VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

TITLE: SIGN LANGUAGE RECOGNITION FOR DEAF AND DUMB PEOPLE USING CNN

REVIEW - 3

ITE2001(Artificial Intelligence)

SUBMITTED TO – PROF. AJIT KUMAR SANTRA

GROUP MEMBERS:

RESHABH KUMAR JAIN - 17BIT0048

SIDHARTH RAJ MIGLANI – 17BIT0145

ARADHYA MATHUR – 17BIT0146

SLOT – A2 + TA2

Table of Contents

1.0 ABSTRACT

Sign Languages are a set of languages that use predefined actions and movements to convey a message. These languages are primarily developed to aid deaf and other verbally challenged people. They use a simultaneous and precise combination of movement of hands, orientation of hands, hand shapes etc.

In this project, we aim towards analysing and recognizing various alphabets from a database of sign images. Database will consist of various images with each image clicked in different light condition with different hand orientation. With such a divergent data set, we will be able to train our system to good levels and thus obtain good results.

2.0 OBJECTIVE

The aim of this project is as follow:

- Aiding deaf and other verbally challenged people

3.0 INTRODUCTION

Hand gesture recognition provides an intelligent, natural, and convenient way of human–computer interaction (HCI). Sign language recognition (SLR) and gesture-based control are two major applications for hand gesture recognition technologies. SLR aims to interpret sign languages automatically by a computer in order to help the deaf communicate with hearing society conveniently. Since sign language is a kind of highly structured and largely symbolic human gesture set, SLR also serves as a good basic for the development of general gesture-based HCI.

3.1 LITERATURE REVIEW

[1] American Sign Language Recognition System: An Optimal Approach

This paper basically focuses on converting the 24 sign language alphabets and numbers into English form. There are many phases in the paper. The first phase deals with pre-processing operations on the sign images. HSV color model is used to mark the skin pixels in the detected skin portion in sign image. For further optimization YCbCr color space is used to construct a skin color model. In the next phase various region properties of the pre-processed image is computed. In the last phase the image is converted to text along with the roundness value calculated for a gesture and number of peaks found. In the end this paper presents a statistical result evaluation with a comparison between the proposed and existing technique.

[2] Hand Gesture Recognition with 3D Convolutional Neural Networks
In this paper an algorithm for hand gesture recognition using 3D convolutional neural networks is proposed. This paper uses the VIVA challenge hand gesture dataset. Each channel is normalized to zero mean and unit variance. The CNN used consists of 2 sub networks, a high-resolution network and low-resolution network. Optimization was performed by stochastic gradient descent. Also, to avoid overfitting of data, spatial-temporal data augmentation and various regularization methods were used. In the end performance was evaluated using leave-one-subject-cross-validation.

[3] Sign Language Recognition
In this paper, web camera is used for capturing the signs from user which will be converted to text and audio. First user has to store their signs into the system then he can use those signs for communication. Here we are using image and video processing for converting text message into speech by using Open cv library. First skin is detected then hand position is determined and then the sign is compared to the signs stored in the local filesystem. If a match is found, result is displayed.

[4] Hand Gesture Recognition for Sign Language Recognition: A Review
This paper presents different methods of hand gesture and sign language recognition used in the past by various researchers. Vision based approach, Glove based approach and color marker-based approach are some of the techniques discussed in paper. The vision-based approach is further divided into approach based on appearance and 3d model.

[5] Implantation of Hand Gesture Recognition Technique for HCI Using Open CV
This paper introduces a gesture recognition technique using python and OpenCV. The proposed algorithm consists of pre-processing, image segmentation and feature extraction. Camera/webcam is used to take the images of gestures. These inputs are used as inputs to the algorithm. First segmentation is used to find similar areas in image. Then using contours, convex hull and convexity defects number of fingers are calculated.

[6] Gesture Recognition using Open-CV
This paper gives states the importance and gives a definition to gesture recognition i.e. a way for computers to understand human body. It then tells us about some areas of gesture recognition like hand gesture recognition, facial recognition, vision-based recognition. Apart from this an elaborate working scheme of its processes and some extensive applications are also mentioned in this paper giving a general idea about how gesture recognition-based systems work.

[7] Implementation of an Efficient Algorithm for Human Hand Gesture Identification
In this paper we delve deeper into the hand gesture recognition. Detecting a hand gesture in real time is an important task where time and memory are important factors to consider. The system has two parts, one performs gesture recognition from live feed while other takes decisions based on these live gestures based an algorithm comprising of series of steps mentioned in the paper like frame detection, hand detection and gesture recognition.

Gesture recognition is further divided into subtasks comprising of MCP detection, Finger detection, Signal generation etc.

[8] Alphabet Sign Language Image Classification Using Deep Learning
In this paper classification of RGB images of static hand signs is discussed. This classification is done using the Densely Connected Convolutional Neural Network. DenseNet is a commonly used classification model due to advantages like alleviating the vanishing gradient in deep networks. It was also able to predict rates with frequencies ranging from 50 to 100Hz and hence is capable of real time prediction.

[9] American Sign Language Recognition using Deep Learning and Computer Vision
In this paper we understand the need of a sign language recognition system. Speech impairment affects a person's ability to speak or hear and hence such people require specialized means of communication such as sign language. This creates great difficulty for non-sign language speakers to understand the sign language users. The focus of this work was to hence create a software that helps in bridging this communication gap. The model proposed first extracts necessary spatial and temporal from video sequences. It then uses inception (type of CNN) to recognize the spatial features and RNN (Recurrent Neural Network) to train on temporal features.

