Sign Language (ASL) fingerspelling interpreter dependent on a convolutional neural organization.

Table of Contents

CHAPTER 1- INTRODUCTION

Communication via gestures is a visual language at first utilized by individuals who are aurally weakened by making signals with hand and outward appearances. There are around 300 distinct motions in Practice the world over which have similarity and distinction with one another. English and American sign dialects are considered as various communicated in dialects. A few nations have more than one communication via gestures. Indeed, even in their own nation, ordinary individuals may not be comfortable with the communication via gestures which may restrict the correspondence with the aurally hindered people, particularly in dim it turns out to be even difficult to decipher the signals. A few tasks to interpret the communication via gestures have been created in these years however are ineffectively sent.

American Sign Language (ASL) is regular linguistic structure that has similar etymological homes as being communicating in dialects, having totally unique syntax, ASL can be express with predetermination of activities of the body. Hence, with developing scope of individuals with deafness, there is besides an ascent sought after for interpreters. Gesture based communication interpretation is one of the among most developing line of exploration these days and its miles the greatest normal way of correspondence for the people with hearing impedances. A hand signal acknowledgment contraption can offer an open door for hard of hearing individuals to converse with vocal people without the need of a translator. The proposed framework plans to see some essential components of sign language and to make an interpretation of them to text and sound. American Sign Language is a noticeable language. This task means to create and assess a model that encourages access for the hard of hearing constantly disabled to advanced substance—specifically the instructive substance and learning objects—making the conditions for more noteworthy social incorporation of hard of hearing a lot hindered individuals.

1.1 PRESENT SYSTEM

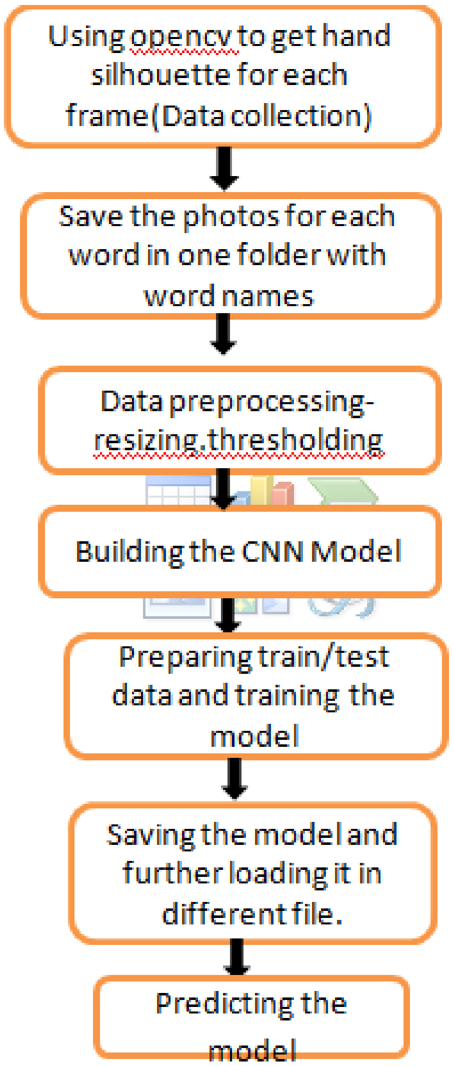
Gesture based communication is a language through which correspondence is possible without the techniques for acoustic sounds. All things being equal, gesture based communication depends on sign examples, i.e., non-verbal communication, direction and developments of the arm to encourage understanding between individuals. There are maybe around 200 communications via gestures being used far and wide today. It has been assessed that there are somewhere in the range of 0.9 and 14 million hearing hindered in India and maybe "one of each five individuals who are hard of hearing on the planet, lives in India", making it the nation with the biggest number of Deaf, and maybe additionally the biggest number of communication via gestures clients. Remembering this, we intend to construct a communication via gestures interpreter that utilizes a glove fitted with sensors that can decipher the 26 English letters, words and sentences in American Sign Language (ASL).design, and position. Diverse glove based methodologies can be used to extricate data .But it is costly and not easy to understand.

1.2 PROPOSED SYSTEM

The proposed framework attempts to do a continuous interpretation of hand signals into identical English content. This framework takes hand motions as contribution through video and interprets it text which could be perceived by a non-signer. There will be utilization of CNN for grouping of hand signals. By conveying this framework, the correspondence hole among signers and non-signers. The proposed framework would be an ongoing framework wherein live sign motions would be handled utilizing picture processing.Then classifiers would be utilized to separate different signs and the deciphered yield would show text . Machine Learning algorithms will be used to train on the data The motivation behind the framework is to improve the current framework here as far as reaction time and precision with the utilization of productive calculations, top notch informational collections and better sensors. Advancements utilized Tensorflow,Keras, OpenCv. we would do the identification by imageprocessing The main advantage of using image processing over. Also by using a threshold value while converting the image from Grayscale to Binary form, Likewise by utilizing an edge esteem while changing the picture from Grayscale over to Binary structure, this framework can be utilized in any foundation and isn't confined to be utilized with Black or White Background.

CHAPTER 2- SYSTEM DESIGN

2.1 SYSTEM FLOWCHART


Using opencv to get hand silhouette for each frame(Data collection)
Save the photos for each word in one folder with word names
Data preprocessing-resizing,thresholding
Building the CNN Model
Preparing train/test data and training the model
Saving the model and further loading it in different file.
Predicting the model

Picture Acquisition

The images are collected through the web camera. This OpenCV video transfer is utilized to catch the whole marking term. The edges are separated from the stream and are handled as grayscale pictures with the element of 50*50. This measurement is steady all through the task as the whole dataset is estimated precisely the equivalent.

Hand Posture Recognition

The preprocessed pictures are taken care of to the keras CNN model. The model that has just been prepared produces the anticipated name. All the motion marks are relegated with a likelihood. The mark with the most noteworthy likelihood is dealt with to be the anticipated mark.

Convolutional Neural Network for Detection

CNN are a class of neural organization that are exceptionally valuable in taking care of PC vision issues. They discovered motivation from the real view of vision that happens in the visual cortex of our cerebrum. They utilize a channel/piece to look over the whole pixel estimations of the picture and make calculations by setting proper loads to empower location of a particular element.

