**SIGN LANGUAGE RECOGNITION USING AI**

**ABSTRACT**

Sign language is an expressive way of communication between normal and dumb-deaf people in order to improve the life style of dumb and deaf people the proposed system is developed. About 466 million people are deaf and dumb in the world. The deaf and dumb people are not involved with the social world because of their disabilities. Sign language is a communication skill that is used to convey a meaning of a speaker has thought using gesture. It is a well-structured code gesture, each gesture has a meaning assigned to it. The gesture is a non-verbal communication, which includes the movement of the1 hand, and other body1Parts. There is two main sign language recognition approach namely image- based1and sensor based. Normal person face problem in communicating with disabled people because they cannot understand sign language. In this sign language recognition system, a sign detector detects numbers, which can be easily extended to cover a wide range of other signs and hand signs, including the alphabet. In this system, we are using a machine-learning model known as CNN. Convolutional Neural Networks (CNNs) can learn complicated objects and patterns because they have an input layer, an output layer, numerous hidden layers, and millions of parameters. Propose a system, which helps normal and deaf dumb people to effectively communicate with each other. In resolving, these difficulties with vocally impaired people used an application to enhance the deaf and dumb people.

1. **INTRODUCTION**

Sign language is an expressive way of communication between normal and dumb-deaf people in order to improve the life style of dumb and deaf people the proposed system is developed. About 466 Million people are deaf and dumb in the world. The deaf and dumb people are not involved with the social world because of their disabilities. Sign language is a communication skill that s used to convey a meaning of a speaker’s thought using gesture. It is a well-structured code gesture, each gesture has a meaning assigned to it. The gesture is a non-verbal communication which includes the movement of the1 hand, and other body1Parts. Basically, there is two1main sign language recognition approach namely image- based1and sensor based. A lot of research is going on image based approach because of an advantage of no need to wear devices like hand gloves and helmet etc like in sensor-based approach. Gesture recognition is gaining importance in many application areas such as human interface communication, multimedia and security. Normal person face problem in communicating with disabled people because they cannot understand sign language. There are not many sign language institutions in our society. So, many of dumb people use usual sort of sign language to communicate and they do not have a customized sign language. It is also not possible for the masses to learn sign language. Therefore, a large communication gap still exists between dumb, deaf and normal people. Despite the large number of dumb and deaf people very less research is done in order to reduce the communication barrier. We propose a system which helps normal and deaf dumb people to effectively communicate with each other. In resolving these difficulties with vocally impaired people, we have used an application to enhance the deaf and dumb people. We provide the solution for deaf and dumb people by using this application.

The idea of sign language interpretation is to develop a technique that can identify predefined sign gestures/movements and use them to transfer information. In sign language translator hand gesture recognition method is used, a camera captures the human body movements and communicates the data to a computer that uses the gestures as input to give output as text or speech. The idea of developing a sign language recognition procedure is to develop an interaction between gifted people and normal people and the recognized signs are used to communicate meaningful information. The principal components of a sign gesture recognition process are data acquisition, hand localization, hand characteristics recognition, and gesture identification. The gesture is an indication of physical behavior or emotional expression. It consists of body gestures and hand gestures. It divides into two categories: static gesture and dynamic gesture. Static hand gesture recognition is performed without any additional devices.

A person commands the machine using his bare hands, the person’s hand’s images are captured. and analyzed to determine the meaning of hand gestures. To recognize static gestures a common classifier or template match-er is used. A dynamic movement is supposed as a route between the first phase and the last phase. Dynamic gesture recognition characterizes hand movements. It has four features: - velocity, shape, location, and orientation. The use of this system in areas/places like:hospital, malls, bus, and railway ticket counters, etc Sign language is the mode of communication which uses visual ways like expressions, hand gestures, and body movements to convey meaning. Sign language is extremely helpful for people who face difficulty with hearing or speaking. Sign language recognition refers to the conversion of these gestures into words or alphabets of existing formally spoken languages. Thus, conversion of sign language into words by an algorithm or a model can help bridge the gap between people with hearing or speaking impairment and the rest of the world.Vision-based hand gesture recognition is an area of active current research in computer vision and machine learning. Being a natural way of human interaction, it is an area where many researchers are working on, with the goal of making human computer interaction (HCI) easier and natural, without the need for any extra devices. So, the primary goal of gesture recognition research is to create systems, which can identify specific human gestures and use them, for example, to convey information. For that, vision-based hand gesture interfaces require fast and extremely robust hand detection, and gesture recognition in real time. Hand gestures are a powerful human communication modality with lots of potential applications and in this context, we have sign language recognition, the communication method of deaf people.

**CHAPTER -2**

**SYSTEM ANALYSIS**

**EXISTING SYSTEM**

In the existing system, blind people can only read Braille script. Braille is a tactile writing system used by people who1are blind people. It is traditionally written1with embossed Traditionally, gesture recognition method was divided into two categories namely vision based and sensor based method. In vision based method, the computer camera is an input device for varies gestures of hands and figures. In sensor based systems, gloves are used which can achieve the accurate positions of hand gesture. Lots of studies have been done on sensor-based approaches like gloves, helmets etc. But wearing it continuously is not possible.

**DISADVANTAGES**

* Using the tool regularly is difficult
* Manual effort is needed to identify
* Time consumption is needed to recognize the symbols
* Guidance is required to translate the signals

**PROPOSED SYSTEM**

In this project, a python software application is developed to translate the signal to the character and the character to the signal for the deaf and dumb people without any intermediate. This proposed system will directly convey the character with this developed application. This system will ensure the accurate character in which the end user gives the input that dynamically process and give the relevant content. The consistency in analyzing the character is more convenient in this proposed system. The very first issue we faced was of dataset. To deal with raw images and that too square image as 3DCNN in Keras as it was a lot more convenient working with only square images. Find any existing dataset for that hence decided to make our own dataset. Second issue was to select a filter, which we could apply on our images so that proper features of the images could be obtained and hence, and then we could provide that image as input for 3DCNN model.

**ADVANTAGES**

* Less time is consumed to give the accurate details
* Character are identified accurately
* The user must be within a defined perimeter area, in front of the camera.
* The user must be within a defined distance range, due to camera limitations.
* Hand pose is defined with a bare hand and not occluded by other objects.
* The proposed system architecture, which consists of two modules, namely: data acquisition, pre-processing and feature extraction and sign language gesture classification.

**3. SYSTEM REQUIREMENTS**

**3.1 HARDWARE REQUIREMENTS**

* Operating system : Windows 10
  + Coding Language : Python
  + Tool : Python IDLE
  + Database : MYMYSQL
  1. **SOFTWARE REQUIREMENTS**
* System : Pentium IV 2.4 GHz.
* Hard Disk : 500 GB.
* Monitor : 15 VGA Colour.
* Mouse : Logitech.
* Ram : 4GB

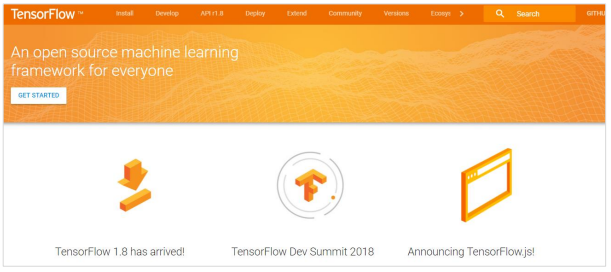
**3.3 SOFTWARE DESCRIPTION**

**3.3.1 .PYTHON**

Python is an [interpreter](https://en.wikipedia.org/wiki/Interpreted_language), [high-level](https://en.wikipedia.org/wiki/High-level_programming_language) and [general-purpose programming language](https://en.wikipedia.org/wiki/General-purpose_programming_language). Python's design philosophy emphasizes [code readability](https://en.wikipedia.org/wiki/Code_readability) with its notable use of [significant indentation](https://en.wikipedia.org/wiki/Off-side_rule). Its [language constructs](https://en.wikipedia.org/wiki/Language_construct) and [object-oriented](https://en.wikipedia.org/wiki/Object-oriented_programming) approach aim to help [programmers](https://en.wikipedia.org/wiki/Programmers) write clear, logical code for small and large-scale projects. Python is dynamically typed and garbage-collected. It supports multiple programming paradigms, including structured (particularly, procedural), object-oriented and functional programming. Python is often described as a "batteries included" language due to its comprehensive standard library. Python is Interpreted − the interpreter processes Python at runtime. You do not need to compile your program before executing it. This is similar to PERL and PHP. Python is Interactive − You can actually sit at a Python prompt and interact with the interpreter directly to write your programs. Python is Object-Oriented − Python supports Object-Oriented style or technique of programming that encapsulates code within objects. Python is a Beginner's Language − Python is a great language for the beginner-level programmers and supports the development of a wide range of applications from simple text processing to WWW browsers to games.

**TENSORFLOW-** **INTRODUCTION**

Tensor Flow is a software library or framework, designed by the Google team to implement machine learning and deep learning concepts in the easiest manner. It combines the computational algebra of optimization techniques for easy calculation of many mathematical expressions. The official website of TensorFlow is mentioned below: https://www.tensorflow.org/



Let us now consider the following important features of TensorFlow:

• It includes a feature of that defines, optimizes and calculates mathematical expressions easily with the help of multi-dimensional arrays called tensors.

• It includes a programming support of deep neural networks and machine learning techniques.

• It includes a high scalable feature of computation with various data sets.

• TensorFlow uses GPU computing, automating management. It also includes a unique feature of optimization of same memory and the data used.

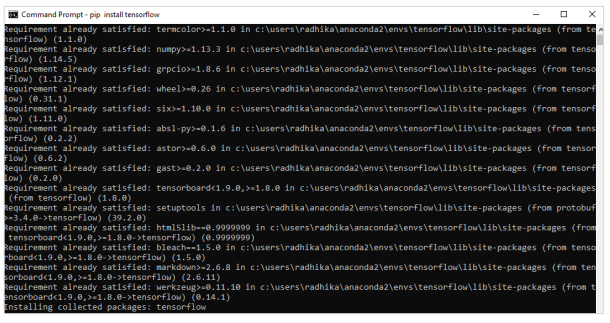
**Why is TensorFlow So Popular?**

TensorFlow is well-documented and includes plenty of machine learning libraries. It offers a few important functionalities and methods for the same. TensorFlow is also called a “Google” product. It includes a variety of machine learning and deep learning algorithms. TensorFlow can train and run deep neural networks for handwritten digit classification, image recognition, word embedding and creation of various sequence models.

**TensorFlow — Installation**

To install TensorFlow, it is important to have “Python” installed in your system. Python version 3.4+ is considered the best to start with TensorFlow installation. Consider the following steps to install TensorFlow in Windows operating system.