[10] Deep Convolutional Neural Networks for Sign Language Recognition
Through this paper we see that the greatest difficulty in sign language recognition is the extraction of complex hand and head movements that are changing constantly. A mobile app for recognition working on the principle of Convolutional Neural networks is proposed and its detail workings discussed. To verify that CNN gives us the best possible result, a comparative analysis of various classifiers algorithms like MDC, Ad boost, ANN and deep ANN are done with the proposed CNN architecture and it is concluded that indeed CNN Is the best way to simulate sign language recognition.

[11] Real Time Detection And Recognition of Indian And American Sign Language using sift
This paper presented a review on Feature Extraction for Indian and American Sign Language. It presented the recent research and development of sign language based on manual communication and body language. Sign language recognition system typically elaborate three steps pre-processing, feature extraction and classification. Classification methods used for recognition are Neural Network (NN), Support Vector Machine (SVM), Hidden Markov Models (HMM), Scale Invariant Feature Transform (SIFT), etc.

[12] Moment based sign language recognition for Indian Languages
In this paper the sign and hand gestures are captured and processed with the help of MATLAB and the converted into speech & text form. The feature extraction of values of the images is evaluated based on 7Hu (7 moments) invariant moment technique and the classification techniques are applied using K-Nearest Neighbor (KNN) is compared with the PNN (Probabilistic Neural Network) for the accuracy rate. The performance of proposed

classifier KNN is decided based on various parameters. Parameters can be calculated by formulas, using 7hu moments technique's for feature extraction will be done, the 7Hu moments are a vector of algebraic invariants that added regular moment. They are invariant under change of translation, size and rotation. Hu moments have been widely used in classification. And KNN (k-nearest neighbor's classification) algorithm will classifies the objects on basis of majority votes of its neighbor's. The object is assigned to the class by most common among its nearest neighbor's k (k is a small positive integer).

[13] Vision-based sign language translation device

In this paper, a vision-based sign language recognition system using LABVIEW for automatic sign language has been presented. The goal of this project is to develop a new vision-based technology for recognizing and translating continuous sign language to text. Although the deaf, hard of hearing and hearing signers can communicate without problems among themselves, there is a serious challenge for deaf community trying to integrate into educational, social and work environments.

[14] Evaluation of Features for Automated Transcription of Dual- Handed Sign Language Alphabets

The SVM is widely known pattern recognition supervised learning technique. Originally, SVM was developed at bell laboratories for solving binary decision problem. The basic SVM takes input data and predict that which two possible classes generate output. So, it is also known as a non-probabilistic binary linear classifier. Multi-class problems are divided into many two-class problems that can be addressed using several SVMs. One against all, one against one, decision directed graphic approach, etc. are some methods to solve multi-class problem using SVMs.

[15] Sign Language Recognition System

This paper is about to design a system that can understand the sign language and gestures accurately so that the deaf and dumb people may communicate with the outside world easily without the need of an interpreter to help them. Here the work is done for the American sign language (ASL) which has its own unique gesture to represents the words and the English alphabets i.e. is from A-Z. The process is to recognize the sign language and gesture as shown in the below figure.

[16] Deaf Mute Communication Interpreter- A Review

This paper aims to cover the various prevailing methods of deaf-mute communication interpreter system. The two-broad classification of the communication methodologies used by the deaf –mute people are - Wearable Communication Device and Online Learning System. Under Wearable communication method, there are Glove based system, Keypad method and Handicom Touch-screen. All the above mentioned three sub-divided methods make use of various sensors, accelerometer, a suitable micro-controller, a text to speech conversion module, a keypad and a touch-screen. The need for an external device to interpret the message between a deaf –mute and non-deaf-mute people can be overcome

by the second method i.e online learning system. The Online Learning System has different methods. The five subdivided methods are- SLIM module, TESSA, Wi-See Technology, SWI_PELE System and Web-Sign Technology.

[17] An Efficient Framework for Indian Sign Language Recognition Using Wavelet Transform

The proposed ISLR system is considered as a pattern recognition technique that has two important modules: feature extraction and classification. The joint use of Discrete Wavelet Transform (DWT) based feature extraction and nearest neighbor classifier is used to recognize the sign language. The experimental results show that the proposed hand gesture recognition system achieves maximum 99.23% classification accuracy while using cosine distance classifier.

[18] Hand Gesture Recognition Using PCA

In this paper authors presented a scheme using a database driven hand gesture recognition based upon skin color model approach and thresholding approach along with an effective template matching with can be effectively used for human robotics applications and similar other applications.. Initially, hand region is segmented by applying skin color model in YCbCr color space. In the next stage thresholding is applied to separate foreground and background. Finally, template based matching technique is developed using Principal Component Analysis (PCA) for recognition.

[19] Hand Gesture Recognition System For Dumb People

Authors presented the static hand gesture recognition system using digital image processing. For hand gesture feature vector SIFT algorithm is used. The SIFT features have been computed at the edges which are invariant to scaling, rotation, addition of noise.

[20] An Automated System for Indian Sign Language Recognition

In this paper a method for automatic recognition of signs on the basis of shape-based features is presented. For segmentation of hand region from the images, Otsu's thresholding algorithm is used, that chooses an optimal threshold to minimize the within-class variance of thresholded black and white pixels. Features of segmented hand region are calculated using Hu's invariant moments that are fed to Artificial Neural Network for classification. Performance of the system is evaluated on the basis of Accuracy, Sensitivity and Specificity.

[21] Hand Gesture Recognition for Sign Language Recognition: A Review

Authors presented various method of hand gesture and sign language recognition proposed in the past by various researchers. For deaf and dumb people, Sign language is

the only way of communication. With the help of sign language, these physical impaired people express their emotions and thoughts to another person.