The CNN is outfitted with layers like convolution layer, max pooling layer, straighten layer, thick layer, dropout layer and a completely associated neural organization layer. These layers together make an exceptionally useful asset that can distinguish highlights in a picture. The beginning layers identify low level highlights that step by step start to distinguish more mind boggling more significant level highlights.

The CNN Architecture working

The CNN model for this venture comprises of 11 layers. There are 3 convolutional layers. The first convolutional layer, which is answerable for distinguishing low level highlights like lines, acknowledges a picture with 50*50 size in the grayscale picture. 16 channels of size 2*2 are utilized in this layer which brings about the age of an actuation guide of 49*49 for each channel which implies the yield is equal to 49*49*16. A rectifier direct unit (relu) layer is likewise added to dispose of any negative qualities on the guide and supplant it with 0. A maxpooling layer is applied which decreases the actuation to 25*25 by just thinking about greatest qualities in 2*2 locales of the guide. This progression builds the likelihood of recognizing the ideal element. This is trailed by a second convolutional layer. It is answerable for distinguishing highlights like points and bends. This layer has 32 channels of size 3*3 which brings about the age of an actuation guide of 23*23 which implies the yield is identical to 23*23*32. A maxpooling layer further lessens the actuation guide to 8*8*32 by finding the most extreme qualities in 3*3 locales of the guide. A third convolutional layer is utilized to distinguish elevated level highlights like signals and shapes. 64 channels of size 5*5 diminish the contribution to a yield of 4*4*64. A maxpooling layer decreases the guide to 1*1*64. The guide is leveled to a 1d cluster of length 64. A thick layer grows the guide to a variety of 128 components. A dropout layer exits irregular guide components to lessen overfitting. Eventually, a thick layer diminishes the guide to a variety of 44 components which speak to the quantity of classes.

Calculation Real time gesture based communication change to text and Start

S1: Set the hand histogram to change with the skin composition and the lighting conditions.

S2: Apply information expansion to the dataset to extend it and along these lines diminish the overfitting.

S3: Split the dataset into train, test and approval informational collections.

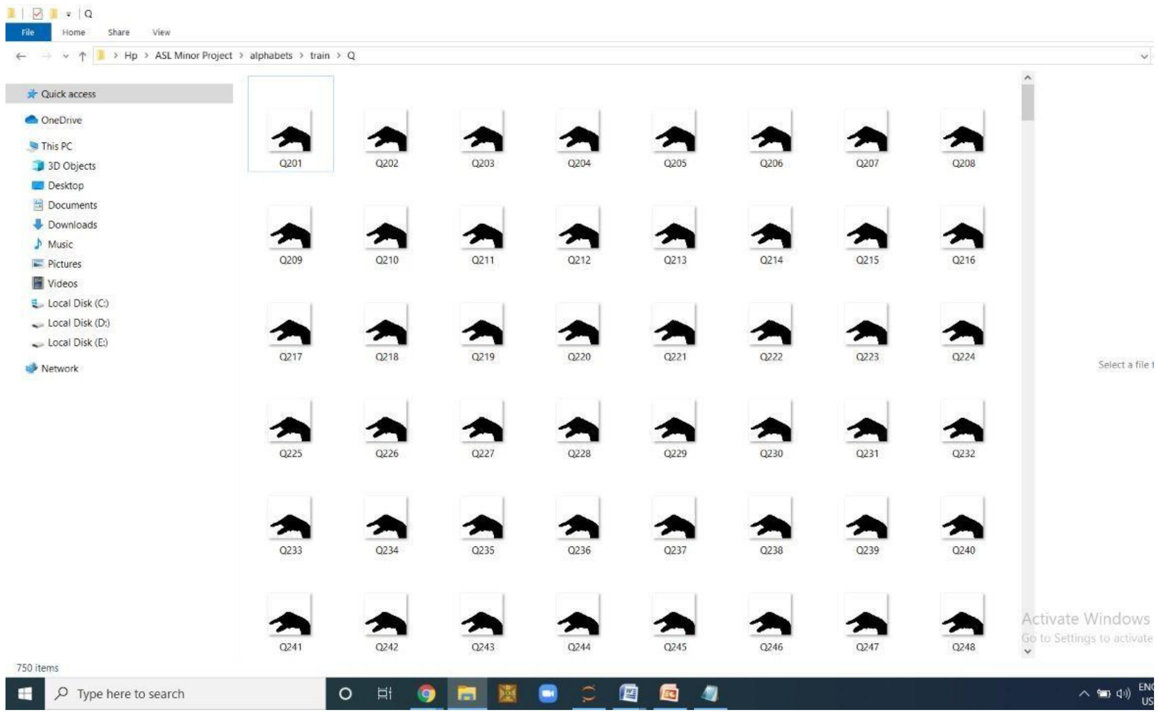
S4: Train the CNN model to fit the dataset.

S5: Generate the model report which incorporates the exactness, mistake and the disarray grid.

S6: Execute the forecast record this document predicts singular motions, cumulates them into words, shows the words as text, transfers the voice yield.

- OpenCV-Its 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 etc.
- The methodology is vision based approach.All the signs are represented with bare hands and so it eliminates the problem of using any artificial devices for interaction.
- Data set generation- There were already made datasets but they were colored and I thought of giving them input in the form of black and white image.Firstly I captured 99 images per alphabet for training data, and for testing around 70 .
- We capture each frame shown by the webcam of the laptop.In each frame ROI was defined denoted by a blue bounded square as shown in the image below.From this the image was converted into grayscale and stored in folders.
- During training the processed image data was split into training, validation, and testing data and written to storage. Training also involves a load dataset.py script that loads the relevant data split into a Dataset class.
- The model used in this classification task is a fairly basic implementation of a Convolutional Neural Network (CNN). As the project requires classification of images, a CNN is the go-to architecture This model consisted of convolutional blocks containing two 2D Convolutional Layers with ReLU activation, followed by Max Pooling and Dropout layers.
- To determine whether our preprocessing of images actually results in a more robust model, we verified on a test set comprised of images from the original dataset, and our own collected image data

2.2 DATASET



There are total 30,000 images ,30 gestures (26 alphabets + 4 special characters) .