**pip install tensorflow**

****

**TensorFlow — Convolutional Neural Networks**

After understanding machine-learning concepts, we can now shift our focus to deep learning concepts. Deep learning is a division of machine learning and is considered as a crucial step taken by researchers in recent decades. The examples of deep learning implementation include applications like image recognition and speech recognition.

Following are the two important types of deep neural networks:

• Convolutional Neural Networks

• Recurrent Neural Networks In this chapter, we will focus on the CNN, Convolutional Neural Networks

**Convolutional Neural Networks**

Convolutional Neural networks are designed to process data through multiple layers of arrays. This type of neural networks is used in applications like image recognition or face recognition. The primary difference between CNN and any other ordinary neural network is that CNN takes input as a two-dimensional array and operates directly on the images rather than focusing on feature extraction which other neural networks focus on. The dominant approach of CNN includes solutions for problems of recognition. Top companies like Google and Facebook have invested in research and development towards recognition projects to get activities done with greater speed.

A convolutional neural network uses three basic ideas:

• Local respective fields

• Convolution

• Pooling

Let us understand these ideas in detail.

CNN utilizes spatial correlations that exist within the input data. Each concurrent layer of a neural network connects some input neurons. This specific region is called local receptive field. Local receptive field focusses on the hidden neurons. The hidden neurons process the input data inside the mentioned field not realizing the changes outside the specific boundary.

If we observe the above representation, each connection learns a weight of the hidden neuron with an associated connection with movement from one layer to another. Here, individual neurons perform a shift from time to time. This process is called “convolution”. The mapping of connections from the input layer to the hidden feature map is defined as “shared weights” and bias included is called “shared bias”. CNN or convolutional neural networks use pooling layers, which are the layers, positioned immediately after CNN declaration. It takes the input from the user as a feature map that comes out of convolutional networks and prepares a condensed feature map. Pooling layers helps in creating layers with neurons of previous layers.

**KERAS**

**INTRODUCTION**

Deep learning is one of the major subfield of machine learning framework. Machine learning is the study of design of algorithms, inspired from the model of human brain. Deep learning is becoming more popular in data science fields like robotics, artificial intelligence(AI), audio & video recognition and image recognition. Artificial neural network is the core of deep learning methodologies. Deep learning is supported by various libraries such as Theano, TensorFlow, Caffe, Mxnet etc., Keras is one of the most powerful and easy to use python library, which is built on top of popular deep learning libraries like TensorFlow, Theano, etc., for creating deep learning models.

**OVERVIEW OF KERAS**

Keras runs on top of open source machine libraries like TensorFlow, Theano or Cognitive Toolkit (CNTK). Theano is a python library used for fast numerical computation tasks. TensorFlow is the most famous symbolic math library used for creating neural networks and deep learning models. TensorFlow is very flexible and the primary benefit is distributed computing. CNTK is deep learning framework developed by Microsoft. It uses libraries such as Python, C#, C++ or standalone machine learning toolkits. Theano and TensorFlow are very powerful libraries but difficult to understand for creating neural networks. Keras is based on minimal structure that provides a clean and easy way to create deep learning models based on TensorFlow or Theano. Keras is designed to quickly define deep learning models. Well, Keras is an optimal choice for deep learning applications.

**FEATURES**

Keras leverages various optimization techniques to make high level neural network API easier and more performant. It supports the following features:

• Consistent, simple and extensible API.

• Minimal structure - easy to achieve the result without any frills.

• It supports multiple platforms and backends.

• It is user friendly framework which runs on both CPU and GPU.

• Highly scalability of computation.

**BENEFITS**

Keras is highly powerful and dynamic framework and comes up with the following advantages:

• Larger community support.

• Easy to test.

• Keras neural networks are written in Python which makes things simpler.

• Keras supports both convolution and recurrent networks.

• Deep learning models are discrete components, so that, you can combine into many ways.

**KERAS ― OVERVIEW OF DEEP LEARNING**

Deep learning is an evolving subfield of machine learning. Deep learning involves analyzing the input in layer by layer manner, where each layer progressively extracts higher level information about the input. Let us take a simple scenario of analyzing an image. Let us assume that your input image is divided up into a rectangular grid of pixels. Now, the first layer abstracts the pixels. The second layer understands the edges in the image. The Next layer constructs nodes from the edges. Then, the next would find branches from the nodes. Finally, the output layer will detect the full object. Here, the feature extraction process goes from the output of one layer into the input of the next subsequent layer. By using this approach, we can process huge amount of features, which makes deep learning a very powerful tool. Deep learning algorithms are also useful for the analysis of unstructured data. Let us go through the basics of deep learning in this chapter.

**Artificial Neural Networks**

The most popular and primary approach of deep learning is using “Artificial neural network” (ANN). They are inspired from the model of human brain, which is the most complex organ of our body. The human brain is made up of more than 90 billion tiny cells called “Neurons”. Neurons are inter-connected through nerve fiber called “axons” and “Dendrites”. The main role of axon is to transmit information from one neuron to another to which it is connected.

Similarly, the main role of dendrites is to receive the information being transmitted by the axons of another neuron to which it is connected. Each neuron processes a small information and then passes the result to another neuron and this process continues. This is the basic method used by our human brain to process huge about of information like speech, visual, etc., and extract useful information from it.

Based on this model, the first Artificial Neural Network (ANN) was invented by psychologist Frank Rosenblatt, in the year of 1958. ANNs are made up of multiple nodes which is similar to neurons. Nodes are tightly interconnected and organized into different hidden layers. The input layer receives the input data and the data goes through one or more hidden layers sequentially and finally the output layer predict something useful about the input data. For example, the input may be an image and the output may be the thing identified in the image, say a “Cat”.

A single neuron (called as perceptron in ANN) can be represented as below:

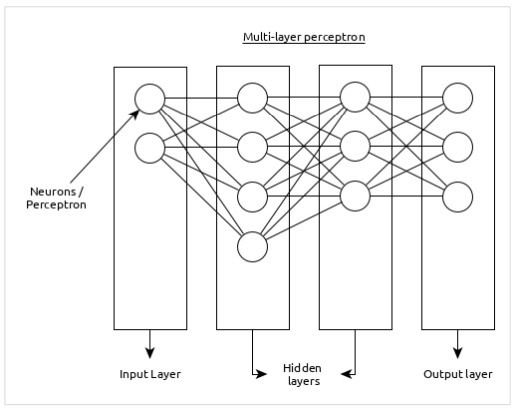
• Multiple input along with weight represents dendrites.

• Sum of input along with activation function represents neurons. Sum actually means computed value of all inputs and activation function represent a function, which modify the Sum value into 0, 1 or 0 to 1.

• Actual output represent axon and the output will be received by neuron in next layer. Let us understand different types of artificial neural networks in this section.

**Multi-Layer Perceptron**

Multi-Layer perceptron is the simplest form of ANN. It consists of a single input layer, one or more hidden layer and finally an output layer. A layer consists of a collection of perceptron. Input layer is basically one or more features of the input data. Every hidden layer consists of one or more neurons and process certain aspect of the feature and send the processed information into the next hidden layer. The output layer process receives the data from last hidden layer and finally output the result.

****

**Convolutional Neural Network (CNN)**

Convolutional neural network is one of the most popular ANN. It is widely used in the fields of image and video recognition. It is based on the concept of convolution, a mathematical concept. It is almost similar to multi-layer perceptron except it contains series of convolution layer and pooling layer before the fully connected hidden neuron layer.

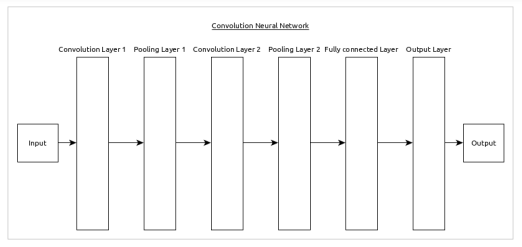
It has three important layers:

• Convolution layer: It is the primary building block and perform computational tasks based on convolution function.

• Pooling layer: It is arranged next to convolution layer and is used to reduce the size of inputs by removing unnecessary information so computation can be performed faster.

• Fully connected layer: It is arranged to next to series of convolution and pooling layer and classify input into various categories.

A simple CNN can be represented as below:



Here,

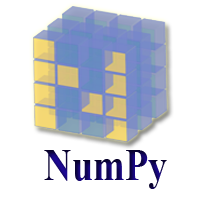
• 2 series of Convolution and pooling layer is used and it receives and process the input (e.g. image).

• A single fully connected layer is used and it is used to output the data (e.g. classification of image)

PYTHON NUMPY

Our Python NumPy Tutorial provides the basic and advanced concepts of the NumPy. Our NumPy tutorial is designed for beginners and professionals.

NumPy stands for numeric python which is a python package for the computation and processing of the multidimensional and single dimensional array elements.



What is NumPy

NumPy stands for numeric python which is a python package for the computation and processing of the multidimensional and single dimensional array elements.

Travis Oliphant created NumPy package in 2005 by injecting the features of the ancestor module Numeric into another module Numarray.

It is an extension module of Python which is mostly written in C. It provides various functions which are capable of performing the numeric computations with a high speed.

NumPy provides various powerful data structures, implementing multi-dimensional arrays and matrices. These data structures are used for the optimal computations regarding arrays and matrices.

In this tutorial, we will go through the numeric python library NumPy.

The need of NumPy

With the revolution of data science, data analysis libraries like NumPy, SciPy, Pandas, etc. have seen a lot of growth. With a much easier syntax than other programming languages, python is the first choice language for the data scientist.

NumPy provides a convenient and efficient way to handle the vast amount of data. NumPy is also very convenient with Matrix multiplication and data reshaping. NumPy is fast which makes it reasonable to work with a large set of data.

There are the following advantages of using NumPy for data analysis.

1. NumPy performs array-oriented computing.
2. It efficiently implements the multidimensional arrays.
3. It performs scientific computations.
4. It is capable of performing Fourier Transform and reshaping the data stored in multidimensional arrays.
5. NumPy provides the in-built functions for linear algebra and random number generation.

Nowadays, NumPy in combination with SciPy and Mat-plotlib is used as the replacement to MATLAB as Python is more complete and easier programming language than MATLAB.

Prerequisite

Before learning Python Numpy, you must have the basic knowledge of Python concepts.

**PILLOW**

**PYTHON PILLOW ― OVERVIEW**

In today’s digital world, we come across lots of digital images. In case, we are working with Python programming language, it provides lot of image processing libraries to add image processing capabilities to digital images. Some of the most common image processing libraries are: OpenCV, Python Imaging Library (PIL), Scikit-image, Pillow. However, in this tutorial, we are only focusing on Pillow module and will try to explore various capabilities of this module. Pillow is built on top of PIL (Python Image Library). PIL is one of the important modules for image processing in Python. However, the PIL module is not supported since 2011 and doesn’t support python 3. Pillow module gives more functionalities, runs on all major operating system and support for python 3. It supports wide variety of images such as “jpeg”, “png”, “bmp”, “gif”, “ppm”, “tiff”. You can do almost anything on digital images using pillow module. Apart from basic image processing functionality, including point operations, filtering images using built-in convolution kernels, and color space conversions.