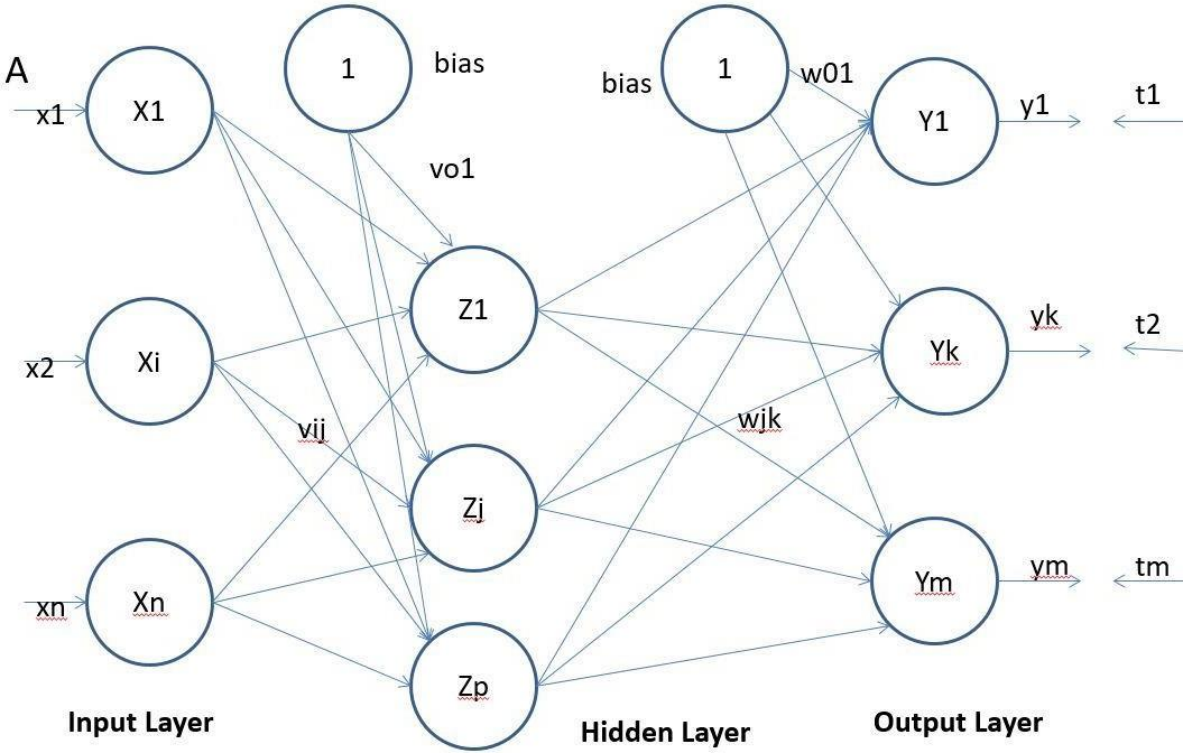
4.0 METHODOLOGY

In this project we will be working on an application that can recognize sign language gestures so that the communication is possible even if someone does not understand sign language. With this work, we intend to take a basic step in bridging this communication gap using Sign Language Recognition. We will start with basic algorithms like logistic regression, KNN and SVM then move on to advance techniques of using convolutional neural networks

The Algorithm for image captioning is

- Convolutional Neural Network

Basic Neural Networks



- X = input training vector (x_1,x_2,x_3…..x_n) [n' in number]
- t = **target output vector (t_1,t_2,t_2…..t_m) ['m' in number]**
- α = **learning rate parameter**

- x_i = i^{th} **input unit**
- v_{0j} = **bias on** j^{th} **hidden unit**
- w_{0k} = **bias on** k^{th} **output unit**
- z_j = j^{th} **hidden unit**
- Then we have input to the j^{th} hidden unit is given by

$$(z_j)_{in} = v_{oj} + \sum_{i=1}^{n} x_i . v_{ij}, j = 1,2...p$$

- The **output from the** j^{th} **hidden unit** is given by

$$z_j = f((z_j)_{in})$$

Here *f()* is the **activation function**

- Let y_k be the k^{th} **output unit**. The **net input** to it is

$$(y_k)_{in} = w_{ok} + \sum_{j=1}^{p} z_j . w_{jk}, k = 1,2...\mathrm{m}$$

- The output from the k^{th} output unit is

$$y_k = f((y_k)_{in})$$

Definition of activation function:- Activation function decides, whether a neuron should be activated or not by calculating weighted sum and further adding bias with it. The purpose of the activation function is to **introduce non-linearity** into the output of a neuron.

Explanation: -

As shown above, neural network has neurons that work in correspondence of *weight, bias* and their respective activation function. In a neural network, we would update the weights and biases of the neurons on the basis of the error at the output. This process is known as *back-propagation*. Activation

functions make the back-propagation possible since the gradients are supplied along with the error to update the weights and biases.

Convolutional Neural Network

Convolutional Neural Networks (ConvNets or CNNs) are a category of Artificial Neural Networks which have proven to be very effective in the field of image recognition and classification. They have been used extensively for the task of object detection, autonomous cars, image captioning etc. First convnet was discovered in the year 1990 by Yann Lecun and the architecture of the model was called as the LeNet architecture. A basic convnet is shown in the fig. below