Each gesture has 750 images in the training dataset and 250 images in test set per gesture.

CHAPTER 3- HARDWARE AND SOFTWARE DETAILS

- **Pycharm Community Edition**

  The community edition of PyCharm is Apache 2 licensed: which means it is free and open source and you can go to GitHub, and take a gander at the source code. You're allowed to utilize it at whatever point, and any place you like, including at work. Moreover, you can fork and adjust it. See the python subfolder README.md for insights regarding PyCharm instead of IntelliJ IDEA.

- **Opencv-Python**

  OpenCV is an enormous open-source library for PC vision, AI, and picture preparing. OpenCV upholds a wide assortment of programming dialects like Python, C++, Java, and so on It can handle pictures and recordings to recognize items, faces, or even the penmanship of a human. At the point when it is incorporated with different libraries, for example, Numpy which is a profoundly enhanced library for mathematical activities, at that point the quantity of weapons increments in your Arsenal i.e whatever tasks one can do in Numpy can be joined with OpenCV.

- **TENSORFLOW**

  TensorFlow is an open source programming library for numerical estimation using dataflow charts. Centers in the outline addresses mathematical exercises, while graph edges address multi-dimensional data bunches (also called tensors) granted between them. The versatile designing grants you to send estimation to in any event one CPUs or GPUs in a work region, laborer, or wireless with a single API." Also plan of TensorFlow models is as of now maintained which makes it more straightforward to use for present day purposes, giving a fight to business libraries, for instance, Deeplearning4j, H2O and Turi.

- **NUMPY**

  NumPy is one of the center libraries in Python programming and offers help for exhibits. A picture is basically a standard NumPy exhibit containing pixels of information focuses. Thusly, by utilizing essential NumPy tasks, for example, cutting, covering, and extravagant ordering, you can change the pixel estimations of a picture.

- **KERAS**

  Keras is a focal piece of the firmly associated TensorFlow 2.0 environment, covering each progression of the AI work process, from information the board to hyperparameter preparing to arrangement arrangements. Since Keras makes it simpler to run new analyses, it engages you to attempt a bigger number of thoughts than your opposition, quicker. Based on top of TensorFlow 2.0, Keras is an industry-strength structure that can scale to enormous groups of GPUs or a whole TPU unit. It's not just conceivable; it's simple

CHAPTER 4 - IMPLEMENTATION OF WORK DETAILS

4.1 Real Life Applications of Automatic Sign Language

1. On the Job. You're working in client support and need to speak with a client who is hard of hearing. You could compose notes to one another, in any case, ASL is a quicker, simpler, and more straightforward approach to impart.

2. Private Conversation. You are at a café sitting in a private corner. You'd prefer to have a private discussion with the individual opposite you. With ASL, your discussions can't be caught and are just clear by the individuals who know ASL and are in the view.

3. Radio Station. You are in a radio broadcast sound stay with a few people taking an interest in a live syndicated program. With ASL you can speak with individuals during a live account without talking and intruding on the show.

4. Recording Studio. You're in a chronicle studio. On the opposite side of the sound-confirmation glass is the account corner with the sound specialist. With ASL, you can without much of a stretch discuss to and fro through the glass even while recording.

4.2 Data Implementation and program execution

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

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

We used Open computer vision(OpenCV) library in order to produce our dataset.Firstly we captured around 800 images of each of the symbol in ASL for training purposes and around 200 images per symbol for testing purpose. First we capture each frame shown by the webcam of our machine. In the each frame we define a region of interest (ROI) which is denoted by a blue bounded square as shown in the image below.From this whole image we extract our ROI which is RGB and convert it into gray scale Image as shown below.
From this whole image we extract our ROI which is RGB and convert it into
gray scale Image as shown below.

CNN Model :

1.It is 1st processed within the 1st convolutional layer mistreatment thirty two filter weights (3x3 pixels each). this can lead to a 126X126 element image, one for every Filter-weights.
2. **First Pooling Layer** : the photographs square measure downsampled mistreatment easy lay pooling of 2x2 i.e we have a tendency to keep the best worth within the 2x2 sq. of array. Therefore, our image is downsampled to 63x63 pixels.
3. **Second Convolution Layer** : Now, these sixty three x sixty three from the output of the primary pooling layer is served as Associate in Nursing input to the second convolutional layer.It is processed within the second convolutional layer mistreatment thirty two filter weights (3x3 pixels each).This will lead to a sixty x sixty element image.
4. **Second Pooling Layer** : The ensuing pictures square measure downsampled once more mistreatment easy lay pool of 2x2 and is reduced to thirty x thirty resolution of pictures.
5. **First Densely Connected Layer** : Now these pictures square measure used as Associate in Nursing input to a totally connected layer with 128 neurons and therefore the output from the second convolutional layer is reshaped to Associate in Nursing array of 30x30x32 =28800 values. The input to the present layer is Associate in Nursing array of 28800 values. The output of those layer is fed to the second Densely Connected Layer.We square measure employing a dropout layer valuable zero. to avoid overfitting.
6. **Second Densely Connected Layer** : Now the output from the first Densely Connected Layer square measure used as Associate in Nursing input to a totally connected layer with ninety six neurons.
7. **Final layer:** The output of the second Densely Connected Layer is Associate in Nursing input for the ultimate layer which can have variety|the amount|the quantity} of neurons because the number of categories we have a tendency to square measure classifying (alphabets + blank symbol).
Activation perform : We have used ReLu (Rectified Linear Unit) in every of the layers(convolutional similarly as totally connected neurons). ReLu calculates max(x,0) for every input element. This adds nonlinearity to the formula and helps to be told additional complicate options.It helps in removing the vanishing gradient downside and rushing up the coaching by reducing the computation time.
Pooling Layer : We apply easy lay pooling to the input image with a pool size of (2, 2) with relu activation perform.This reduces the quantity of parameters therefore modification the computation price and reduces overfitting.
Dropout Layers: The problem of overfitting, wherever once coaching, the weights of the network are therefore tuned to the coaching examples {they square measure|they're} providing the network doesn't perform well once given new examples.This layer "drops out" a random set

of activations in this layer by setting them to zero.The network ought to be able to offer the proper classification or output for a selected example even if a number of the activations square measure born out.

