

Sign Language Detection: An add on feature over Video-conferencing Platforms

A Project-II Report

Submitted in partial fulfillment of requirement of the

Degree of

**BACHELOR OF TECHNOLOGY in COMPUTER
SCIENCE & ENGINEERING**

BY

Sharvari Mishra EN18CS301239

Shruti Gupta EN18CS30150

Simi Jain EN18CS301262

Under the Guidance of

Prof. Preetesh Purohit



Department of Computer Science & Engineering

Faculty of Engineering

MEDI-CAPS UNIVERSITY, INDORE- 453331

May 2022

Sign Language Detection: An add on feature over Video-conferencing Platforms

A Project-II Report

Submitted in partial fulfillment of requirement of the
Degree of

**BACHELOR OF TECHNOLOGY in COMPUTER
SCIENCE & ENGINEERING**

BY

Sharvari Mishra EN18CS301239

Shruti Gupta EN18CS301250

Simi Jain EN18CS301262

Under the Guidance of
Prof. Preetesh Purohit



**Department of Computer Science & Engineering
Faculty of Engineering
MEDI-CAPS UNIVERSITY, INDORE- 453331**

May 2022

Report Approval

The project work “**Sign Language Detection: An add on feature over Video-conferencing Platforms**” is hereby approved as a creditable study of an engineering/computer application subject carried out and presented in a manner satisfactory to warrant its acceptance as prerequisite for the Degree for which it has been submitted.

It is to be understood that by this approval the undersigned do not endorse or approve any statement made, opinion expressed, or conclusion drawn there in; but approve the “Project Report ” only for the purpose for which it has been submitted.

Internal Examiner

Name:

Designation

Affiliation

External Examiner

Name:

Designation

Affiliation

Declaration

We hereby declare that the project entitled “**Sign Language Detection: An add on feature over Video-conferencing Platforms**” submitted in partial fulfillment for the award of the degree of Bachelor of Technology ‘Computer Science and Engineering’ completed under the supervision of **Prof. Preetesh Purohit, Department of Computer Science and Engineering**, Faculty of Engineering, Medi-Caps University Indore is an authentic work.

Further, we declare that the content of this Project work, in full or in parts, have neither been taken from any other source nor have been submitted to any other Institute or University for the award of any degree or diploma.

19/05/2022

Sharvari Mishra

19/05/2022

Shruti Gupta

19/05/2022

Simi Jain

Certificate

I, **Preetesh Purohit** certify that the project entitled **Sign Language Detection: An add on feature over Video-conferencing Platforms** submitted in partial fulfillment for the award of the degree of Bachelor of Technology of Computer Applications by **Sharvari Mishra, Shruti Gupta, Simi Jain** is the record carried out by them under my guidance and that the work has not formed the basis of award of any other degree elsewhere.

Prof. Preetesh Purohit
Department of Computer Science Engineering
Medi-Caps University, Indore

Dr. Pramod S. Nair
Head of the Department
Computer Science & Engineering
Medi-Caps University, Indore

Acknowledgements

We would like to express my deepest gratitude to Honorable Chancellor, **Shri R C Mittal**, who has provided us with every facility to successfully carry out this project, and our profound indebtedness to **Prof. (Dr.) Dilip K. Patnaik**, Vice Chancellor, Medi-Caps University, whose unfailing support and enthusiasm has always boosted up our morale. We also thank **Prof. (Dr.) D K Panda**, Pro Vice Chancellor, **Dr. Suresh Jain**, Dean Faculty of Engineering, Medi-Caps University, for giving us a chance to work on this project. We would also like to thank our Head of the Department **Dr. Pramod S. Nair** for his continuous encouragement for betterment of the project.

We express our heartfelt gratitude to our Internal Guide Prof. Preetesh Purohit and Project Coordinator Asst. Prof. Prasanna kapse without whose continuous help and support, this project would ever have reached to the completion.