**IMAGE ARCHIVES**

The Python Imaging Library is best suited for image archival and batch processing applications. Python pillow package can be used for creating thumbnails, converting from one format to another and print images, etc.

**IMAGE DISPLAY**

You can display images using Tk PhotoImage, BitmapImage and Windows DIB interface, which can be used with PythonWin and other Windows-based toolkits and many other Graphical User Interface (GUI) toolkits.

For debugging purposes, there is a show () method to save the image to disk which calls the external display utility.

**Image Processing**

The Pillow library contains all the basic image processing functionality. You can do image resizing, rotation and transformation.

Pillow module allows you to pull some statistics data out of image using histogram method, which later can be used for statistical analysis and automatic contrast enhancement.

**PYTHON PILLOW — ENVIRONMENT SETUP**

This chapter discusses how to install pillow package in your computer.

Installing pillow package is very easy, especially if you’re installing it using pip.

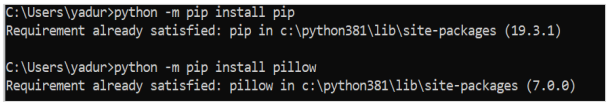
**Installing Pillow using pip**

To install pillow using pip, just run the below command in your command prompt:

python -m pip install pip

python -m pip install pillow

In case, if pip and pillow are already installed in your computer, above commands will simply mention the ‘requirement already satisfied’ as shown below:

****

**PYTHON PILLOW — USING IMAGE MODULE**

To display the image, pillow library is using an image class within it. The image module inside pillow package contains some important inbuilt functions like, load images or create new images, etc.

**Opening, rotating anddisplaying an image**

To load the image, we simply import the image module from the pillow and call the Image.open(), passing the image filename.

Instead of calling the Pillow module, we will call the PIL module as to make it backward compatible with an older module called Python Imaging Library (PIL). That’s why our code starts with “from PIL import Image” instead of “from Pillow import Image”.

Next, we’re going to load the image by calling the Image.open() function, which returns a value of the Image object data type. Any modification we make to the image object can be saved to an image file with the save() method. The image object we received using Image.open(), later can be used to resize, crop, draw or other image manipulation method calls on this Image object.

**Example**

Following example demonstrates the rotation of an image using python pillow:

from PIL import Image

#Open image using Image module

im = Image.open("images/cuba.jpg")

#Show actual Image im.show()

#Show rotated Image

im = im.rotate(45) im.show()

**Output**

If you save the above program as Example.py and execute, it displays the original and rotated images using standard PNG display utility, as follows:

**SCIPY**

What is SciPy

The SciPy is an open-source scientific library of Python that is distributed under a BSD license. It is used to solve the complex scientific and mathematical problems. It is built on top of the Numpy extension, which means if we import the SciPy, there is no need to import Numpy. The **Scipy** is pronounced as **Sigh pi**, and it depends on the Numpy, including the appropriate and fast N-dimension array manipulation.

It provides many user-friendly and effective numerical functions for numerical integration and optimization.

The **SciPy** library supports **integration, gradient optimization, special functions, ordinary differential equation solvers, parallel programming tools**, and many more. We can say that **SciPy** implementation exists in every complex numerical computation.

The **scipy** is a data-processing and system-prototyping environment as similar to MATLAB. It is easy to use and provides great flexibility to scientists and engineers.

## History

Python was expanded in the 1990s to include an array type for numerical computing called numeric. This numeric package was replaced by Numpy (blend of Numeric and NumArray) in 2006. There was a growing number of extension module and developers were interested to create a complete environment for scientific and technical computing. **Travis Oliphant**, **Eric Jones**, and **Pearu Peterson** merged code they had written and called the new package **SciPy**. The newly created package provided a standard collection of common numerical operation on the top of Numpy.

## Why use SciPy?

SciPy contain significant mathematical algorithms that provide easiness to develop sophisticated and dedicated applications. Being an open-source library, it has a large community across the world to the development of its additional module, and it is much beneficial for scientific application and data scientists.

## Numpy vs. SciPy

Numpy and SciPy both are used for mathematical and numerical analysis. Numpy is suitable for basic operations such as sorting, indexing and many more because it contains array data, whereas SciPy consists of all the numeric data.

Numpy contains many functions that are used to resolve the linear algebra, Fourier transforms, etc. whereas SciPy library contains full featured version of the linear algebra module as well many other numerical algorithms.

# SciPy Sub - Packages

SciPy has the number of sub-packages for the various scientific computing domains. The following table is given below:

|  |  |  |
| --- | --- | --- |
| **Sr.** | **Sub-Package** | **Description** |
| **1.** | scipy.cluster | Cluster algorithms are used to vector quantization/ Kmeans. |
| **2.** | scipy.constants | It represents physical and mathematical constants. |
| **3.** | scipy.fftpack | It is used for **Fourier transform**. |
| **4.** | scipy.integrate | Integration routines |
| **5.** | scipy.interpolation | Interpolation |
| **6.** | scipy.linalg | It is used for linear algebra routine. |
| **7.** | scipy.io | It is used for data input and output. |
| **8.** | scipy.ndimage | It is used for the n-dimension image. |
| **9.** | scipy.odr | Orthogonal distance regression. |
| **10.** | scipy.optimize | It is used for optimization. |
| **11.** | scipy.signal | It is used in signal processing. |
| **12.** | scipy.sparse | Sparse matrices and associated routines. |
| **13.** | scipy.spatial | Spatial data structures and algorithms. |
| **14.** | scipy.special | Special Function. |
| **15.** | scipy.stats | Statistics. |
| **16.** | scipy.weaves | It is a tool for writing. |

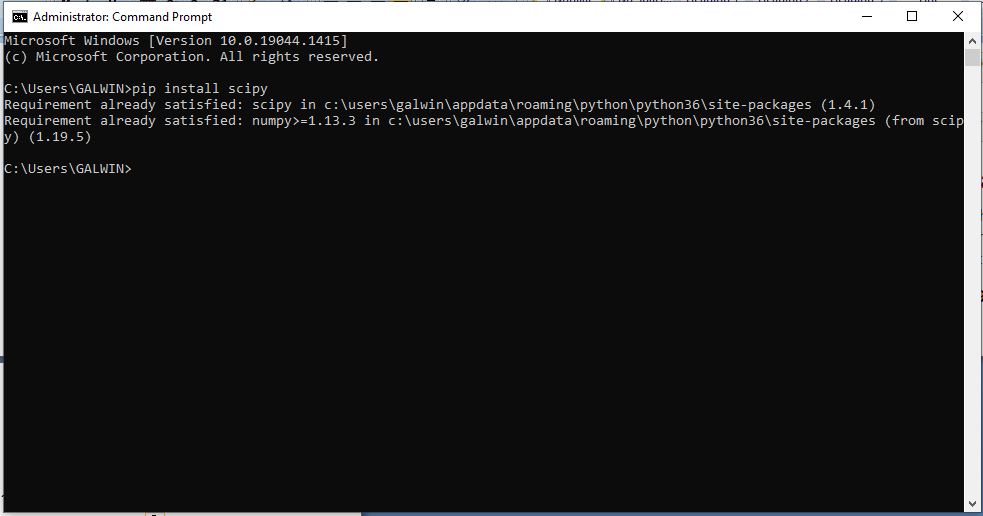
# SciPy Installation

We will learn about the core functionality of SciPy. Before working with SciPy, it should be installed in the system.

* **Install SciPy using pip**

We can install the SciPy library by using **pip** command; run the following command in the terminal:

1. pip install scipy



**OPEN CV**

OpenCV is an open-source library for the computer vision. It provides the facility to the machine to recognize the faces or objects. In this tutorial we will learn the concept of OpenCV using the Python programming language.

Our OpenCV tutorial includes all topics of Read and Save Image, Canny Edge Detection, Template matching, Blob Detection, Contour, Mouse Event, Gaussian blur and so on.

What is OpenCV?

OpenCV is a Python open-source library, which is used for computer vision in Artificial intelligence, Machine Learning, face recognition, etc.

In OpenCV, the CV is an abbreviation form of a computer vision, which is defined as a field of study that helps computers to understand the content of the digital images such as photographs and videos.

The purpose of computer vision is to understand the content of the images. It extracts the description from the pictures, which may be an object, a text description, and three-dimension model, and so on. For example, cars can be facilitated with computer vision, which will be able to identify and different objects around the road, such as traffic lights, pedestrians, traffic signs, and so on, and acts accordingly.

Computer vision allows the computer to perform the same kind of tasks as humans with the same efficiency. There are a two main task which are defined below:

* **Object Classification -** In the object classification, we train a model on a dataset of particular objects, and the model classifies new objects as belonging to one or more of your training categories.
* **Object Identification -** In the object identification, our model will identify a particular instance of an object - for example, parsing two faces in an image and tagging one as Virat Kohli and other one as Rohit Sharma.

## History

OpenCV stands for Open Source Computer Vision Library, which is widely used for image recognition or identification. It was officially launched in 1999 by Intel. It was written in C/C++ in the early stage, but now it is commonly used in Python for the computer vision as well.

The first alpha version of OpenCV was released for the common use at the IEEE Conference on Computer Vision and Pattern Recognition in 2000, and between 2001 and 2005, five betas were released. The first 1.0 version was released in 2006.

The second version of the OpenCV was released in October 2009 with the significant changes. The second version contains a major change to the C++ interface, aiming at easier, more type-safe, pattern, and better implementations. Currently, the development is done by an independent Russian team and releases its newer version in every six months.

## How OpenCV Works

In this tutorial, we will learn how computers perform image recognition.

### How does computer recognize the image?

Human eyes provide lots of information based on what they see. Machines are facilitated with seeing everything, convert the vision into numbers and store in the memory. Here the question arises how computer convert images into numbers. So the answer is that the pixel value is used to convert images into numbers. A pixel is the smallest unit of a digital image or graphics that can be displayed and represented on a digital display device.

The picture intensity at the particular location is represented by the numbers. In the above image, we have shown the pixel values for a grayscale image consist of only one value, the intensity of the black color at that location.

There are two common ways to identify the images:

**1. Grayscale**

Grayscale images are those images which contain only two colors black and white. The contrast measurement of intensity is black treated as the weakest intensity, and white as the strongest intensity. When we use the grayscale image, the computer assigns each pixel value based on its level of darkness.

**2. RGB**

An RGB is a combination of the red, green, blue color which together makes a new color. The computer retrieves that value from each pixel and puts the results in an array to be interpreted.