Optimizer : We have used Adam optimizer for change the model in response to the output of the loss perform. Adam combines the benefits of 2 extensions of 2 random gradient descent algorithms particularly accommodative gradient algorithm(ADA GRAD) and root mean sq. propagation(RMSProp)

CHAPTER 5-SOURCE CODE

- **COLLECTDATA.PY FILE**

```
import cv2
import numpy as np
import os

# Train or test
#mode = 'train'
#directory = 'data/'+mode+'/'



import matplotlib.pyplot as plt
from IPython.display import clear_output

main_foldername=input("DATASET FOLDER NAME : ")
train_foldername=input("TRAINING FOLDER NAME : ")
test_foldername=input("TESTING FOLDER NAME : ")
```

O/P

```
DATASET FOLDER NAME : alphabets
TRAINING FOLDER NAME : train
TESTING FOLDER NAME : test
```

```
gesture_foldername=input("GESTURE NAME : ")

path1=os.path.join(main_foldername,train_foldername,gesture_foldername)
path2=os.path.join(main_foldername,test_foldername,gesture_foldername)

if not os.path.exists(path1):
    os.makedirs(path1)
if not os.path.exists(path2):
    os.makedirs(path2)
```

```
cap = cv2.VideoCapture(0)
count=1
while True:
    _, frame = cap.read()
    # Simulating mirror image
    frame = cv2.flip(frame, 1)

    # Getting count of existing images
    # Printing the count in each set to the screen
    # Coordinates of the ROI
    x1 = int(0.5*frame.shape[1])
    y1 = 10
    x2 = frame.shape[1]-10
    y2 = int(0.5*frame.shape[1])
    # Drawing the ROI
    # The increment/decrement by 1 is to compensate for the bounding box
    cv2.rectangle(frame, (x1-1, y1-1), (x2+1, y2+1), (255,0,0) ,1)
    # Extracting the ROI
    roi = frame[y1:y2, x1:x2]
    roi = cv2.resize(roi, (64, 64))
    cv2.imshow("Frame", frame)
    #cv2.show()
    roi = cv2.cvtColor(roi, cv2.COLOR_BGR2GRAY)
    _, roi = cv2.threshold(roi, 120, 255, cv2.THRESH_BINARY)
    #if count<=50:
    ##    count+=1
    if count<=200:
        count+=1
    elif count<=950:
        cv2.imwrite(path1+'\\'+gesture_foldername+str(count)+'.png',roi)
        count+=1
    elif count<=1200:
        cv2.imwrite(path2+'\\'+gesture_foldername+str(count)+'.png',roi)
        count+=1
    elif count>1200:
```

```
    break
    interrupt = cv2.waitKey(10)
    if interrupt & 0xFF == 27: # esc key
    break

cap.release()
cv2.destroyAllWindows()
```

O/P

```
GESTURE NAME : Z2BLANK
```

- **TRAIN_DATA.PY FILE**

```
# Importing the Keras libraries and packages
from keras.models import Sequential
from keras.layers import Convolution2D
from keras.layers import MaxPooling2D
from keras.layers import Flatten
from keras.layers import Dense

# Step 1 - Building the CNN

# Initializing the CNN
classifier = Sequential()

# First convolution layer and pooling
classifier.add(Convolution2D(32, (3, 3), input_shape=(64, 64, 1),
activation='relu'))
classifier.add(MaxPooling2D(pool_size=(2, 2)))
# Second convolution layer and pooling
classifier.add(Convolution2D(32, (3, 3), activation='relu'))
# input_shape is going to be the pooled feature maps from the previous
convolution layer
```

```
classifier.add(MaxPooling2D(pool_size=(2, 2)))

# Flattening the layers
classifier.add(Flatten())

# Adding a fully connected layer
classifier.add(Dense(units=128, activation='relu'))
#classifier.add(Dense(units=128, activation='relu'))
classifier.add(Dense(units=30, activation='softmax')) # softmax for more
than 2

# Compiling the CNN
classifier.compile(optimizer='adam',           loss='categorical_crossentropy',
metrics=['accuracy'])
# categorical_crossentropy for more than 2

classifier.fit_generator(
        training_set,
        steps_per_epoch=22500, # No of images in training set
        epochs=10,
        validation_data=test_set,
        validation_steps=7500)# No of images in test set


# Saving the model
model_json = classifier.to_json()
with open("model-bw.json", "w") as json_file:
    json_file.write(model_json)
classifier.save_weights('model-bw.h5')
```

- **PREDICT.PY FILE CODE**

```
import numpy as np
from keras.models import model_from_json
import operator
import cv2
import sys, os
import webbrowser
import subprocess

# Loading the model
json_file = open("model-bw.json", "r")
model_json = json_file.read()
json_file.close()
loaded_model = model_from_json(model_json)
# load weights into new model
loaded_model.load_weights("model-bw.h5")
print("Loaded model from disk")

cap = cv2.VideoCapture(0)
#whitebg = cv2.imread('"C:\\Users\\Hp\\Pictures\\AIphotos\\whitebg.jpg"')

# Category dictionary
#categories = {0: 'ZERO', 1: 'ONE', 2: 'TWO', 3: 'THREE', 4: 'FOUR', 5:
'FIVE'}
categories = {0: 'A', 1: 'B', 2: 'C',3: 'D', 4: 'E', 5: 'F',6: 'G', 7: 'H', 8: 'I',
            9: 'J', 10: 'K', 11: 'L',12: 'M', 13: 'N', 14: 'O',15: 'P', 16: 'Q',
            17: 'R', 18: 'S',19: 'T', 20: 'U', 21: 'V',22: 'W', 23: 'X', 24: 'Y',25: 'Z',
            26: 'SPACE', 27: 'REMOVE 1', 28: 'REMOVE ALL', 29:
'BLANK'}
s=""
d={}
p=""
count=0

while True:

    count+=1
```