Sharvari Mishra

Shruti Gupta

Simi Jain

B.Tech. IV Year

Department of Computer Science & Engineering

Faculty of Engineering

Medi-Caps University, Indore

Executive Summary

Online meeting platforms are getting more and more inclusive. As a result, we can connect with more communities easier than ever. And with video conferencing becoming the new normal, Sign Language users are having a hard time keeping up with the sudden increase in use of such services. Whether it's a business meeting or an academic lecture, hiring an interpreter may be a costly and time-consuming endeavor. Miscommunication and information leakage are also possibilities, which cannot be afforded for a high-profile conference.

The virtualization of meetings promotes collaboration among colleagues, target markets, associates, teachers, and students from different countries and race. Each region has its own interpretations of a particular gesture. There are somewhere between 138 and 300 different types of sign language used throughout the world today. New sign languages frequently evolve amongst groups of deaf children and adults. It's next to impossible for an individual to know every variation of sign language. The various types of sign language is again adding a limitation for hiring a manual sign interpreter.

Considering the above limitations, a feature is required that identifies various signs and gestures and delivers their meaning to ordinary people over video conferencing platforms. This will not only bridge the communication barrier but also help these specially abled people to come forward and showcase their full potential without depending on a manual interpreter.

Table of Contents

| | Page No. |
|---|-----------------|
| Report Approval | ii |
| Declaration | iii |
| Certificate | iv |
| Acknowledgement | v |
| Executive Summary | vi |
| Table of Contents | vii |
| List of figures | viii |
| List of tables | ix |
| Abbreviations | x |
| Notations & Symbols | xi |
| Chapter 1 Introduction | |
| 1.1 Introduction | 1 |
| 1.2 Literature Review | 1 |
| 1.3 Objectives | 3 |
| 1.4 Significance | 3 |
| 1.5 Research Design | 4 |
| 1.6 Source of Data | 5 |
| 1.7 Chapter Scheme | 5 |
| Chapter 2 Report on Present Investigation | |
| 2.1 Experimental Set-up | 6 |
| Chapter 3 Main Chapter | |
| 3.1 Technologies/Algorithms Used | 12 |
| 3.2 Development Tool | 21 |
| 3.3 Working of our Project | 25 |
| 3.4 Testing | 26 |
| Chapter 4 Results and Discussions | 30 |
| Chapter 5 Summary and Conclusions | 33 |
| Chapter 6 Future scope | 34 |
| Bibliography | 35 |
| List of Publications | 37 |

List of Figures

| Sr. No | Figure | Page No. |
|--------|---|----------|
| 1 | Hello | 7 |
| 2 | Yes | 7 |
| 3 | No | 7 |
| 4 | See You | 7 |
| 5 | Thank You | 8 |
| 6 | Labelled Image and it's Xml file | 9 |
| 7 | Neural network | 14 |
| 8 | Neural network with many convolutional layers | 15 |
| 9 | SSD_v2_mobilenet architecture | 16 |
| 10 | Command for starting the Development (React) server | 21 |
| 11 | Interface of visual studio code | 23 |
| 12 | Image Annotation of Hello sign | 24 |
| 13 | XML file of Hello sign | 25 |
| 14 | Workflow of Sign Language System | 26 |
| 15 | Result of Hello | 31 |
| 16 | Result of No | 31 |
| 17 | Result of yes | 31 |
| 18 | Result of Thank You | 31 |
| 19 | Result of See You | 32 |

List of Tables

| Sr.No | Table | Page No. |
|-------|----------------------|----------|
| 1. | Confusion Matrix | 10 |
| 2. | Experimental Results | 29 |
| 3. | Test Cases | 30 |