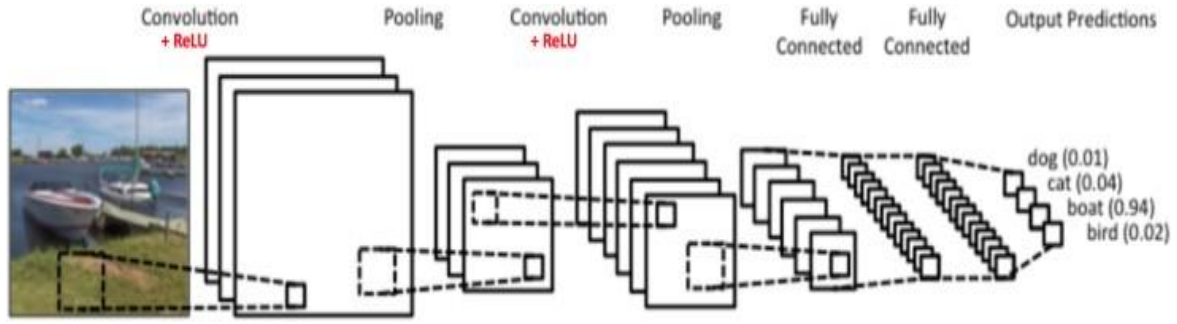


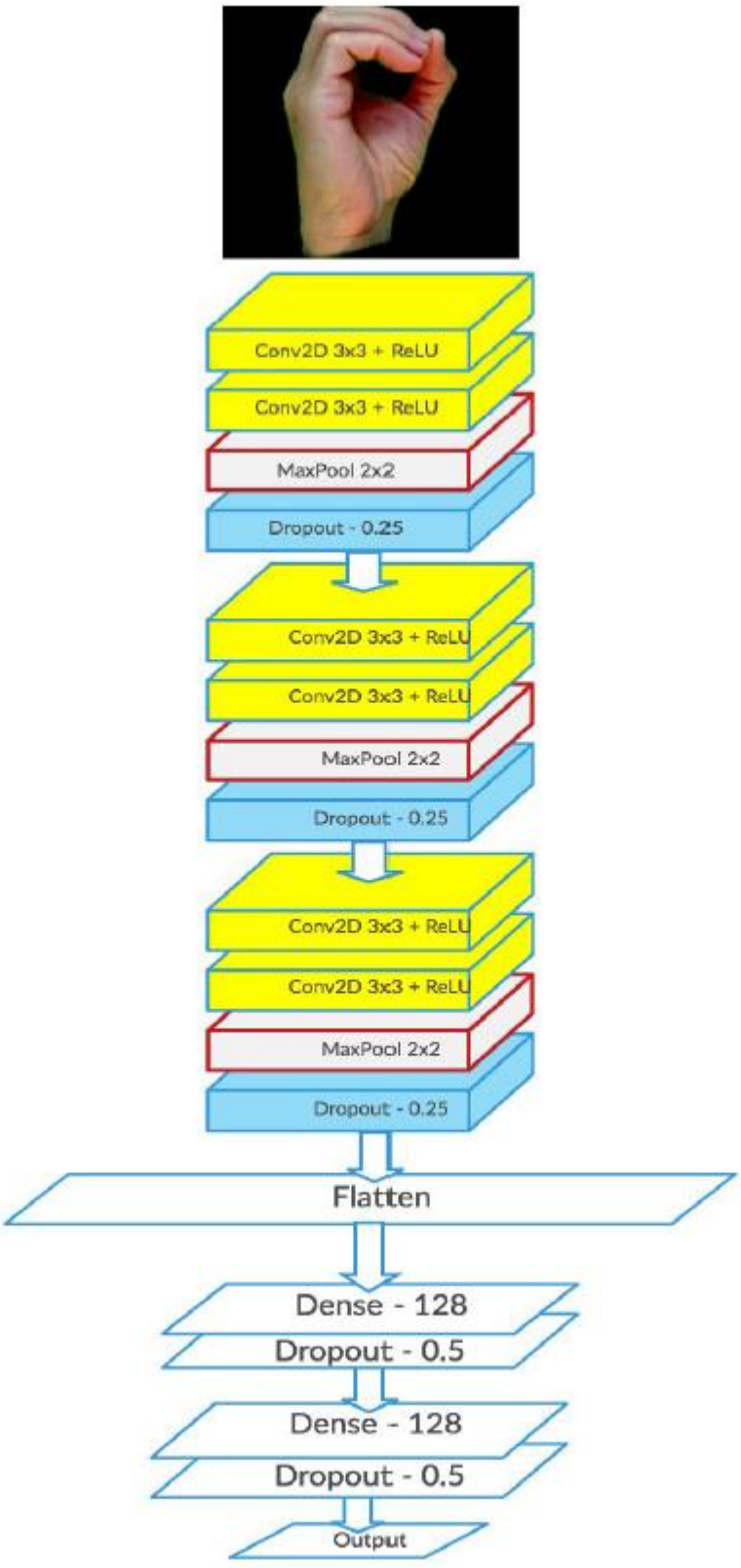
Fig. 1: A simple convnet architecture

The entire architecture of a convnet can be explained using four main operations namely,

- Convolution
- Pooling or Sub Sampling
- Classification (Fully Connected Layer)

These operations are the basic building blocks of every Convolutional Neural Network, so understanding how these works is an important step to developing a sound understanding of ConvNets. We will discuss each of these operations in detail below.

Most implementations surrounding this task have attempted it via transfer learning, but our network was trained from scratch. Our general architecture was a fairly common CNN architecture, consisting of multiple convolutional and dense layers. The architecture included 3 groups of 2 convolutional layers followed by a max-pool layer and a dropout layer, and two groups of fully connected layer followed by a dropout layer and one final output layer.