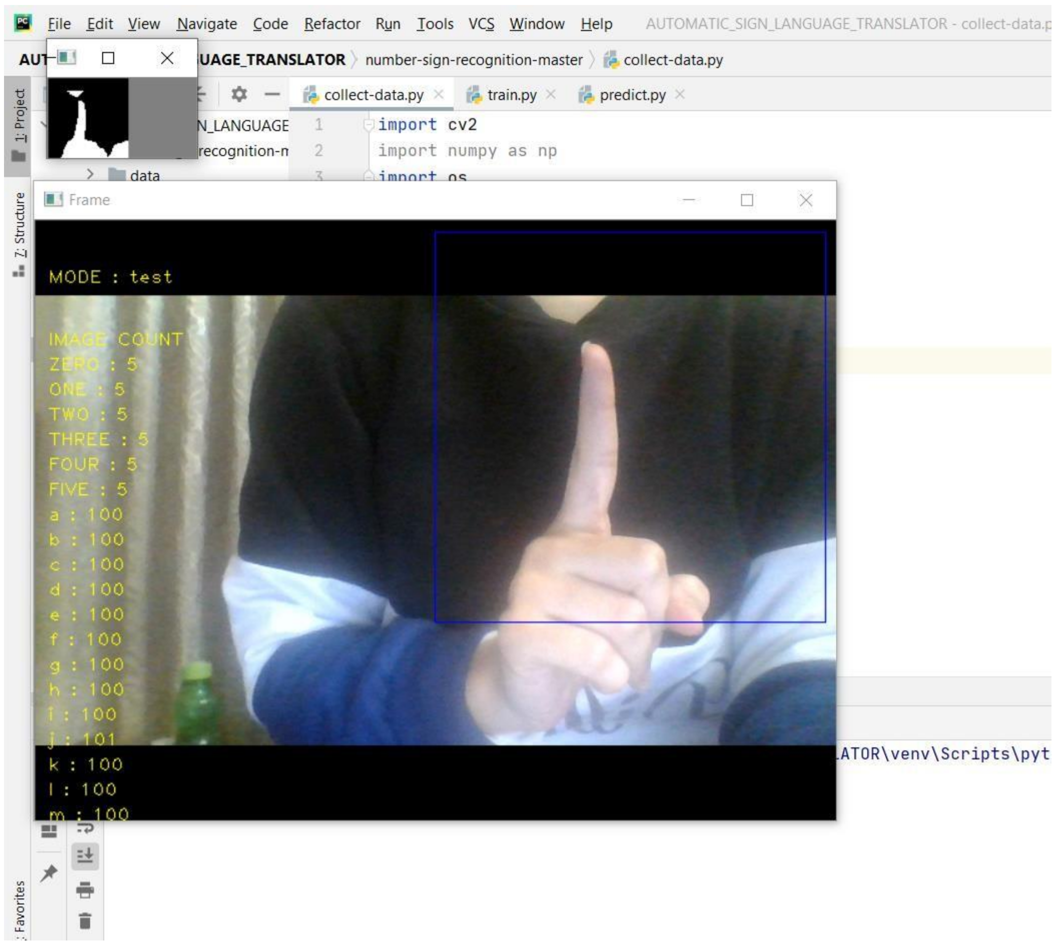
```
_, frame = cap.read()
# Simulating mirror image
frame = cv2.flip(frame, 1)

# Got this from collect-data.py
# Coordinates of the ROI
x1 = int(0.5*frame.shape[1])
y1 = 10
x2 = frame.shape[1]-10
y2 = int(0.5*frame.shape[1])
# Drawing the ROI
# The increment/decrement by 1 is to compensate for the bounding box
cv2.rectangle(frame, (x1-1, y1-1), (x2+1, y2+1), (255,0,0) ,1)
# Extracting the ROI
roi = frame[y1:y2, x1:x2]

# Resizing the ROI so it can be fed to the model for prediction
roi = cv2.resize(roi, (64, 64))
roi = cv2.cvtColor(roi, cv2.COLOR_BGR2GRAY)
_, test_image = cv2.threshold(roi, 120, 255, cv2.THRESH_BINARY)
cv2.imshow("test", test_image)
# Batch of 1
result = loaded_model.predict(test_image.reshape(1, 64, 64, 1))
prediction = {'A': result[0][0],
              'B': result[0][1],
              'C': result[0][2],
              'D': result[0][3],
              'E': result[0][4],
              'F': result[0][5],
              'G': result[0][6],
              'H': result[0][7],
              'I': result[0][8],
              'J': result[0][9],
              'K': result[0][10],
              'L': result[0][11],
              'M': result[0][12],
              'N': result[0][13],
              'O': result[0][14],
              'P': result[0][15],
              'Q': result[0][16],
              'R': result[0][17],
              'S': result[0][18],
              'T': result[0][19],
              'U': result[0][20],
              'V': result[0][21],
              'W': result[0][22],
              'X': result[0][23],
              'Y': result[0][24],
              'Z': result[0][25],
              'SPACE': result[0][26],
              'REMOVE 1': result[0][27],
              'REMOVE ALL': result[0][28],
              'BLANK': result[0][29]}


  # Sorting based on top prediction
  prediction = sorted(prediction.items(), key=operator.itemgetter(1),
reverse=True)

  # Displaying the predictions
  cv2.putText(frame,prediction[0][0],(10,120)
cv2.FONT_HERSHEY_PLAIN, 1, (0,255,255), 1)
  #cv2.Waitkey(30)
  #s+=prediction[0][0]
  if count<=500:
    if prediction[0][0] in d:
      d[prediction[0][0]]+=1
    else:
      d[prediction[0][0]]=1
    count+=1
    if count>500:
        inverse = [(value, key) for key, value in d.items()]
        a=max(inverse)[1]
        if a=='SPACE':
            s+=' '
            cv2.putText(frame, 'SPACE PRINTED' , (10, 100),
cv2.FONT_HERSHEY_PLAIN, 1, (0,255,255), 1)
            cv2.waitKey(1000)
        elif a=='REMOVE 1':
            s=s[:-1]
        elif a=='REMOVE ALL':
            s=''
        elif a!='BLANK':
            s+=a
        d={}
        count=0

    cv2.putText(frame, s, (10, 400), cv2.FONT_HERSHEY_PLAIN, 1,
(0,255,255), 1)
    cv2.imshow("Frame", frame)
    interrupt = cv2.waitKey(10)
    if interrupt & 0xFF == 27: # esc key
        break

#print(s)
cap.release()
cv2.destroyAllWindows()
```