Abbreviations

| | |
|--------|------------------------------|
| AI | Artificial Intelligence |
| ASL | American Sign Language |
| CNN | Convolutional Neural Network |
| GUI | Graphical User Interface |
| ML | Machine Learning |
| NPM | Node Package Manager |
| OpenCV | Open-Source Computer Vision |
| PCA | Principal Component Analysis |
| SSD | Single Short Detector |
| VOC | Visual Object Classes |
| YOLO | You Only Look Once |
| XML | Ex-tensible Markup Language |

Chapter-1

1.1 Introduction

In this era, where inclusivity and sense of equality are considered a basic necessity for all. But this gets challenged when an individual who is fluent in sign and uses it to communicate with someone who isn't. We often take our ability to interact with others for granted. More than 70 million around the world uses sign language. But significant barriers to communicating in sign language are depriving these specially-abled people to enjoy even basic interaction with others. Although deaf and mute people can settle everything in writing, but there might be such situations where the corporation of manual sign language interpreters becomes a necessity. Hiring a translator can be exorbitant. The objective of this project is to tear down this communication barrier a bit without having to learn sign explicitly or need an interpreter.

With the recent rise of video conferencing platforms, we identify the problem of signers not “getting the floor” when communicating, which either leads to them being ignored or to a cognitive load on other participants, always checking to see if someone starts signing[1].

A Sign Language Detection System can be thought as a must-have feature for every video conferencing platform which comes into contact with people, who are hard-of-hearing or are not able to speak, on a regular basis. The proposed system revolves around the idea of a camera-based sign language recognition system that will provide the users with an interactive tool, capable of detecting hand gestures in real-time along with providing the percentage accuracy and further translating into text. This text is rendered as a header over the detected frame. The Model is trained over the input dataset using a neural network, provided for a particular sign, and further uses this training to detect the gestures in real-time.

1.2 Literature Review

The detection of hand gestures with more precision is the key area for many researchers. Different techniques and algorithms, such as glove-based method, sensor-based approach, computing approaches such as neural network, fuzzy logic have been used to provide an easy-to-use environment for the detection of sign language.

Authors S. Shrenika and M. Madhu Bala[2] implemented the sign language recognition system based on the image processing technique. This system recognizes the ASL sign alphabets using an edge detection algorithm. It also includes the removal of noise by smoothing algorithm. As a result, it will display the text meaning of character alphabet, but this system is limited only to the detection of one hand.

Zhen Zhan et. al. in[3] , used a MYO armband for data collecting and Neural Network for processing and detecting Chinese Sign Language (CSL). The latency was good, but one had to wear armband all the time.

R. Bhadra and S. Kar proposed Sign Language Detection System based on Deep Multi-layered Convolution Neural Network. The system detects static and dynamic hand gestures with an accuracy of 99.89% but the system does not apply to real-time detection of sign gestures[4] .

Joyeeta Singha et. al.[5] developed the system using an Eigenvector-based image processing algorithm to recognize several Indian sign language signs for live video sequences with 96.25% accuracy. This system eliminated the difficulty of glove-based approach, but it is limited only to recognizing ISL alphabets.

In glove-based method (Wang and Popovic[6]), the sensors in the gloves can detect the movement of hands and pass the information to the computer. This approach has high accuracy in gesture recognition but is quite expensive and inconvenient to the user.

Same approach was used by Deora and Bajaj[7] for recognizing ISL alphabets and numerals. For easy hand segmentation, the signer must wear a red and blue coloured glove while collecting data. And it's all done with Principal Component Analysis (PCA). The recognition rate is 94 percent. However, the system only recognises static motions and ignores signs in which both hands overlap.

Flex sensors also came into the picture, which were developed for mute audiences researched in Irget.net[8] , in which the user's hand is attached with the flex sensors. The Flex sensor responds individually to the bend of each finger. Taking this value, the controller starts to react with speech, each flex sensor holds unique voice stored in the APR Kit and for each sign it will play a unique voice. The work is done only for certain alphabets and not for the words or phrases, and the obtained accuracy is very low.