Conv2D 3x3 + ReLU
Conv2D 3x3 + ReLU
MaxPool 2x2
Dropout - 0.25
Conv2D 3x3 + ReLU
Conv2D 3x3 + ReLU
MaxPool 2x2
Dropout - 0.25
Conv2D 3x3 + ReLU
Conv2D 3x3 + ReLU
MaxPool 2x2
Dropout - 0.25
Flatten
Dense - 128
Dropout - 0.5
Dense - 128
Dropout - 0.5
Output

- OpenCV-OpenCV (Open Source Computer Vision Library) is an open source computer vision and machine learning software library. OpenCV was built to provide a common infrastructure for computer vision applications and to accelerate the use of machine perception in the commercial products. Being a BSD-licensed product, OpenCV makes it easy for businesses to utilize and modify the code.The library has more than 2500 optimized algorithms, which includes a comprehensive set of both classic and state-of-the-art computer vision and machine learning algorithms. These algorithms can be used to detect and recognize faces, identify objects, classify human actions in videos, track camera movements, track moving objects, extract 3D models of objects, produce 3D point clouds from stereo cameras, stitch images together to produce a high resolution image of an entire scene, find similar images from an image database, remove red eyes from images taken using flash, follow eye movements, recognize scenery and establish markers to overlay it with augmented reality, etc.

- Keras-Keras is one of the leading high-level neural networks APIs. It is written in Python and supports multiple back-end neural network computation engines.Keras was created to be user friendly, modular, easy to extend, and to work with Python. The API was "designed for human beings, not machines," and "follows best practices for reducing cognitive load."Neural layers, cost functions, optimizers, initialization schemes, activation functions, and regularization schemes are all standalone modules that you can combine to create new models. New modules are simple to add, as new classes and functions. Models are defined in Python code, not separate model configuration files.

- NumPy-NumPy (Numerical Python) is a linear algebra library in Python. It is a very important library on which almost every data science or machine learning Python packages such as SciPy (Scientific Python), Mat−plotlib (plotting library), Scikit-learn, etc depends on to a reasonable extent.Besides its obvious scientific uses, *NumPy* can also be used as an efficient multi-dimensional container of generic data. Arbitrary data types can be defined. This allows *NumPy* to seamlessly and speedily

integrate with a wide variety of databases.The core functionality of NumPy is its "ndarray", for *n*-dimensional array, data structure. These arrays are strided views on memory. In contrast to Python's built-in list data structure (which, despite the name, is a dynamic array), these arrays are homogeneously typed: all elements of a single array must be of the same type.Such arrays can also be views into memory buffers allocatedy C/C++, Cython, and Fortran extensions to the CPython interpreter without the need to copy data around, giving a degree of compatibility with existing numerical libraries.

- Scikit-Learn-Scikit-learn is a library in Python that provides many unsupervised and supervised learning algorithms. It's built upon some of the technology you might already be familiar with, like NumPy, pandas, and Matplotlib.It features various classification, regression and clustering algorithms including support vector machines, random forests, gradient boosting, *k*-means and DBSCAN.Scikit-learn is largely written in Python, and uses numpy extensively for high-performance linear algebra and array operations. Furthermore, some core algorithms are written in Cython to improve performance.

5.0 PLATFORM:

1) **Anaconda:** Anaconda is a free and open-source distribution of the Python and R programming languages for scientific computing (data science, machine learning applications, large-scale data processing, predictive analytics, etc.), that aims to simplify package management and deployment. Package versions are managed by the package management system *conda*. The Anaconda distribution is used by over 15 million users and includes more than 1500 popular data-science packages suitable for Windows, Linux, and MacOS. Anaconda Navigator is a desktop graphical user interface (GUI) included in Anaconda distribution that allows users to launch applications and manage conda packages, environments and channels without using command-line commands. Navigator can search for packages on Anaconda Cloud or in a local Anaconda Repository, install them in an environment, run the packages and update them. It is available for Windows, macOS and Linux. **Anaconda distribution** comes with more than

1,500 packages as well as the conda package and virtual environment manager. It also includes a GUI, **Anaconda Navigator**, as a graphical alternative to the command line interface (CLI).

The big difference between conda and the pip package manager is in how package dependencies are managed, which is a significant challenge for Python data science and the reason conda exists.

When pip installs a package, it automatically installs any dependent Python packages without checking if these conflict with previously installed packages. It will install a package and any of its dependencies regardless of the state of the existing installation. Because of this, a user with a working installation of, for example, Google Tensorflow, can find that it stops working having used pip to install a different package that requires a different version of the dependent numpy library than the one uses by Tensorflow. In some cases, the package may appear to work but produce different results in detail.

In contrast, conda analyses the current environment including everything currently installed, and, together with any version limitations specified (e.g. the user may wish to have Tensorflow version 2,0 or higher), works out how to install a compatible set of dependencies, warning if this cannot be done.

2) **Jupyter Notebook:** Jupyter Notebook (formerly IPython Notebooks) is a web-based interactive computational environment for creating Jupyter notebook documents. The "notebook" term can colloquially make reference to many different entities, mainly the Jupyter web application, Jupyter Python web server, or Jupyter document format depending on context. A Jupyter Notebook document is a JSON document, following a versioned schema, and containing an ordered list of input/output cells which can contain code, text (using Markdown), mathematics, plots and rich media, usually ending with the ".ipynb" extension.

   A Jupyter Notebook can be converted to a number of open standard output formats (HTML, presentation
   slides, LaTeX, PDF, ReStructuredText, Markdown, Python) through "Download As" in the web interface, via the nbconvert library or "jupyter nbconvert" command line interface in a shell.

   To simplify visualisation of Jupyter notebook documents on the web, the nbconvert library is provided as a service through NbViewer which can take a URL to any publicly available notebook document, convert it to HTML on the fly and display it to the user.

A Jupyter kernel is a program responsible for handling various types of request (code execution, code completions, inspection), and providing a reply. Kernels talk to the other components of Jupyter using ZeroMQ over the network, and thus can be on the same or remote machines. Unlike many other Notebook-like interfaces, in Jupyter, kernels are not aware that they are attached to a specific document, and can be connected to many clients at once. Usually kernels allow execution of only a single language, but there are a couple of exceptions.