O/P

```
Loaded model from disk
```

CHAPTER 6- MODEL’S PHOTOGRAPH/OUTPUT SCREEN

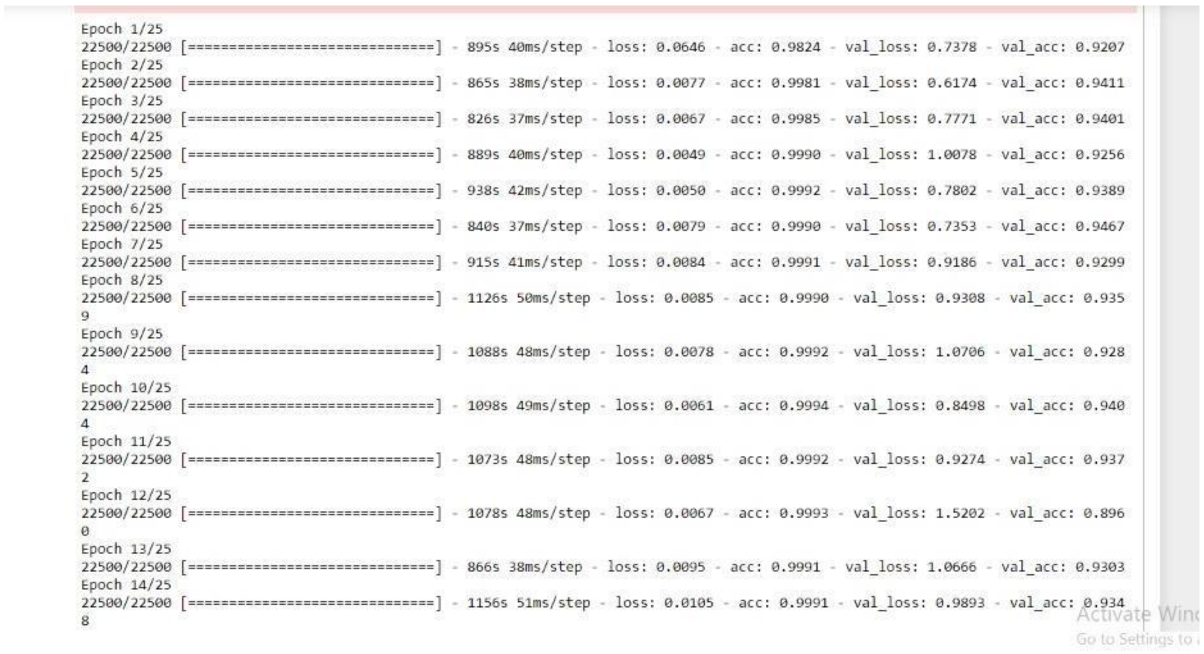
- **DATASET GENERATION**


File Edit View Navigate Code Refactor Run Tools VCS Window Help
AUTOMATIC_SIGN_LANGUAGE_TRANSLATOR - collect-data.p
UAGE_TRANSLATOR 〉 number-sign-recognition-master 〉 collect-data.py
collect-data.py
train.py
predict.py
1: Project
Z: Structure
Favorites
N_LANGUAGE
recognition-n
data
Frame

```
import cv2
import numpy as np
import os
```

```
MODE : test

IMAGE COUNT
ZERO : 5
ONE : 5
TWO : 5
THREE : 5
FOUR : 5
FIVE : 5
a : 100
b : 100
c : 100
d : 100
e : 100
f : 100
g : 100
h : 100
i : 100
j : 101
k : 100
l : 100
m : 100
```