Fariha Nasir et. al. in [9], used Kinect to capture the 3D video stream and the joints of interest in the human skeleton. It had 97% accuracy of words detection but it does not recognize differences at finger level and also gestures performed in forward or backward sequence.

T. Aadit, J. Deepak, P. Janhavi, and D. Jyoti[10] implemented a tool to recognize hand gestures for Indian Sign Language using the Convolution Neural Network (CNN) Deep Learning algorithm for mute people. This system gives the user a frontend that can be used for daily purposes. Therefore, this system can further improve the success rate by implementing sentences and phrases for better user experience.

D. Talukder and F. Jahara proposed a system capable of detecting Bangla sign language for deaf and mute communities using YOLOv4 as an object detection model and generate both textual sentences and speech in real-time. It also proposed three new signs for the task of sentencing generation. In addition, this system will be more useful if it is implemented in several other sign languages[11].

But as far as video conferencing and video meetings are concerned there is not much that has been published so far. Recently, Kelly Earley - a former journalist said that Google is developing sign language detection for videoconferencing. Amit Moryossef, an intern at Google Research, recently published a blog post describing some of the work that the Google has been doing to make videoconferencing technology more accessible to people communicating through sign language.

According to the literature survey, most of the above focused on improving accuracy and correcting gestures, but we are trying to add sign language recognition feature on video conferencing platforms, also Google is working on it. Video conferencing applications do not have a mechanism for detecting sign language and we are therefore trying to do so. It can help these specially-abled people to communicate easily and effectively.

1.3 Objectives

Although sign language is the most natural way of exchanging information among deaf people it has been observed that they are facing difficulties with normal people interaction. This project is being built keeping in mind the main target audience, which is the deaf and the mute population. This feature will be extremely user friendly and uncomplicated to ensure maximum efficiency and participation.

Focusing on the main objective that is “**Sign language Gestures to text translation.**” The translation will be in real time, it will be easy to establish a normal conversational flow which will bring in a sense of inclusion and acceptance as well as normalcy. The purpose of this project is to use a vision-based system for sign language recognition and at the same time, select hand features that could be used with machine learning algorithms allowing the application to detect in real-time. For that, the user must be positioned in front of the camera, doing the sign language gestures, that will be interpreted by the system and their classification will be displayed on the top side of the detected frame. The only essential requirement is a working web camera, which is needed to detect the hand gestures, which is readily available to almost everyone in this age.

The application will be robust to ensure that it can handle any exceptions and errors thrown in its way without disrupting it's working.

1.4 Significance

Communication is an essential tool in human existence. It is a fundamental and effective way of sharing thoughts, feelings and opinions. However, a substantial fraction of the world's population lacks this ability. The latest report of WHO revealed that 466 million people were suffering from hearing loss in 2019, which amount to 5% of the world population with 432 million (or 83%) of them being adults, and 34 million (17%) of them are children

The WHO also estimated that the number would double (i.e. 900 million people) by 2050. In these fast-growing deaf-mute people, there is a need to break the communication barrier that adversely affects the lives and social relationships of deaf-mute people. However, the challenges of employing interpreters are the flexible structure of sign languages combined with insufficient numbers of expert sign language interpreters across the globe. Therefore, there is a need for a technology-based system that can complement conventional sign language interpreters.

Sign Language Detection feature embedded over video conferencing platforms can aid this gap a little and will provide a communication ease to sign language user. No Need of a manual interpreter neither a need of learning sign language explicitly.

This feature will sign allows these specially abled people to voice opinions as easily as a normal person and will give them a sense of inclusivity in society.

1.5 Research Design

People seem unaware of what being mute entails, we guess that's because in a world where the majority of people can speak it's not something most people can really emphasize with. The hardest part for them is how some people assume that having a disability means that these people are not capable enough to study or work. Therefore, they are considered as aloof and are always left out. They are still a person; they just can't speak but are as capable as a normal person.

Rather than sympathizing over their situation, easing down their communication struggle is least we can do and this is what they would expect from us.