In this project we used the jupyter notebook provided by google known as Google Colab. It is a free cloud service, based on Jupyter Notebooks for machine-learning education and research. It provides a runtime fully configured for deep learning and free-of-charge access to a robust GPU.

6.0 Sample Code

Code 1
This piece of code is used to capture images for dataset

```
import cv2
import numpy as np
import os

print("TYPE THE LETTER YOU WANT TO CAPTURE IN CAPITAL")
letter=input()
path = 'C:\\Users\\Sankalp Jain\\ML
Projects\\Sign_Lang_Recog\\Dataset_Thresh\\'+letter
os.mkdir(path)
vc = cv2.VideoCapture(0,cv2.CAP_DSHOW)
pic_no = 0
total_pic = 800
flag_capturing = False
while(vc.isOpened()):
    # read image
    rval, frame = vc.read()
    frame = cv2.flip(frame, 1)
    x=str(pic_no)
    cv2.putText(frame,x,(50, 50),cv2.FONT_HERSHEY_SIMPLEX,2,(0,0,255),2)
    # get hand data from the rectangle sub window on the screen
    cv2.rectangle(frame, (100,100), (300,300), (0,255,0),0)
    cv2.imshow("image", frame)
    crop_img = frame[100:300, 100:300]
    blur = cv2.GaussianBlur(crop_img, (7, 7), 0)
    grey = cv2.cvtColor(blur, cv2.COLOR_BGR2GRAY)
    kernel = np.ones((5, 5))
    dilation = cv2.dilate(grey, kernel, iterations=1)
    erosion = cv2.erode(dilation, kernel, iterations=1)
    final = cv2.GaussianBlur(erosion, (5, 5), 0)
    thresh = cv2.threshold(final,0, 255,
cv2.THRESH_BINARY_INV+cv2.THRESH_OTSU)[1]
    cv2.imshow("threshold", thresh)
    if(pic_no%2==0):
```

```
        thresh=cv2.flip(thresh,1)

    if flag_capturing:
        pic_no += 1
        save_img = cv2.resize(thresh, (50,50) )
        save_img = np.array(save_img)
        cv2.imwrite(path + "/" + str(pic_no) + ".jpg", save_img)

    keypress = cv2.waitKey(1)
    if pic_no == total_pic:
        flag_capturing = False
        break
    if keypress == ord('q'):
        break
    elif keypress == ord('c'):
        flag_capturing = not(flag_capturing)

vc.release()
cv2.destroyAllWindows()
```

Code 2

We uploaded the NPY file to google drive. This code is used to train our CNN model and create a h5 file to use for prediction

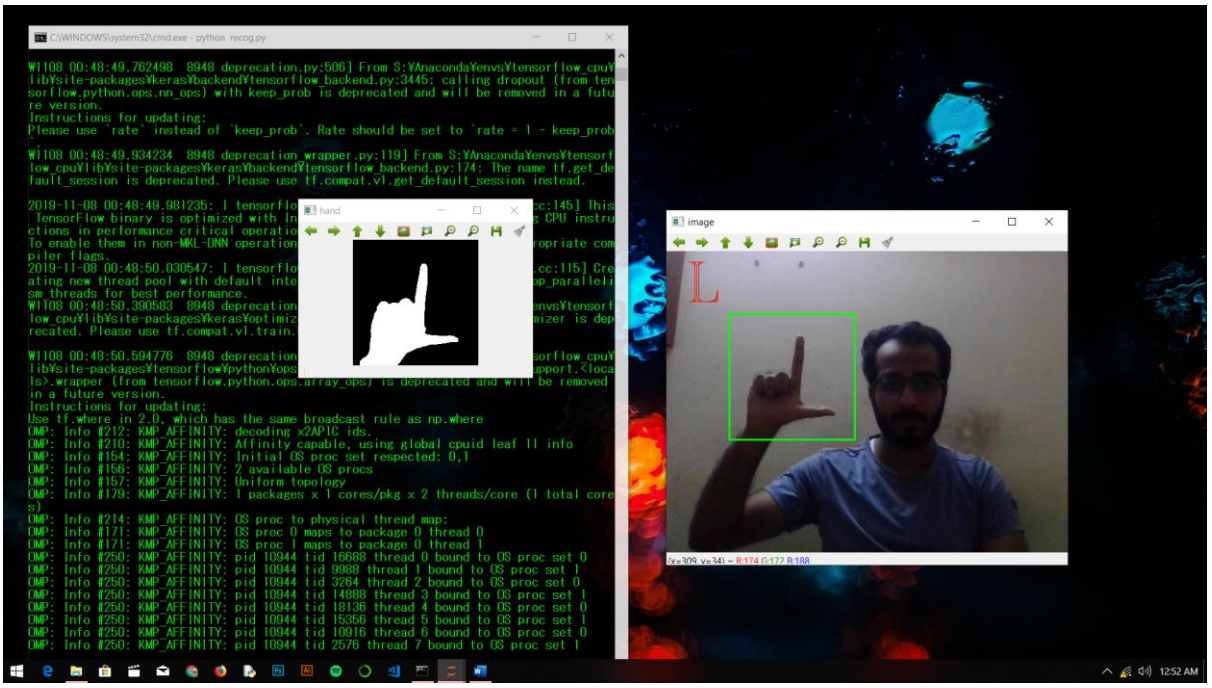
```
#!/usr/bin/env python
# coding: utf-8

from google.colab import drive
drive.mount("/content/drive/")

import numpy as np
from keras.layers import Conv2D, Dense, Flatten, Dropout, MaxPooling2D
from keras.models import Sequential, save_model
from keras.utils import np_utils
from keras.callbacks import ModelCheckpoint
from sklearn.model_selection import train_test_split
from sklearn.utils import shuffle
import os
import cv2
```

```
X=np.load('/content/drive/My Drive/X_array.npy')
Y=np.load('/content/drive/My Drive/Y_array.npy')

Y = np_utils.to_categorical(Y)
Y.shape
Y.shape

categories = Y.shape[1]

X, Y = shuffle(X, Y, random_state=0)
X.shape
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.3)
print(X_train.shape, X_test.shape)
print(Y_train.shape, Y_test.shape)
model = Sequential()
model.add(Conv2D(16, kernel_size=(3,3), activation = 'relu',input_shape=(50,50 ,1) ))
model.add(MaxPooling2D(pool_size = (2, 2)))

model.add(Conv2D(32, kernel_size = (3, 3), activation = 'relu'))
model.add(MaxPooling2D(pool_size = (2, 2)))

model.add(Conv2D(64, kernel_size = (3, 3), activation = 'relu'))
model.add(MaxPooling2D(pool_size = (2, 2)))

model.add(Flatten())
model.add(Dense(128, activation = 'relu'))
model.add(Dropout(0.20))
model.add(Dense(categories, activation = 'softmax'))

model.summary()

model.compile(optimizer='Adam', metrics=['accuracy'],
loss='categorical_crossentropy')

history = model.fit(X_train, Y_train, batch_size=128, epochs=50,
validation_data=[X_test, Y_test])

model.save('/content/drive/My Drive/leg_model.h5')
```