Why OpenCV is used for Computer Vision?

* OpenCV is available for free of cost.
* Since the OpenCV library is written in C/C++, so it is quit fast. Now it can be used with Python.
* It require less RAM to usage, it maybe of 60-70 MB.
* Computer Vision is portable as OpenCV and can run on any device that can run on C.

# Installation of the OpenCV

### Install OpenCV using Anaconda

The first step is to download the latest Anaconda graphic installer for Windows from it [official site](https://www.anaconda.com/distribution/). Choose your bit graphical installer. You are suggested to install 3.7 working with Python 3.

**MYSQL SERVER**

**Definition:**

Microsoft MYSQL Server is a powerful relational database management system catering to high-end users with advanced needs. Along with Oracle, Microsoft MYSQL Server is widely regarded as one of the two main full-featured database systems on the market today.

**MYSQL Server:**

Microsoft MYSQL Server offers tight integration with the Back Office series of server products. Check out this collection of MYSQL Server links about migrating to MYSQL Server, performance tuning, development and more!

**Structured Query Language (MYSQL)**

The Structured Query Language (MYSQL) forms the backbone of most modern database systems. These links provide the best resources on the Net for neophytes and expert database administrators alike!

###### MYSQL Server Management

Applications Manager MYSQL Server Monitoring software capability helps DBAs monitor the performance and availability of production databases. It is an agentless monitoring solution that provides out-of-the-box performance metrics for ensuring the MYSQL Server runs efficiently.

Applications Manager provides a web client that helps you visualize a MYSQL Server Database farm and provides in-depth monitoring data that helps you make educated decisions about usage patterns, plan capacity and alert you of impending problems. The Root Cause analysis window helps the operations team to troubleshoot performance problems quickly. Additionally the grouping capability helps you to group your databases based on the business process supported and helps the operations team to prioritize alerts as they are received.

MYSQL server monitoring has the ability to connect to the database source & monitor various system table column values, collect data, etc. and also notify through alerts, if the database system properties are beyond a given threshold.

Some of the components that are monitored in MS MYSQL database are:

###### MYSQL Server Monitoring Capabilities

* Out-of-the-box monitoring of MS MYSQL availability and performance.
* Monitors performance statistics such as memory usage, buffer manager statistics, connection statistics, cache details, MYSQL statistics, etc., Alerts can be configured for these parameters.
* Based on the thresholds configured, notifications and alerts are generated if the MYSQL Server or any specified database within the server is not accessible. Actions are executed automatically based on configurations.
* Performance graphs and reports are available instantly. Reports can be grouped and displayed based on availability, health, and connection time.

###### DB2 Monitoring

Applications Manager DB2 Server Monitoring capability helps database administrators (DBAs) monitor the availability and performance of production databases. It is an agentless database monitoring software that provides out-of-the-box performance metrics for ensuring the IBM DB2 database server runs efficiently.

The database monitoring tool provides a web client that helps you to visualize the network of DB2 Servers. It provides in-depth monitoring data that helps you make educated decisions about usage patterns, plan capacity and alert you of impending problems. The Root Cause analysis window helps the operations team to troubleshoot performance problems quickly. Additionally the grouping apability helps you to group your databases based on the business process supported and helps the operations team to prioritize alerts as they are received.

###### Introduction to MYSQL

The Structured Query Language (MYSQL) is the language of databases. All modern relational databases, including Access, FileMaker Pro, Microsoft MYSQL Server and Oracle use MYSQL as their basic building block. In fact, it’s often the only way that you can truly interact with the database itself. All of the fancy graphical user interfaces that provide data entry and manipulation functionality are nothing more than MYSQL translators. They take the actions you perform graphically and convert them to MYSQL commands understood by the database.At this point, you might be thinking that you’re not a programmer and learning a programming language is certainly not up your alley. Fortunately, at its core, MYSQL is a simple language. It has a limited number of commands and those commands are very readable and are almost structured like English sentences.

Before we get started, it’s important that you have a basic understanding of how databases work. If you’re comfortable with terms like table, relation and query, feel free to plow right ahead! If not, you may wish to read the article before moving on.Let’s look at an example. Suppose you have a simple database designed to keep the inventory for a convenience store. One of the tables in your database might contain the prices of the items on your shelves indexed by unique stock numbers that identify each item.

**CHAPTER 4**

**IMPLEMENTATION**

**MODULES**

* Open CV
* Sign Dataset
* Pre-process sign language
* Result
* Algorithm
  + 3DCNN

**MODULES DESCRIPTION**

**OPEN CV**

OpenCV refers to ASCII text file laptop Vision, machine learning, and image process and currently it plays a serious role in data processing that is extremely necessary in today’s systems. By mistreatment it, one will method pictures and videos to spot objects, faces, or perhaps handwriting of an individual's. once it's integrated with numerous libraries, like NumPy, python is capable of process the OpenCV array structure for analysis. to spot image pattern and its numerous options we tend to use vector area and perform mathematical operations on these options. it's a cross- platform middle-to-high level API that consists of a number of hundred C functions. it's free for each non-commercial and business use. OpenCV has the advantage of being a multi-platform framework; it supports each Windows and UNIX, and additional recently, Mac OS. It needs less RAM for its usage, it perhaps of 60-70 MB. mistreatment OpenCV library, can −

Read and write pictures

Capture and save videos

Process pictures (filter, transform)

Perform feature detection

**Dataset Upload**

The project makes use of the Live camera a sign language hand symbols dataset. Yes,No,Thank you,please,love you the original dataset. Table 1 presents the test set accuracies for standard 3DCNN with two layers with 3 × 3 size kernel and 2 × 2 MaxPooling for various input sizes from 360 × 360 to 48 × 48 on both sign dataset datasets. The dataset from a Kaggle sign language dataset is used for training and testing. ASL dataset created by B. Kang et al is used. It is a collection of live hand gesture images for each of the 5 classes. These gestures are recorded for a total of five subjects. The gestures include Yes,No,Thank you,please,love you because these require movements of hand and thus cannot be captured in the form of an image. Some of the gestures are very similar, (0/o) , (V/2) and (W/6). These are classified by context or meaning.The size of the test was randomly selected 10% of the available data

**PRE-PROCESS SIGN LANGUAGE**

**CONVOLUTION LAYER**

The primary purpose of Convolution just in case of a CNN is to extract options from the input image. Convolution preserves the spatial relationship between pixels by learning image options victimisation tiny squares of computer file. The convolution layer’s parameters include a collection of learnable filters. each filter is tiny spatially (along dimension and height), however extends through the complete depth of the input volume. throughout the passing play, every filter is convolved across the dimension and height of the input volume and computes dot merchandise between the entries of the filter and therefore the input at any position. because the filter convolves over the dimension and height of the input volume it produces a 2-dimensional activation map that provides the responses of that filter at each spatial position.

### **RECTIFIED LINEAR UNIT**

An additional operation known as ReLU has been used once each Convolution operation. A corrected long measure (ReLU) could be a cell of a neural network that uses the subsequent activation operate to calculate its output given x:

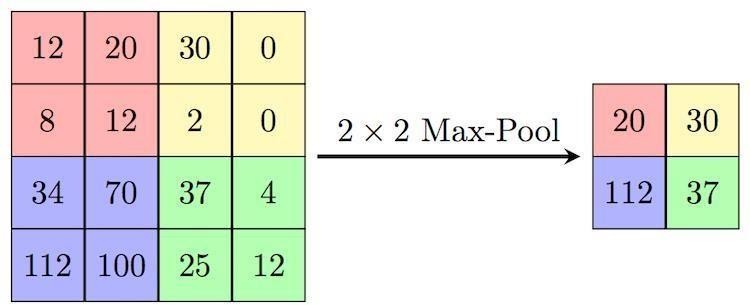
*A(x)=max(0,x)*

Using these cells is a lot of economical than sigmoid and still forwards a lot of data compared to binary units. once initializing the weights uniformly, half the weights square measure negative. This helps produce a thin feature illustration. Another positive facet is that the comparatively low cost computation. No function needs to be calculated. This operate additionally prevents the vanishing gradient error, since the gradients square measure linear functions or zero however in no case nonlinear functions.

### **POOLING LAYER**

Spatial Pooling (also known as subsampling or down sampling) reduces the spatial property of every feature map however retains the foremost necessary data. spatial Pooling are often of various types: liquid ecstasy, Average, Sum etc. just in case of liquid ecstasy Pooling, a spatial neighborhood (for example, a 2×2 window) is outlined and therefore the largest part is taken from the corrected feature map inside that window. just in case of average pooling the typical or total of all parts therein window is taken. In follow, liquid ecstasy Pooling has been shown to figure higher. liquid ecstasy Pooling reduces the input by applying the most operate over the input xi.

The operate of Pooling is to increasingly scale back the spatial size of the input illustration. especially, pooling makes the input representations (feature dimension) smaller and a lot of manageable. It reduces the quantity of parameters and computations within the network, therefore, dominant over-fitting. It makes the network invariant to tiny transformations, distortions and translations within the input image (a tiny distortion in input won't amendment the output of Pooling. It additionally helps America make AN nearly scale invariant illustration. {this is|this is often|this are often} terribly powerful since objects can be detected in a picture regardless of wherever they're situated.



**FULLY CONNECTED LAYER**

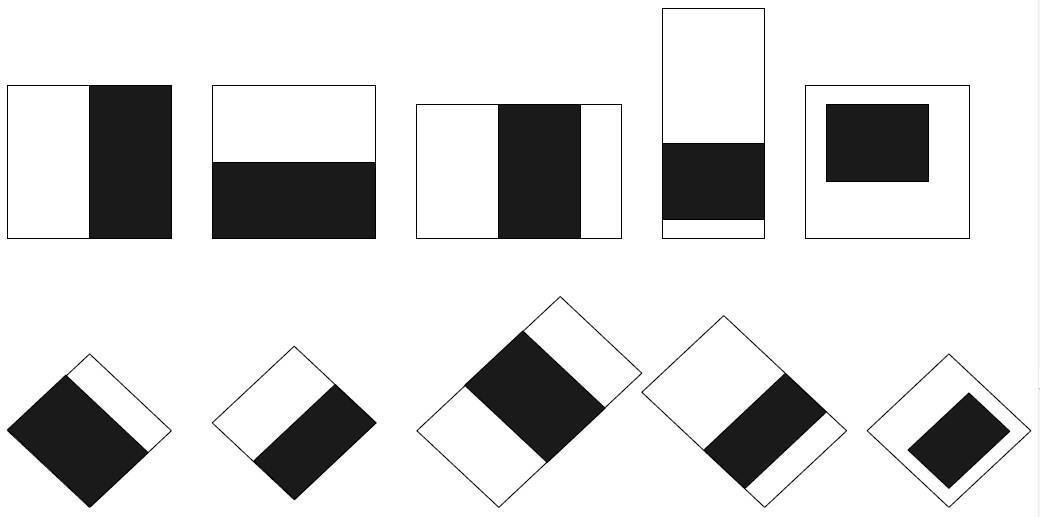
Fully Connected layer may be a ancient Multi-Layer Perceptron that uses a SoftMax activation perform within the output layer. The term “Fully Connected” implies that each somatic cell within the previous layer is connected to each somatic cell on future layer. The output from the convolutional and pooling layers represents high-level options of the input image. the aim of the totally Connected layer is to use these options for classifying the input image into numerous categories supported the coaching dataset. SoftMax is employed for activation functions. It treats the outputs as scores for every category. within the SoftMax, the perform mapping stayed unchanged and these scores area unit understood because the un-normalized log chances for every category. with the exception of classification, adding a fully-connected layer is additionally a (usually) low-cost method of learning non-linear mixtures of those options. Most of the options from convolutional and pooling layers could also be smart for the classification task, however mixtures of these options can be even higher. The total of output chances from the totally Connected Layer is one. this is often ensured by mistreatment the because the activation perform within the output layer of the totally Connected Layer.