We are all witnessing how technology is fundamentally changing the way we live, work and relate to one another and to the external world. Now more than ever, the advent of AI technology has the potential to transform one's life. Why not use the emerging ML technology to tear down the struggles of mute people.

Just like how we use to match pictures with their respective names in our kindergarten classes. Using the same concept, we thought of matching the signs to their respective meaning, but making it more accessible by detecting the gestures in real time and then displaying their correct label. And thereby proposing this sign language recognition feature to be scaled over video conferencing platform. This will increase the usability as well as participation of mute population.

1.6 Source of Data

1.6.1 Data Collection

We created and collected our dataset according to our requirement. Since this is just a demo project to showcase its usage possibility over video conferencing platforms. We trained our model over just few signs.

We collected 16 images of each gesture ensuring that the captured dataset covers different angles and orientation of each sign to achieve a higher accuracy.

1.7 Chapter Scheme

Communication is an essential tool in human existence. It is a fundamental and effective way of sharing thoughts, feelings and opinions. However, a substantial fraction of the world's population lacks this ability. The latest report of WHO revealed that 466 million people were suffering from hearing loss in 2019, which amount to 5% of the world population with 432 million (or 83%) of them being adults, and 34 million (17%) of them are children.

The WHO also estimated that the number would double (i.e. 900 million people) by 2050. In these fast-growing deaf-mute people, there is a need to break the communication barrier that adversely affects the lives and social relationships of deaf-mute people. However, the challenges of employing interpreters are the flexible structure of sign languages combined with insufficient numbers of expert sign language interpreters across the globe. Therefore, there is a need for a technology-based system that can complement conventional sign language interpreters.

A Sign Language Detection System can be thought as a must-have feature for every video conferencing platform which comes into contact with people, who are hard-of-hearing or are not able to speak, on a regular basis. The proposed system revolves around the idea of a camera-based sign language recognition system that will provide the users with an interactive tool, capable of detecting hand gestures in real-time along with providing the percentage accuracy and further translating into text. This text is rendered as a header over the detected frame. The Model is trained over the input dataset using a neural network, provided for a particular sign, and further uses this training to detect the gestures in real-time.

Chapter 2

Report on Present Investigation

2.1 Experimental Set-up

For the experimental setup, we are focusing on few signs for our demo application. To accomplish this, we constructed a dataset by gathering 16 images of each selected sign using Python and OpenCV, an open-source computer vision and machine learning package. To label our dataset, we utilized an open-source annotation tool called LabelImg, which generates xml files including information about the orientation of each labelled image. Finally, we trained a Convolutional Neural Network (CNN) using our labelled dataset. We used the SSD mobilenet_v2_fpn-lite 320x320 coco17 tpu-8 object detection model to recognize real-time hand gestures.

2.2 Procedures adopted

Sign language recognition has been widely studied across different domains and sign languages. As sign language corpora are usually small[12], previous works take one of two approaches to reduce the network's parameters: (1) using pose estimation on the original videos[1], [9] or (2) using pre-trained CNNs to get a feature vector per frame[3], [4], [10], [11]. We used the second approach, i.e., a pre-trained neural convolution network (CNN) to implement our demonstration application due to its high precision for image classification and feature extraction.

2.2.1 Creating Dataset

Instead of working over all the signs of ASL, we selected few signs for implementing purpose. Then collected 16 pictures of each selected sign through python OpenCV, which is huge open-source library for computer vision and machine learning as shown Fig.1 Collected images were taken considering different positions of and direction to increase the detection accuracy.