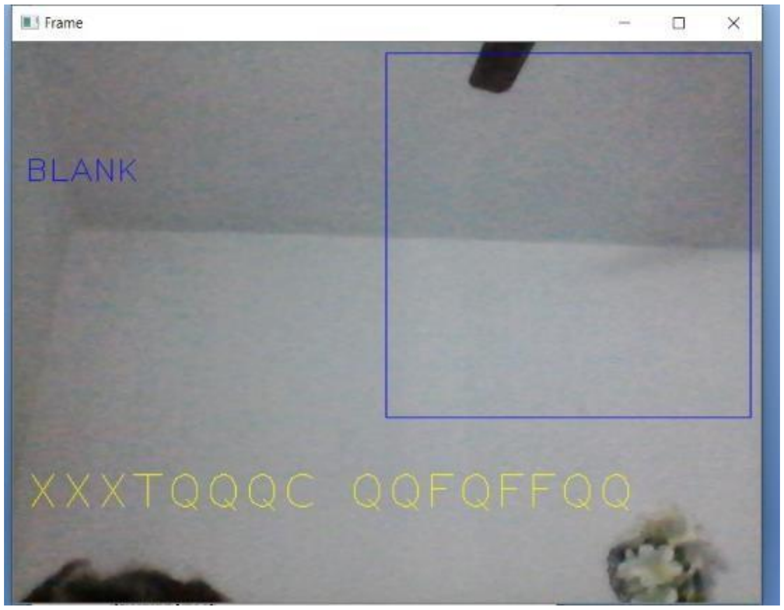
```
ATOR\venv\Scripts\pyt
```

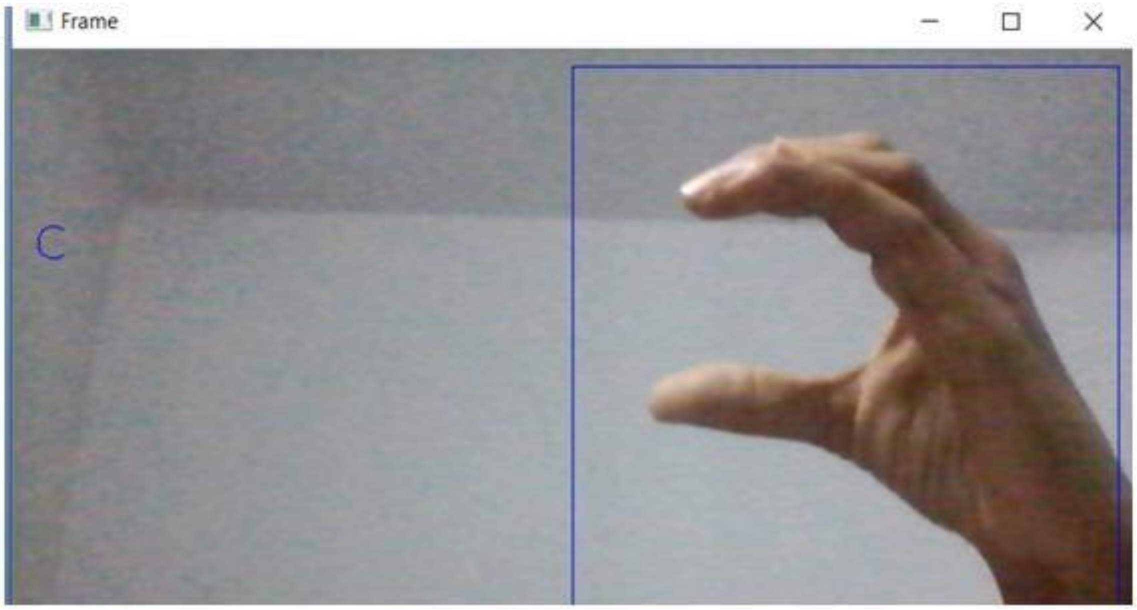


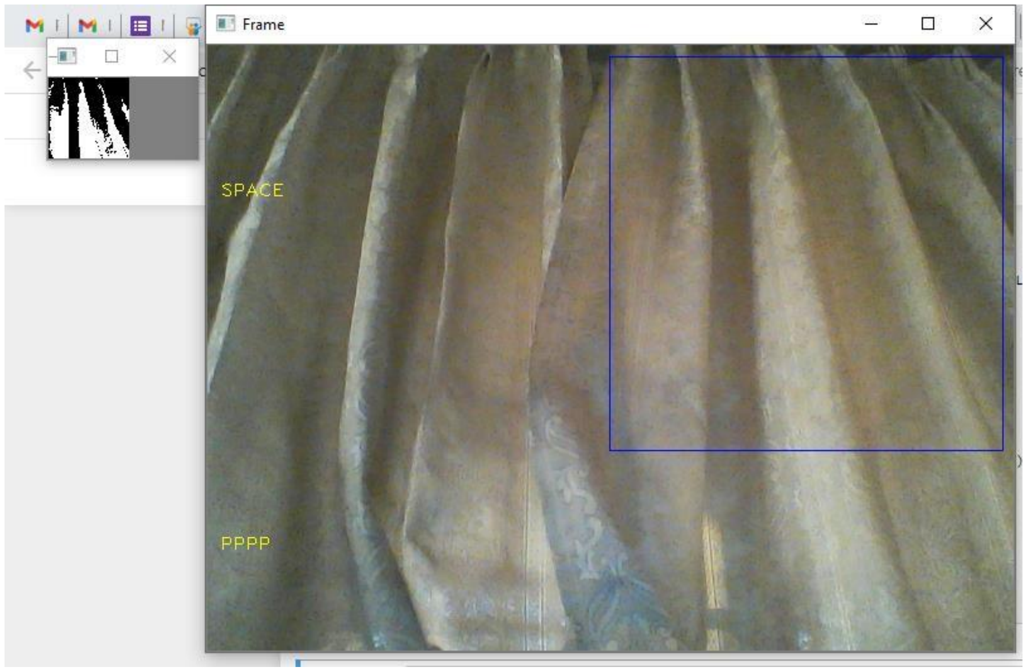
```
Epoch 1/25
22500/22500 [==============================] - 895s 40ms/step - loss: 0.0646 - acc: 0.9824 - val_loss: 0.7378 - val_acc: 0.9207
Epoch 2/25
22500/22500 [==============================] - 865s 38ms/step - loss: 0.0077 - acc: 0.9981 - val_loss: 0.6174 - val_acc: 0.9411
Epoch 3/25
22500/22500 [==============================] - 826s 37ms/step - loss: 0.0067 - acc: 0.9985 - val_loss: 0.7771 - val_acc: 0.9401
Epoch 4/25
22500/22500 [==============================] - 889s 40ms/step - loss: 0.0049 - acc: 0.9990 - val_loss: 1.0078 - val_acc: 0.9256
Epoch 5/25
22500/22500 [==============================] - 938s 42ms/step - loss: 0.0050 - acc: 0.9992 - val_loss: 0.7802 - val_acc: 0.9389
Epoch 6/25
22500/22500 [==============================] - 840s 37ms/step - loss: 0.0079 - acc: 0.9990 - val_loss: 0.7353 - val_acc: 0.9467
Epoch 7/25
22500/22500 [==============================] - 915s 41ms/step - loss: 0.0084 - acc: 0.9991 - val_loss: 0.9186 - val_acc: 0.9299
Epoch 8/25
22500/22500 [==============================] - 1126s 50ms/step - loss: 0.0085 - acc: 0.9990 - val_loss: 0.9308 - val_acc: 0.935
9
Epoch 9/25
22500/22500 [==============================] - 1088s 48ms/step - loss: 0.0078 - acc: 0.9992 - val_loss: 1.0706 - val_acc: 0.928
4
Epoch 10/25
22500/22500 [==============================] - 1098s 49ms/step - loss: 0.0061 - acc: 0.9994 - val_loss: 0.8498 - val_acc: 0.940
4
Epoch 11/25
22500/22500 [==============================] - 1073s 48ms/step - loss: 0.0085 - acc: 0.9992 - val_loss: 0.9274 - val_acc: 0.937
2
Epoch 12/25
22500/22500 [==============================] - 1078s 48ms/step - loss: 0.0067 - acc: 0.9993 - val_loss: 1.5202 - val_acc: 0.896
0
Epoch 13/25
22500/22500 [==============================] - 866s 38ms/step - loss: 0.0095 - acc: 0.9991 - val_loss: 1.0666 - val_acc: 0.9303
Epoch 14/25
22500/22500 [==============================] - 1156s 51ms/step - loss: 0.0105 - acc: 0.9991 - val_loss: 0.9893 - val_acc: 0.934
8
```

TOTAL EPOCHS-25

- **TRAINED MODEL PREDICTION PHOTOGRAPH**




Frame
C


Frame
SPACE
PPPP

CHAPTER 7- SYSTEM TESTING

In American Sign Language (ASL), we utilize the 5 Parameters of ASL to depict how a sign carries on inside the underwriter's space. The boundaries are handshape, palm direction, development, area, and articulation/non-manual signs.