**HAAR-CASCADE CLASSIFIERS**

This technique of object detection was projected by Paul Viola and Michael Jones in their paper. A Haar classifier, or a Haar cascade classifier, may be a machine learning object detection program that identifies objects in a picture and video. A Haar- Feature is simply sort of a kernel in CNN, except that during a CNN, the values of the kernel area unit determined by coaching, whereas a Haar-Feature is manually determined. Haar- options area unit smart at police work edges and contours. This makes it particular effective in face detection. The Haar feature endlessly traverses from the highest left of the image to all-time low right to look for the actual feature.Positive pictures – These pictures contain the pictures that we would like our classifier to spot. Negative pictures – pictures of everything else, that don't contain the item we would like to find.

**FUTURE EXTRACTION**

Features ar primarily the most building block for any dataset. it's a measurable property of the item in analysis. More often, options ar represented because the input variable ‘x’ that is employed for predicting the labels, ‘y’. In giant datasets, the amount of options is additionally giant, so it'll take longer in process the feature set. the method of feature extraction comes into image here. Feature extraction may be a method of spatiality reduction by that associate initial set of information is reduced to additional manageable teams for process. A characteristic of those giant knowledge sets may be a sizable amount of variables that need a great deal of computing resources to method. Feature extraction is that the name for strategies



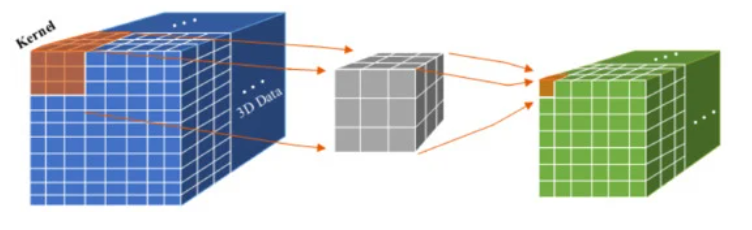
that choose and /or mix variables into options, effectively reducing the number of knowledge that has to be processed, whereas still accurately and utterly describing the initial knowledge set. the method of feature extraction is beneficial after you ought to cut back the amount of resources required for process while not losing vital or relevant info. Feature extraction may cut back the number of redundant knowledge for a given analysis. Also, the reduction of the info and also the machine’s efforts in building variable mixtures (features) facilitate the speed of learning and generalization steps within the machine learning method.

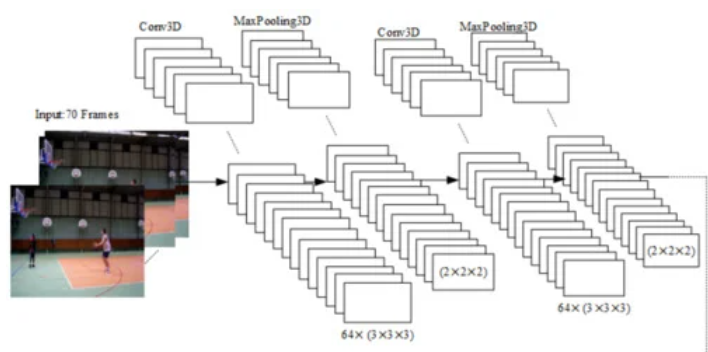
**RESULT**

The creation of our proposed model, we have used the tensorflow deep learning. The framework provides for the creation of deep networks by choosing appropriate layers and specifying the preceding and succeeding layers in the design. The inputs to the framework can be in the HDF5 format, which is particularly suitable for the representation of 2D data, such as a kaggle dataset. The steps in preparing the data are explained in the previous section, and are the same for each kaggle dataset images. Hence, we have one HDF5 file representing all the sign language symbols, and each HDF5 file has the data along with the label. This label is used in both the training and testing phase.

**3DCNN –Three-dimensional convolutional neural network**

The primary purpose of 3D Convolution in case of a 3DCNN is to extract features from the input image. Convolution preserves the spatial relationship between pixels by learning image features using small squares of input data. The convolution layer’s parameters consist of a set of learnable filters. Every filter is small spatially (along width and height), but extends through the full depth of the input volume. During the forward pass, each filter is convolved across the width and height of the input volume and computes dot products between the entries of the filter and the input at any position. As the filter convolves over the width and height of the input volume it produces a 3-dimensional activation map that gives the responses of that filter at every spatial position.





**3DCNN-based methods**

### **Rectified Linear Unit**

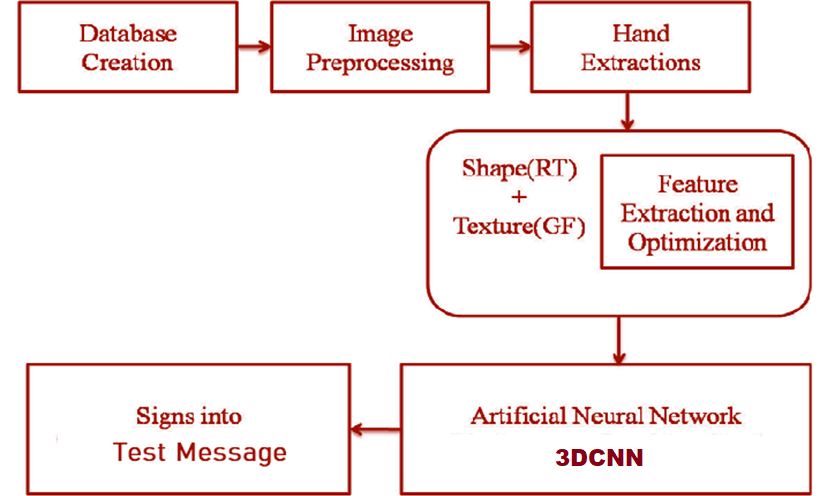
An additional operation known as ReLU has been used once each Convolution operation. A corrected long measure (ReLU) could be a cell of a neural network that uses the subsequent activation operate to calculate its output given x:

*A(x)=max(0,x)*

Using these cells is a lot of economical than sigmoid and still forwards a lot of data compared to binary units. once initializing the weights uniformly, half the weights square measure negative. This helps produce a thin feature illustration. Another positive facet is that the comparatively low cost computation. No function needs to be calculated. This operate additionally prevents the vanishing gradient error, since the gradients square measure linear functions or zero however in no case nonlinear functions.

**6. SYSTEM DESIGN**

**6.1 SYSTEM ARCHITECTURE**



**6. 2 DATAFLOW DIAGRAM**

**LEVEL 0**

Login

Admin

**LEVEL 1**

Login

Live Image

Prediction of image

DataBase

Processing CNN

Admin

Database

Result

**7. SYSTEM TESTING**

System testing involves user training system testing and successful running of the developed proposed system. The user tests the developed system and changes are made according to their needs. The testing phase involves the testing of developed system using various kinds of data.An elaborate testing of data is prepared and the system is tested using the test data. While testing, errors are noted and the corrections are made. The corrections are also noted for the future use. The users are trained to operate the developed system.

# **INTRODUCTION**

System testing is the stage of implementation that is aimed at ensuring that the system works accurately and efficiently before live operation commences. Testing is vital to the success of the system. System testing makes logical assumption that if all the parts of the system are correct, then the goal will be successfully achieved. A series of testing are done for the proposed system before the system is ready for the user acceptance testing.

# The following are the types of Testing:

# Unit Testing

# Integration testing

* Validation
* Output testing
* User acceptance testing

# **UNIT TESTING:**

Unit testing focuses verification efforts on the smallest unit of the software design, the module. This is also known as “module testing”. The modules are tested separately. This testing was carried out during programming stage itself. In this testing each module is found to be working satisfactorily as regards to the expected output from the module.

# **INTEGRATION TESTING:**

Data can be lost across an interface: one module can have adverse efforts on another. Integration testing is the systematic testing for construction of program structure, while at the same time conducting tests to uncover errors associated within the interface. Here correction is difficult because the isolation of cause is complicated by the cast expense of the entire program. Thus in the integration testing step, all the errors uncovered are corrected for the next testing steps.

# **VALIDATION TESTING:**

At the conclusion of integration testing, software is completely assembled as a package, interfacing errors have been uncovered and corrected and a final series of software tests begins validation test has been conducted one of the two possible conditions exists. One is the function or performance characteristics confirm to specification and are accepted and the other is deviation from specification is uncovered and a deficiency list is created.

# **SYSTEM TESTING:**

After performance validation testing, the next step is output testing of the proposed system since no system could be useful if it does not produce the required output in a specific format. Asking the users about the format required by them tests the outputs generated by the system under consideration. Here, the output format is considered in two ways, one is on the screen and the other is printed format. The output format on the screen is found to be correct, as the format was designed in the system design phase according to the user needs. For the hard copy also the output comes as the specified requirements by the user.

# **USER ACCEPTANCE TESTING:**

User acceptance testing of a system is the key factor of the success of any system. The system under study is tested for the user acceptance by constantly keeping in touch with the prospective system users at any time of developing and making changes whenever required.

**DEPLOYMENT**

During the deployment design phase of the solution life cycle, you design a high-level deployment architecture and a low-level implementation specification, and prepare a series of plans and specifications necessary to implement the solution. Project approval occurs in the deployment design phase.The whole process has been designed for the user side to enable the standard level of security to their important information and data that has been stored into the cloud. It develops the administrators' performance evaluation in a better way.

**MAINTENANCE**

Maintainability is considered, inherent to the building system design, ensuring the ease, accuracy, safety, and economy of maintenance tasks within that system. The purpose of maintainability is to improve effectiveness and efficiency of maintenance.