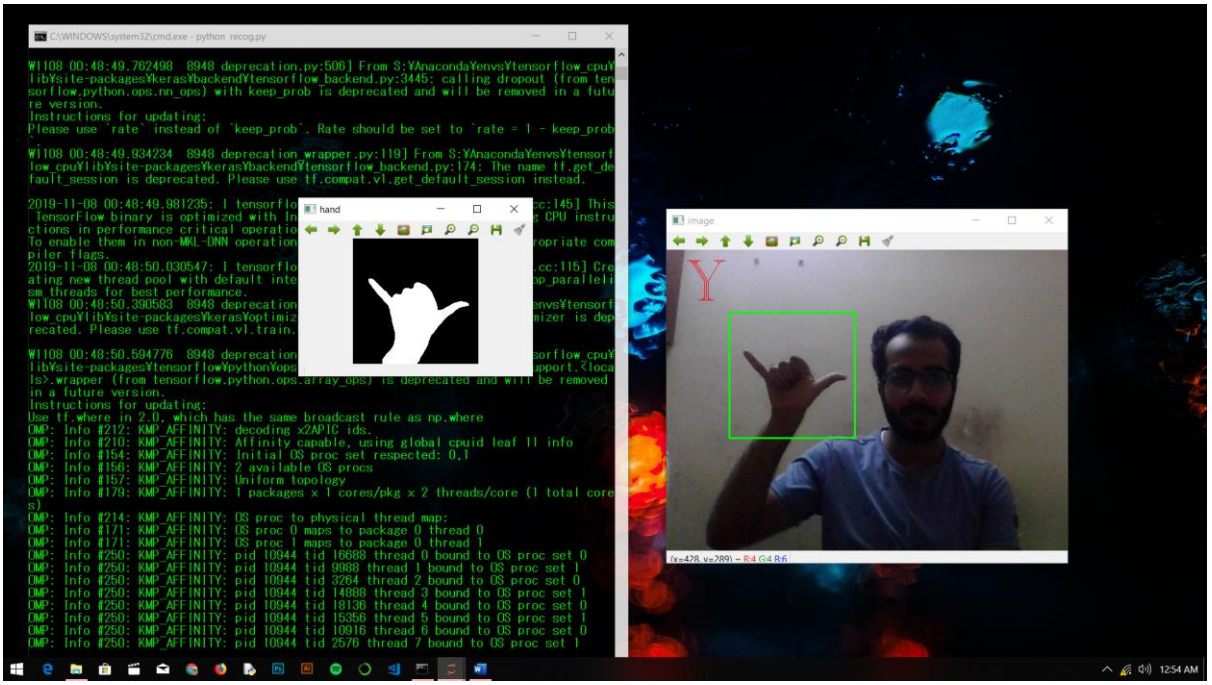
7.0 Test Cases:

For the input of our project, we captured 800 images of the sign corresponding to the letters of an alphabet. Two such cases where we run our trained model in our recognition program, which is shown below:

Test case 1: It consists of creating the images of sign for alphabet ‘L’ in the program



Test Case 2: It consists of creating the images of sign for alphabet ‘Y’ in the program.



The model was able to predict both these letters correctly. Although not having a plane background and lighting conditions might affect the output greatly, in general our model was working fine and predicting the right alphabet for the sign shown.

8.0 RESULT AND DISCUSSION

The results of our project were as follows:

While training the model, the first parameter we considered was the loss over each epoch. To calculate the loss, keras provides val_loss which is the average loss over each epoch. The loss observed initially during training is high but falls gradually
Below is the summary of the model we used consisting of 3 convolutional layers,3 maxpooling layers and 2 dense layers:

9.0 CONCLUSION AND FUTURE SCOPE

Our project aims to make communication simpler between deaf and dumb people by introducing Computer in communication path so that sign language can be automatically captured, recognized, translated to text and displayed it on LCD. There are various methods for sign language conversion. Some of them use wired electronic glove and others use visual based approach.

Electronicgloves are costly and one person cannot use the glove of other person. In vision based approach, different techniques are used to recognize and match the captured gestures with gestures in database. Converting RGB image to binary and matching it with database using a comparing algorithm is simple, efficient and robust technique. This technique is sufficiently accurate to convert sign language into text.