There are a few different ways to "do" the "TEST" sign. Every one of them center around indicating a kind of "question mark." This by and large depends on beginning with a straight forefinger changing into an "x" finger while moving descending as though drawing a question mark noticeable all around and afterward "dabbing it."

An extremely basic form of the sign TEST begins with forefingers straight, at that point twists them into "x" handshapes as the hands move descending. At that point the hands change into palm-down "5" handshapes.

Firstly the application is to be organized or completed. The second stage covers the unit testing utilized for testing the individual functionality, namely(Sign to message, text to sign and add a motion.

For unit tests, from the start the environment will be worked for testing the application. After unit tests entire framework will be tried with connection and mediation of all components. The applicationwill be tried utilizing android cell phone.

CHAPTER 8- CONCLUSION

Gesture based communication acknowledgment is a difficult issue on the off chance that we think about all the potential mixes of signals that an arrangement of this sort needs to comprehend and decipher. That being stated, most likely the most ideal approach to take care of this issue is to separate it into less difficult issues, and the framework introduced here would relate to a potential answer for one of them.

The task is a straightforward exhibition of how CNN can be utilized to tackle PC vision issues with a very serious level of exactness. A finger spelling gesture based communication interpreter is acquired which has an exactness of 95%. The task can be reached out to other communications via gestures by building the relating dataset and preparing the CNN.

8.1 LIMITATIONS

- Gesture based communication interpretation innovations are restricted similarly as communicated in language interpretation. None can decipher with 100% precision.
- There will be fluctuations sometimes even after training a lot of data.
- Indeed, communication via gestures interpretation advances are a long ways behind their communicated in language partners. This is, in no insignificant way, because of the way that marked dialects have numerous articulators. Where communicated in dialects are verbalized through the vocal plot, marked dialects are enunciated through the hands, arms, head, shoulders, middle, and parts of the face. This multi-channel verbalization makes interpreting communications via gestures extremely troublesome.
- An extra test for communication via gestures MT is the way that there is no formal composed arrangement for marked dialects. There are documentations frameworks yet no composing framework has been received generally enough, by the worldwide Deaf people group, that it very well may be considered the 'composed type' of an offered hint language. Gesture based communications at that point are recorded in different video designs.

8.2 FUTURE SCOPE

In future work, proposed framework can be created and executed utilizing Raspberry Pi. Picture Processing part should be improved so System would have the option to convey in the two ways i.e.it should be equipped for changing ordinary language over to gesture based communication and the other way around. We will attempt to perceive signs which incorporate movement. In addition we will zero in on changing over the arrangement of motions into text for example word and sentences and afterward changing over it into the discourse which can be heard.

- In future, with the use of sensor based technology in ISL some words which uses wrist movement can also be detected so that those signs can be expressed by glove. Similarly elbow movement and facial recognition can be the two areas. Concept of video conferencing with the deaf and dumb people can be introduced for Indian Sign Language for advanced communication.

- Communication between the deaf and dumb people in Indian sign language without using sensor based technology can also be done using Smartphone platform which will provide easy to use environment, mobility and high growth rate.

CHAPTER 9- BIBILOGRAPHY

- G.K. Kharate, A.S. Ghotkar," Vision based multi-feature hand gesture recognition for indian sign language manual signs", *Int. J. Smart Sens. Intell. Syst., 9 (1) (2016)*
- S.B. Patil, G. Sinha, "Distinctive feature extraction for indian sign language (ISL) gesture using scale invariant feature transform (SIFT)", *J. Inst. Eng. (India): Series B, 98 (1) (2017), pp. 19-26.*
- Suharjitoa*, Ricky Andersonb , Fanny Wiryanab , Meita Chandra Ariestab , Gede Putra Kusuma, "Sign Language Recognition Application Systems for Deaf-Mute People: A Review Based on Input-Process-Output", *Procedia Computer Science 116 (2017)*
- Rajesh George Rajan, M Judith Leo, "A Comprehensive Analysis on Sign Language Recognition System", *ISSN: 2277-3878, Volume-7, Issue-6, March 2019*
- Beena M.V. Asst. professor, CSE Dept. ,»Beena M.V. Asst. professor, CSE Dept»., *Volume 117 No. 20 2017, 9-15*
- Lean Karlo Tolentino,Ronnie serfa juan,August thio-ac, "Static Sign Language Recognition Using Deep Learning", *December 2019*
- Rawan A. AI Rashid Agha, Polla Fattah, Muhammed N.Sefer , "A comprehensive study on sign languages recognition systems using (SVM, KNN, CNN and ANN*)", Article No.: 28 Pages 1–6-October 2018*
- Ridley College, St. Catharines," Research of a Sign Language Translation System Based on Deep Learning", *2019 International Conference on Artificial Intelligence and Advanced Manufacturing (AIAM) IEEE Explore 09 January 2020*
- Anjan Kumar Talukdar,Kandarpa kumar sharma, "CNN-Based Real-Time Indian Sign Language Recognition System", *Advances in Computational Intelligence and Informatics (pp.71-79) April-2020*
- Ankit Ojha, Ayush Pandey," Sign Language to Text and Speech Translation in Real Time Using Convolutional Neural Network", *NCAIT – 2020 (Volume 8 – Issue 15)August 2020*