**8. CONCLUSION**

The Sign language recognition system is developed to bridge the communication gaps between speech-impaired people and normal people. In this survey, we briefly discussed various systems. The System consists of software in which software system is much cost-effective, accurate. As in matlab they used flex sensor, image processing, etc. A software system can be implemented by using various algorithms/architectures like RF,KNN,Logistic reversion, CNN and 3DCNN. In the software system, found that 3DCNN algorithms had the highest recognition rate used the most. Most systems used sign language alphabets not actual few word.

**FUTURE ENHANCEMENT**

This system is focused on improving the performance and adding more special features with regard to the current system. We are focusing on implementing support for multiple card handling process between each database, which would improve the overall performance and also in addition we are trying to globalize the process, in which the data can be dispatched along with different environments. The card handling behavior can be enhanced via mobile technologies with recent advanced logs. Once the user was missed the card the system will be to provide a future to lock the card by using their mobile phones.

**9. BIBLIOGRAPHY**

**Book Reference**

* Python Programming. Python Programming for Beginners, Python Programming for Intermediates 115 Pages·2016·9.89 MB·137,411
* Master Python Programming with a unique Hands-On Project Have you always wanted to learn computer A Python Book: Beginning Python, Advanced Python, and Python278 Pages·2013·1.3 MB·144,167
* Python Data Analytics: Data Analysis and Science Using Pandas, matplotlib, and the Python ...Gray hat Python : Python programming for hackers and reverse 220 Pages·2009·3.04 MB·92,820
* Black hat Python : Python programming for hackers and pentesters195 Pages·2015·10.04 MB·66,359
* Python Practical Python Programming For Beginners and Experts161 Pages·2016·11.15 MB·34,951

**REFERENCES**

[1]. P. K. Athira, C. J. Sruthi, A. Lijiya, “A Signer Independent Sign Language Recognition with Co-articulation Elimination from Live Videos: An Indian Scenario”,2019.https://www.sciencedirect.com/science/article/pii/S131915781831228X

[2]. M. Jaiswal, V. Sharma, A. Sharma, S. Saini, R. Tomar, “An Efficient Binarized Neural Network for Recognizing Two Hands Indian Sign Language Gestures in Real-time nvironment”,2020https://ieeexplore.ieee.org/abstract/document/9342454

[3]. M. Al-Hammadi, G. Muhammad, w. Abdul, M. Alsulaiman, M. A. Bencherif, M. A. Mekhtiche, “Hand Gesture Recognition for Sign Language Using 3DCNN”, 2020.https://ieeexplore.ieee.org/abstract/document/9078786

[4]Sevgi Z. Gurbuz, Ali Cafer Gurbuz, "Evie American Sign Language Recognition Using RF Sensing"2020.https://link.springer.com/article/1878685

[5]R. Rastgoo, K. Kiani, S. Escalera, “Video-based isolated hand sign language recognition using a deep cascaded model”, 2020.https://link.springer.com/article/10.1007%2Fs11042-020-09048-5

[6]. R. Rastgoo, K. Kiani, S. Escalera, “Video-based isolated hand sign language recognition using a deep cascaded model”, 2020.https://link.springer.com/article/10.1007%2Fs11042-020-09048-5

**Web References:**

# www.w3schools.com

# www.python.com

* www.7lesson.com
* www.apython.com
* www.pythonsourcecode.com

**APPENDIX**

**A.1 SOURCE CODE**

import tensorflow as tf

from google.protobuf import text\_format

from object\_detection import exporter

from object\_detection.protos import pipeline\_pb2

slim = tf.contrib.slim

flags = tf.app.flags

flags.DEFINE\_string('input\_type', 'image\_tensor', 'Type of input node. Can be '

'one of [`image\_tensor`, `encoded\_image\_string\_tensor`, '

'`tf\_example`]')

flags.DEFINE\_string('input\_shape', None,

'If input\_type is `image\_tensor`, this can explicitly set '

'the shape of this input tensor to a fixed size. The '

'dimensions are to be provided as a comma-separated list '

'of integers. A value of -1 can be used for unknown '

'dimensions. If not specified, for an `image\_tensor, the '

'default shape will be partially specified as '

'`[None, None, None, 3]`.')

flags.DEFINE\_string('pipeline\_config\_path', None,

'Path to a pipeline\_pb2.TrainEvalPipelineConfig config '

'file.')

flags.DEFINE\_string('trained\_checkpoint\_prefix', None,

'Path to trained checkpoint, typically of the form '

'path/to/model.ckpt')

flags.DEFINE\_string('output\_directory', None, 'Path to write outputs.')

flags.DEFINE\_string('config\_override', '',

'pipeline\_pb2.TrainEvalPipelineConfig '

'text proto to override pipeline\_config\_path.')

flags.DEFINE\_boolean('write\_inference\_graph', False,

'If true, writes inference graph to disk.')

tf.app.flags.mark\_flag\_as\_required('pipeline\_config\_path')

tf.app.flags.mark\_flag\_as\_required('trained\_checkpoint\_prefix')

tf.app.flags.mark\_flag\_as\_required('output\_directory')

FLAGS = flags.FLAGS

def main(\_):

pipeline\_config = pipeline\_pb2.TrainEvalPipelineConfig()

with tf.gfile.GFile(FLAGS.pipeline\_config\_path, 'r') as f:

text\_format.Merge(f.read(), pipeline\_config)

text\_format.Merge(FLAGS.config\_override, pipeline\_config)

if FLAGS.input\_shape:

input\_shape = [

int(dim) if dim != '-1' else None

for dim in FLAGS.input\_shape.split(',')

]

else:

input\_shape = None

exporter.export\_inference\_graph(

FLAGS.input\_type, pipeline\_config, FLAGS.trained\_checkpoint\_prefix,

FLAGS.output\_directory, input\_shape=input\_shape,

write\_inference\_graph=FLAGS.write\_inference\_graph)

if \_\_name\_\_ == '\_\_main\_\_':

tf.app.run()

import numpy as np

import os

import sys

import tensorflow as tf

from distutils.version import StrictVersion

from collections import defaultdict

from PIL import Image

from object\_detection.utils import ops as utils\_ops

from time import sleep

import urllib.request

import six.moves.urllib as urllib

# This is needed since the notebook is stored in the object\_detection folder.

sys.path.append("..")

if StrictVersion(tf.\_\_version\_\_) < StrictVersion('1.9.0'):

raise ImportError('Please upgrade your TensorFlow installation to v1.9.\* or later!')

from utils import label\_map\_util

from utils import visualization\_utils as vis\_util

MODEL\_NAME = 'inference\_graph'

PATH\_TO\_FROZEN\_GRAPH = MODEL\_NAME + '/frozen\_inference\_graph.pb'

PATH\_TO\_LABELS = 'training/labelmap.pbtxt'

detection\_graph = tf.Graph()

with detection\_graph.as\_default():

od\_graph\_def = tf.GraphDef()

with tf.gfile.GFile(PATH\_TO\_FROZEN\_GRAPH, 'rb') as fid:

serialized\_graph = fid.read()

od\_graph\_def.ParseFromString(serialized\_graph)

tf.import\_graph\_def(od\_graph\_def, name='')

category\_index = label\_map\_util.create\_category\_index\_from\_labelmap(PATH\_TO\_LABELS, use\_display\_name=True)

def run\_inference\_for\_single\_image(image, graph):

if 'detection\_masks' in tensor\_dict:

# The following processing is only for single image

detection\_boxes = tf.squeeze(tensor\_dict['detection\_boxes'], [0])

detection\_masks = tf.squeeze(tensor\_dict['detection\_masks'], [0])

# Reframe is required to translate mask from box coordinates to image coordinates and fit the image size.

real\_num\_detection = tf.cast(tensor\_dict['num\_detections'][0], tf.int32)

detection\_boxes = tf.slice(detection\_boxes, [0, 0], [real\_num\_detection, -1])

detection\_masks = tf.slice(detection\_masks, [0, 0, 0], [real\_num\_detection, -1, -1])

detection\_masks\_reframed = utils\_ops.reframe\_box\_masks\_to\_image\_masks(

detection\_masks, detection\_boxes, image.shape[0], image.shape[1])

detection\_masks\_reframed = tf.cast(

tf.greater(detection\_masks\_reframed, 0.5), tf.uint8)

# Follow the convention by adding back the batch dimension

tensor\_dict['detection\_masks'] = tf.expand\_dims(

detection\_masks\_reframed, 0)

image\_tensor = tf.get\_default\_graph().get\_tensor\_by\_name('image\_tensor:0')

# Run inference

output\_dict = sess.run(tensor\_dict,

feed\_dict={image\_tensor: np.expand\_dims(image, 0)})

# all outputs are float32 numpy arrays, so convert types as appropriate

output\_dict['num\_detections'] = int(output\_dict['num\_detections'][0])

output\_dict['detection\_classes'] = output\_dict[

'detection\_classes'][0].astype(np.uint8)

output\_dict['detection\_boxes'] = output\_dict['detection\_boxes'][0]

output\_dict['detection\_scores'] = output\_dict['detection\_scores'][0]

global a2

if 'detection\_masks' in output\_dict:

output\_dict['detection\_masks'] = output\_dict['detection\_masks'][0]

return output\_dict

import serial

ser = serial.Serial(port = "COM32", baudrate = '9600',timeout=0.5)

import cv2

cap = cv2.VideoCapture(0)

url='http://192.168.1.4:8080/shot.jpg'

try:

with detection\_graph.as\_default():

with tf.Session() as sess:

# Get handles to input and output tensors

ops = tf.get\_default\_graph().get\_operations()

all\_tensor\_names = {output.name for op in ops for output in op.outputs}

tensor\_dict = {}

for key in [

'num\_detections', 'detection\_boxes', 'detection\_scores',

'detection\_classes', 'detection\_masks'

]:

tensor\_name = key + ':0'

if tensor\_name in all\_tensor\_names:

tensor\_dict[key] = tf.get\_default\_graph().get\_tensor\_by\_name(

tensor\_name)

while True:

imgPath = urllib.request.urlopen(url)

imgNp = np.array(bytearray(imgPath.read()), dtype=np.uint8)

image\_np = cv2.imdecode(imgNp, -1)

#(\_\_, image\_np) = cap.read()

# Expand dimensions since the model expects images to have shape: [1, None, None, 3]

image\_np\_expanded = np.expand\_dims(image\_np, axis=0)

cv2.imwrite('capture.jpg',image\_np)

# Actual detection.

output\_dict = run\_inference\_for\_single\_image(image\_np, detection\_graph)