Our hand gesture recognition system was implemented for all 26 letters of alphabet excluding j and z as they involved motion which was beyond the scope of this project.We collected 800 thresholded images of size 50X50 of each alphabet to form the dataset amounting to a total of 19,200 images on which training was conducted.In the end our system was able to predict majority of letters, with minor discrepancies in case of inadequate lighting and other related factors.

Primarily we would like to reduce the error in prediction due factors like changes in lighting conditions by using some suitable means in order to get a better accuracy.

We may move on to recognising words, from as large a dictionary as possible, from American Sign Language. Another method to improve the performance is by using a

more accurate method for finger detection from the time-series curve like near-convex decomposition.

In future work, proposed system can be developed and implemented using Raspberry Pi. Image Processing part should be improved so that System would be able to communicate in both directions i.e.it should be capable of converting normal language to sign language and vice versa. We will try to recognize signs which include motion. Moreover we will focus on converting the sequence of gestures into text i.e. word and sentences and then converting it into the speech which can be heard.

10.0 REFERENCES

[1] Shivashankara S, Srinath S, " American Sign Language Recognition System: An Optimal Approach ", International Journal of Image, Graphics and Signal Processing (IJIGSP), Vol.10, No.8, pp. 18-30, 2018.DOI: 10.5815/ijigsp.2018.08.03

[2] P. Molchanov, S. Gupta, K. Kim and J. Kautz, "Hand gesture recognition with 3D convolutional neural networks," 2015 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), Boston, MA, 2015, pp. 1-7.
DOI: 10.1109/CVPRW.2015.7301342

[3] Jadhav, Akshay N., Gayatri Tatkar, Gauri Hanwate and Rutwik Patwardhan. “Sign Language Recognition.” (2017).

[4] Pandey, Pratibha and Vinay K. Jain. “Hand Gesture Recognition for Sign Language Recognition: A Review.” (2015).

[5] Nayana, P. B., and Sanjeev Kubakaddi. "Implentation of hand gesture recognition technique for HCI using open CV." gesture1 (2014): 5.

[6] M. B. Khan, K. Mishra and M. A. Qadeer, "Gesture recognition using Open-CV," 2017 7th International Conference on Communication Systems and Network Technologies (CSNT), Nagpur, 2017, pp. 167-171. doi: 10.1109/CSNT.2017.8418531

[7] M. A. Qureshi, A. Aziz, M. A. Saeed, M. Hayat and J. S. Rasool, "Implementation of an efficient algorithm for Human hand gesture identification," 2011 Saudi International Electronics, Communications and Photonics Conference (SIECPC), Riyadh, 2011, pp. 1-5.doi: 10.1109/SIECPC.2011.5876948

[8] R. Daroya, D. Peralta and P. Naval, "Alphabet Sign Language Image Classification Using Deep Learning," TENCON 2018 - 2018 IEEE Region 10 Conference, Jeju, Korea (South), 2018, pp. 0646-0650.doi: 10.1109/TENCON.2018.8650241

[9] K. Bantupalli and Y. Xie, "American Sign Language Recognition using Deep Learning and Computer Vision," 2018 IEEE International Conference on Big Data (Big Data), Seattle, WA, USA, 2018, pp. 4896-4899.doi: 10.1109/BigData.2018.8622141

[10] G. A. Rao, K. Syamala, P. V. V. Kishore and A. S. C. S. Sastry, "Deep convolutional neural networks for sign language recognition," 2018 Conference on Signal Processing And Communication Engineering Systems (SPACES), Vijayawada, 2018, pp. 194-197.doi: 10.1109/SPACES.2018.8316344

[11] Neelam K. Gilorkar, Manisha M. Ingle, “Real Time Detection And Recognition Of Indian And American Sign Language Using Sift”, International Journal of Electronics and Communication Engineering & Technology (IJECET), Volume 5, Issue 5, pp. 11-18, May 2014
[12] Umang Patel & Aarti G. Ambekar “Moment based sign language recognition for Indian Languages “2017 Third International Conference on Computing, Communication, Control and Automation (ICCUBEA)
[13] Vision based sign language translation device (conference paper: February 2013 by Yellapu Madhuri and Anburajan Mariamichael).
[14] Himanshu Lilha, Devashish Shivmurthy, “Evaluation of Features for Automated Transcription of Dual- Handed Sign Language Alphabets”, International Conference on Image Information Processing (ICIIP), 3-5 Nov 2011.
[15] Er. Aditi Kalsh, Dr. N.S. Garewal “Sign Language Recognition System” International Journal of Computational Engineering Research 2013
[16] Sunitha K. A, Anitha Saraswathi.P, Aarthi.M, Jayapriya. K, Lingam Sunny, “Deaf Mute Communication Interpreter- A Review”, International Journal of Applied Engineering Research ,Volume 11, pp 290-296 , 2016.
[17] Mathavan Suresh Anand, Nagarajan Mohan Kumar, Angappan Kumaresan, “ An Efficient Framework for Indian Sign Language Recognition Using Wavelet Transform” Circuits and Systems, Volume 7, pp 1874-1883, 2016.
[18] Mandeep Kaur Ahuja, Amardeep Singh, “Hand Gesture Recognition Using PCA”, International Journal of Computer Science Engineering and Technology (IJCSET), Volume 5, Issue 7, pp. 267-27, July 2015.
[19] Sagar P.More, Prof. Abdul Sattar, “Hand gesture recognition system for dumb people”,
[20] International Journal of Science and Research (IJSR)
[21] Chandandeep Kaur, Nivit Gill, “An Automated System for Indian Sign Language Recognition”, International Journal of Advanced Research in Computer Science and Software Engineering