# Visualization of the results of a detection.

if output\_dict['detection\_classes'][0] == 1 and output\_dict['detection\_scores'][0] > 0.60:

print('Hello')

ser.write('1'.encode())

vis\_util.visualize\_boxes\_and\_labels\_on\_image\_array(

image\_np,

output\_dict['detection\_boxes'],

output\_dict['detection\_classes'],

output\_dict['detection\_scores'],

category\_index,

instance\_masks=output\_dict.get('detection\_masks'),

use\_normalized\_coordinates=True,

line\_thickness=8)

ser.write('1'.encode())

if output\_dict['detection\_classes'][0] == 2 and output\_dict['detection\_scores'][0] > 0.75 :

print('Love You')

ser.write('2'.encode())

vis\_util.visualize\_boxes\_and\_labels\_on\_image\_array(

image\_np,

output\_dict['detection\_boxes'],

output\_dict['detection\_classes'],

output\_dict['detection\_scores'],

category\_index,

instance\_masks=output\_dict.get('detection\_masks'),

use\_normalized\_coordinates=True,

line\_thickness=8)

ser.write('2'.encode())

if output\_dict['detection\_classes'][0] == 3 and output\_dict['detection\_scores'][0] > 0.40:

print('No')

ser.write('3'.encode())

vis\_util.visualize\_boxes\_and\_labels\_on\_image\_array(

image\_np,

output\_dict['detection\_boxes'],

output\_dict['detection\_classes'],

output\_dict['detection\_scores'],

category\_index,

instance\_masks=output\_dict.get('detection\_masks'),

use\_normalized\_coordinates=True,

line\_thickness=8)

ser.write('3'.encode())

if output\_dict['detection\_classes'][0] == 4 and output\_dict['detection\_scores'][0] > 0.30 :

print('Please')

ser.write('4'.encode())

vis\_util.visualize\_boxes\_and\_labels\_on\_image\_array(

image\_np,

output\_dict['detection\_boxes'],

output\_dict['detection\_classes'],

output\_dict['detection\_scores'],

category\_index,

instance\_masks=output\_dict.get('detection\_masks'),

use\_normalized\_coordinates=True,

line\_thickness=8)

ser.write('4'.encode())

if output\_dict['detection\_classes'][0] == 5 and output\_dict['detection\_scores'][0] > 0.60:

print('ThankYou')

ser.write('5'.encode())

vis\_util.visualize\_boxes\_and\_labels\_on\_image\_array(

image\_np,

output\_dict['detection\_boxes'],

output\_dict['detection\_classes'],

output\_dict['detection\_scores'],

category\_index,

instance\_masks=output\_dict.get('detection\_masks'),

use\_normalized\_coordinates=True,

line\_thickness=8)

ser.write('5'.encode())

if output\_dict['detection\_classes'][0] == 6 and output\_dict['detection\_scores'][0] > 0.30:

print('Yes')

ser.write('6'.encode())

vis\_util.visualize\_boxes\_and\_labels\_on\_image\_array(

image\_np,

output\_dict['detection\_boxes'],

output\_dict['detection\_classes'],

output\_dict['detection\_scores'],

category\_index,

instance\_masks=output\_dict.get('detection\_masks'),

use\_normalized\_coordinates=True,

line\_thickness=8)

ser.write('6'.encode())

cv2.imshow('object\_detection', cv2.resize(image\_np,(800,600)))

if cv2.waitKey(1)& 0xFF == ord('q'):

cap.release()

cv2.destroyAllWindows()

break

except Exception as e:

print(e)

#cap.release()

"""Binary to run train and evaluation on object detection model."""

from \_\_future\_\_ import absolute\_import

from \_\_future\_\_ import division

from \_\_future\_\_ import print\_function

from absl import flags

import tensorflow as tf

from object\_detection import model\_hparams

from object\_detection import model\_lib

flags.DEFINE\_string(

'model\_dir', None, 'Path to output model directory '

'where event and checkpoint files will be written.')

flags.DEFINE\_string('pipeline\_config\_path', None, 'Path to pipeline config '

'file.')

flags.DEFINE\_integer('num\_train\_steps', None, 'Number of train steps.')

flags.DEFINE\_boolean('eval\_training\_data', False,

'If training data should be evaluated for this job. Note '

'that one call only use this in eval-only mode, and '

'`checkpoint\_dir` must be supplied.')

flags.DEFINE\_integer('sample\_1\_of\_n\_eval\_examples', 1, 'Will sample one of '

'every n eval input examples, where n is provided.')

flags.DEFINE\_integer('sample\_1\_of\_n\_eval\_on\_train\_examples', 5, 'Will sample '

'one of every n train input examples for evaluation, '

'where n is provided. This is only used if '

'`eval\_training\_data` is True.')

flags.DEFINE\_string(

'hparams\_overrides', None, 'Hyperparameter overrides, '

'represented as a string containing comma-separated '

'hparam\_name=value pairs.')

flags.DEFINE\_string(

'checkpoint\_dir', None, 'Path to directory holding a checkpoint. If '

'`checkpoint\_dir` is provided, this binary operates in eval-only mode, '

'writing resulting metrics to `model\_dir`.')

flags.DEFINE\_boolean(

'run\_once', False, 'If running in eval-only mode, whether to run just '

'one round of eval vs running continuously (default).'

)

FLAGS = flags.FLAGS

def main(unused\_argv):

flags.mark\_flag\_as\_required('model\_dir')

flags.mark\_flag\_as\_required('pipeline\_config\_path')

config = tf.estimator.RunConfig(model\_dir=FLAGS.model\_dir)

train\_and\_eval\_dict = model\_lib.create\_estimator\_and\_inputs(

run\_config=config,

hparams=model\_hparams.create\_hparams(FLAGS.hparams\_overrides),

pipeline\_config\_path=FLAGS.pipeline\_config\_path,

train\_steps=FLAGS.num\_train\_steps,

sample\_1\_of\_n\_eval\_examples=FLAGS.sample\_1\_of\_n\_eval\_examples,

sample\_1\_of\_n\_eval\_on\_train\_examples=(

FLAGS.sample\_1\_of\_n\_eval\_on\_train\_examples))

estimator = train\_and\_eval\_dict['estimator']

train\_input\_fn = train\_and\_eval\_dict['train\_input\_fn']

eval\_input\_fns = train\_and\_eval\_dict['eval\_input\_fns']

eval\_on\_train\_input\_fn = train\_and\_eval\_dict['eval\_on\_train\_input\_fn']

predict\_input\_fn = train\_and\_eval\_dict['predict\_input\_fn']

train\_steps = train\_and\_eval\_dict['train\_steps']

if FLAGS.checkpoint\_dir:

if FLAGS.eval\_training\_data:

name = 'training\_data'

input\_fn = eval\_on\_train\_input\_fn

else:

name = 'validation\_data'

# The first eval input will be evaluated.

input\_fn = eval\_input\_fns[0]

if FLAGS.run\_once:

estimator.evaluate(input\_fn,

num\_eval\_steps=None,

checkpoint\_path=tf.train.latest\_checkpoint(

FLAGS.checkpoint\_dir))

else:

model\_lib.continuous\_eval(estimator, FLAGS.checkpoint\_dir, input\_fn,

train\_steps, name)

else:

train\_spec, eval\_specs = model\_lib.create\_train\_and\_eval\_specs(

train\_input\_fn,

eval\_input\_fns,

eval\_on\_train\_input\_fn,

predict\_input\_fn,

train\_steps,

eval\_on\_train\_data=False)

# Currently only a single Eval Spec is allowed.

tf.estimator.train\_and\_evaluate(estimator, train\_spec, eval\_specs[0])

if \_\_name\_\_ == '\_\_main\_\_':

tf.app.run()

# Copyright 2017 The TensorFlow Authors. All Rights Reserved.

#

# Licensed under the Apache License, Version 2.0 (the "License");

# you may not use this file except in compliance with the License.

# You may obtain a copy of the License at

#

# http://www.apache.org/licenses/LICENSE-2.0

#

# Unless required by applicable law or agreed to in writing, software

# distributed under the License is distributed on an "AS IS" BASIS,

# WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.

# See the License for the specific language governing permissions and

# limitations under the License.

# ==============================================================================

r"""Training executable for detection models.

This executable is used to train DetectionModels. There are two ways of

configuring the training job:

1) A single pipeline\_pb2.TrainEvalPipelineConfig configuration file

can be specified by --pipeline\_config\_path.

Example usage:

./train \

--logtostderr \

--train\_dir=path/to/train\_dir \

--pipeline\_config\_path=pipeline\_config.pbtxt

2) Three configuration files can be provided: a model\_pb2.DetectionModel

configuration file to define what type of DetectionModel is being trained, an

input\_reader\_pb2.InputReader file to specify what training data will be used and

a train\_pb2.TrainConfig file to configure training parameters.

Example usage:

./train \

--logtostderr \

--train\_dir=path/to/train\_dir \

--model\_config\_path=model\_config.pbtxt \

--train\_config\_path=train\_config.pbtxt \

--input\_config\_path=train\_input\_config.pbtxt

"""

import functools

import json

import os

import tensorflow as tf

from object\_detection.builders import dataset\_builder

from object\_detection.builders import graph\_rewriter\_builder

from object\_detection.builders import model\_builder

from object\_detection.legacy import trainer

from object\_detection.utils import config\_util

tf.logging.set\_verbosity(tf.logging.INFO)

flags = tf.app.flags

flags.DEFINE\_string('master', '', 'Name of the TensorFlow master to use.')

flags.DEFINE\_integer('task', 0, 'task id')

flags.DEFINE\_integer('num\_clones', 1, 'Number of clones to deploy per worker.')

flags.DEFINE\_boolean('clone\_on\_cpu', False,

'Force clones to be deployed on CPU. Note that even if '

'set to False (allowing ops to run on gpu), some ops may '

'still be run on the CPU if they have no GPU kernel.')

flags.DEFINE\_integer('worker\_replicas', 1, 'Number of worker+trainer '

'replicas.')

flags.DEFINE\_integer('ps\_tasks', 0,

'Number of parameter server tasks. If None, does not use '

'a parameter server.')

flags.DEFINE\_string('train\_dir', '',

'Directory to save the checkpoints and training summaries.')

flags.DEFINE\_string('pipeline\_config\_path', '',

'Path to a pipeline\_pb2.TrainEvalPipelineConfig config '

'file. If provided, other configs are ignored')

flags.DEFINE\_string('train\_config\_path', '',

'Path to a train\_pb2.TrainConfig config file.')

flags.DEFINE\_string('input\_config\_path', '',

'Path to an input\_reader\_pb2.InputReader config file.')

flags.DEFINE\_string('model\_config\_path', '',

'Path to a model\_pb2.DetectionModel config file.')

FLAGS = flags.FLAGS

@tf.contrib.framework.deprecated(None, 'Use object\_detection/model\_main.py.')

def main(\_):

assert FLAGS.train\_dir, '`train\_dir` is missing.'

if FLAGS.task == 0: tf.gfile.MakeDirs(FLAGS.train\_dir)

if FLAGS.pipeline\_config\_path:

configs = config\_util.get\_configs\_from\_pipeline\_file(

FLAGS.pipeline\_config\_path)

if FLAGS.task == 0:

tf.gfile.Copy(FLAGS.pipeline\_config\_path,

os.path.join(FLAGS.train\_dir, 'pipeline.config'),

overwrite=True)

else:

configs = config\_util.get\_configs\_from\_multiple\_files(

model\_config\_path=FLAGS.model\_config\_path,

train\_config\_path=FLAGS.train\_config\_path,

train\_input\_config\_path=FLAGS.input\_config\_path)

if FLAGS.task == 0:

for name, config in [('model.config', FLAGS.model\_config\_path),

('train.config', FLAGS.train\_config\_path),

('input.config', FLAGS.input\_config\_path)]:

tf.gfile.Copy(config, os.path.join(FLAGS.train\_dir, name),

overwrite=True)

model\_config = configs['model']

train\_config = configs['train\_config']

input\_config = configs['train\_input\_config']

model\_fn = functools.partial(

model\_builder.build,

model\_config=model\_config,

is\_training=True)

def get\_next(config):

return dataset\_builder.make\_initializable\_iterator(

dataset\_builder.build(config)).get\_next()

create\_input\_dict\_fn = functools.partial(get\_next, input\_config)

env = json.loads(os.environ.get('TF\_CONFIG', '{}'))

cluster\_data = env.get('cluster', None)

cluster = tf.train.ClusterSpec(cluster\_data) if cluster\_data else None

task\_data = env.get('task', None) or {'type': 'master', 'index': 0}

task\_info = type('TaskSpec', (object,), task\_data)

# Parameters for a single worker.

ps\_tasks = 0

worker\_replicas = 1

worker\_job\_name = 'lonely\_worker'

task = 0

is\_chief = True

master = ''

if cluster\_data and 'worker' in cluster\_data:

# Number of total worker replicas include "worker"s and the "master".

worker\_replicas = len(cluster\_data['worker']) + 1

if cluster\_data and 'ps' in cluster\_data:

ps\_tasks = len(cluster\_data['ps'])

if worker\_replicas > 1 and ps\_tasks < 1:

raise ValueError('At least 1 ps task is needed for distributed training.')

if worker\_replicas >= 1 and ps\_tasks > 0:

# Set up distributed training.

server = tf.train.Server(tf.train.ClusterSpec(cluster), protocol='grpc',

job\_name=task\_info.type,

task\_index=task\_info.index)

if task\_info.type == 'ps':

server.join()

return

worker\_job\_name = '%s/task:%d' % (task\_info.type, task\_info.index)

task = task\_info.index

is\_chief = (task\_info.type == 'master')

master = server.target

graph\_rewriter\_fn = None

if 'graph\_rewriter\_config' in configs:

graph\_rewriter\_fn = graph\_rewriter\_builder.build(

configs['graph\_rewriter\_config'], is\_training=True)

trainer.train(

create\_input\_dict\_fn,

model\_fn,

train\_config,

master,

task,

FLAGS.num\_clones,

worker\_replicas,

FLAGS.clone\_on\_cpu,

ps\_tasks,

worker\_job\_name,

is\_chief,

FLAGS.train\_dir,

graph\_hook\_fn=graph\_rewriter\_fn)

if \_\_name\_\_ == '\_\_main\_\_':

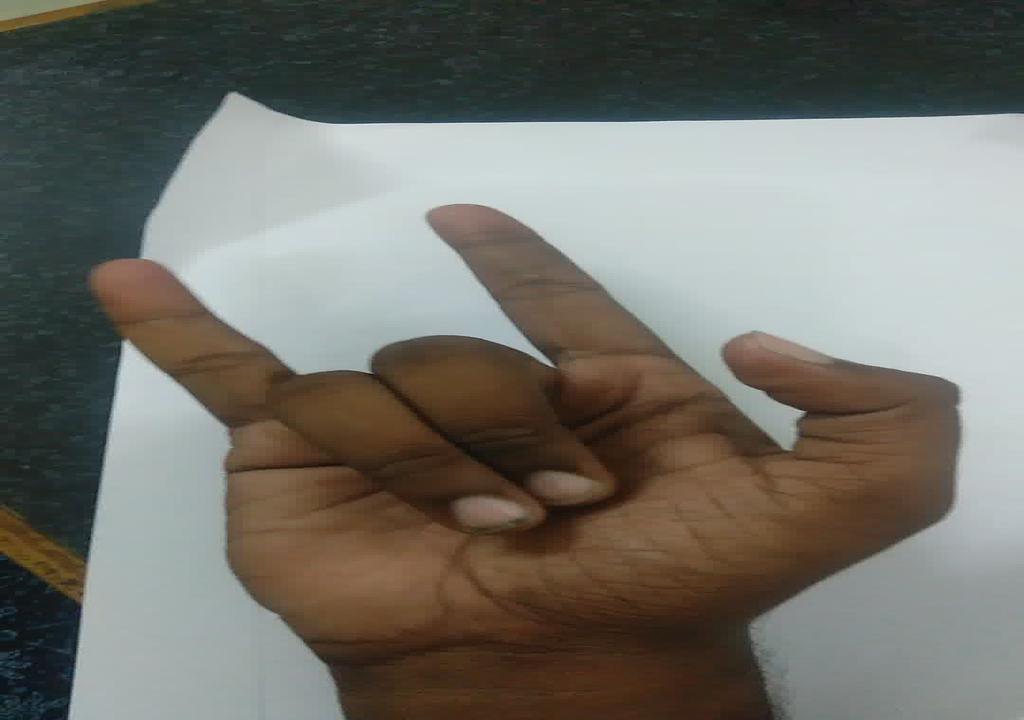
tf.app.run()

**A.2 SCREEN LAYOUT**

**Sign Language detector Hello**

****

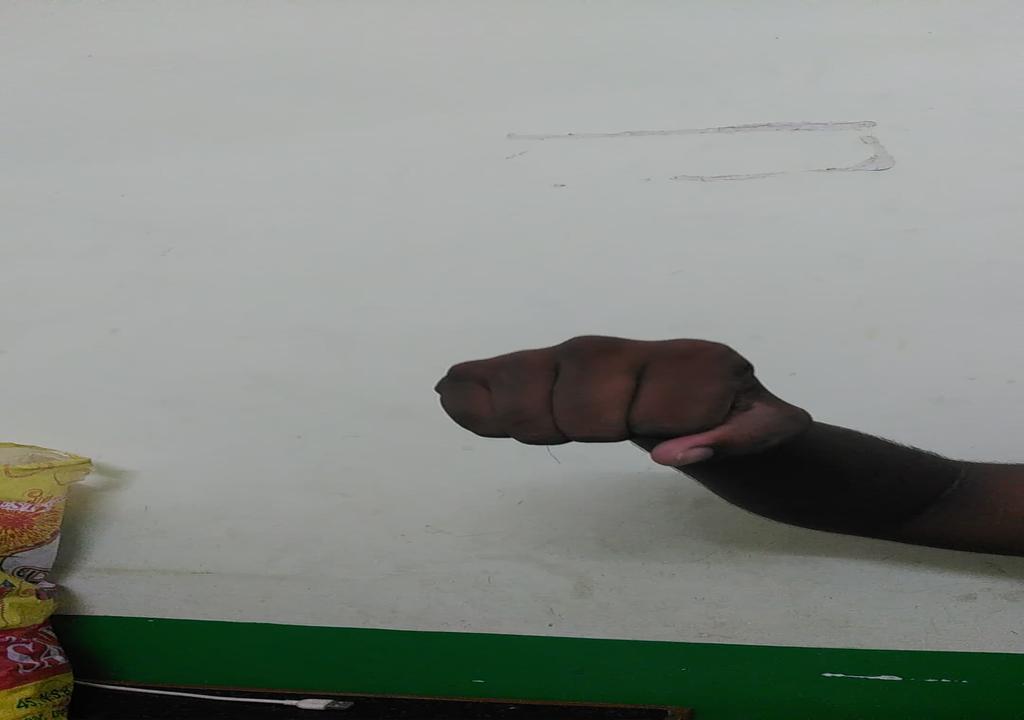
**Sign Language detector Please**

****

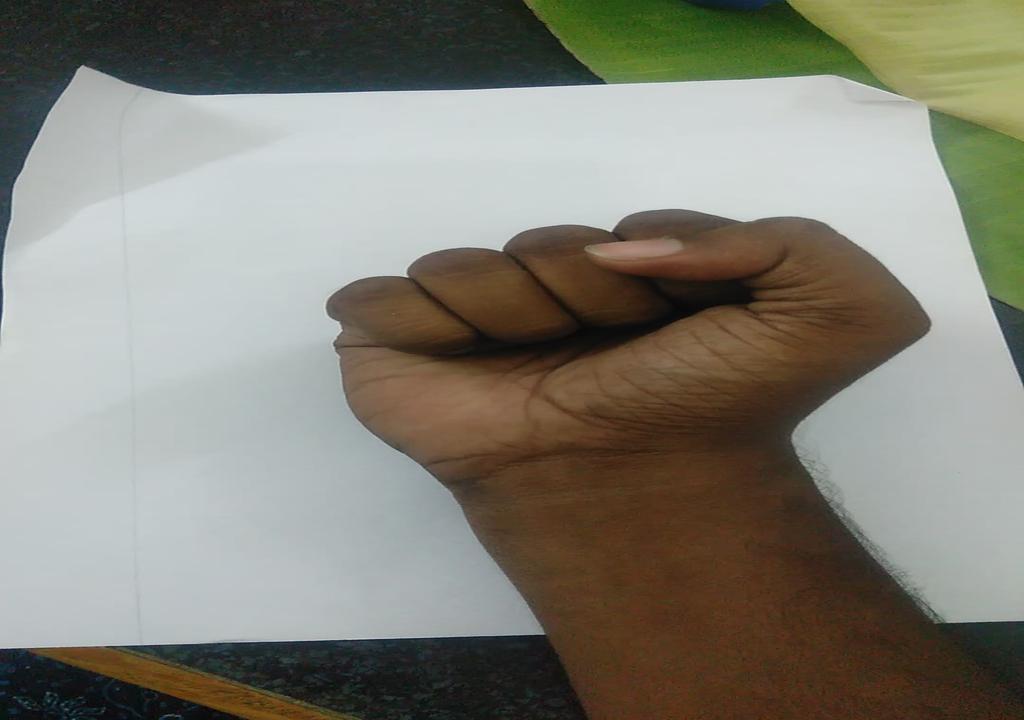
**Sign Language detector Love You**

****

**Sign Language detector Yes**

****

**Sign Language detector No**

****