Figure 1: Hello



Figure 2: Yes

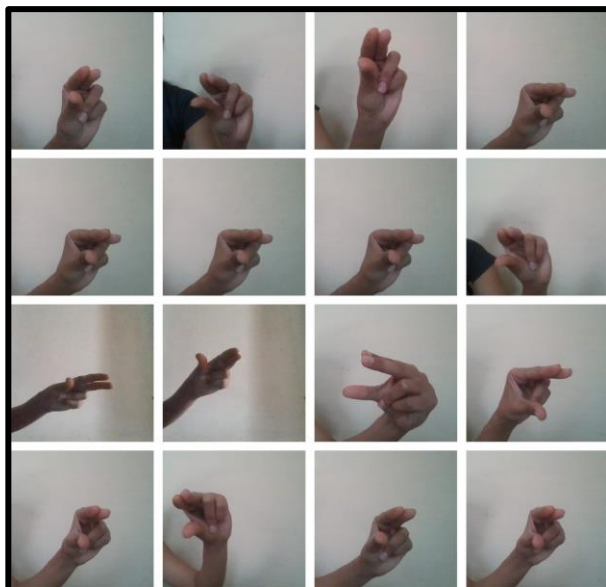


Figure 3: No

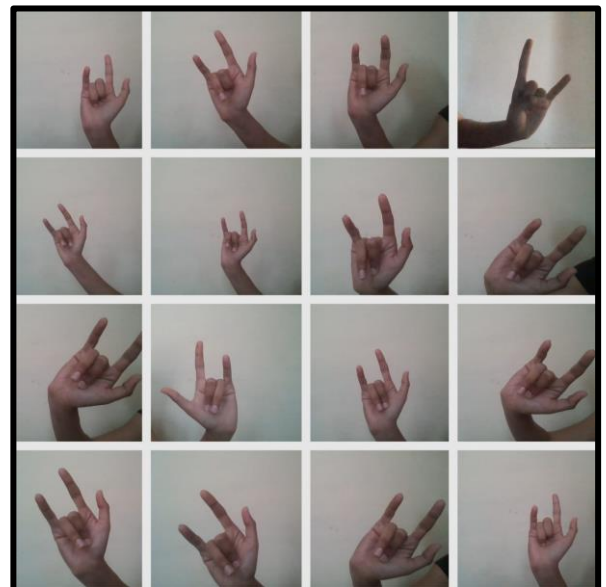


Figure 4: See You

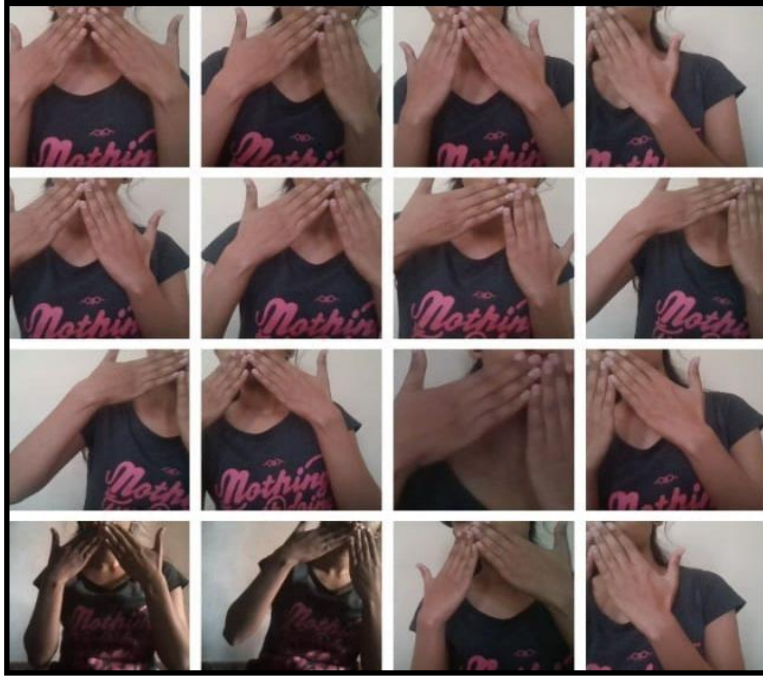
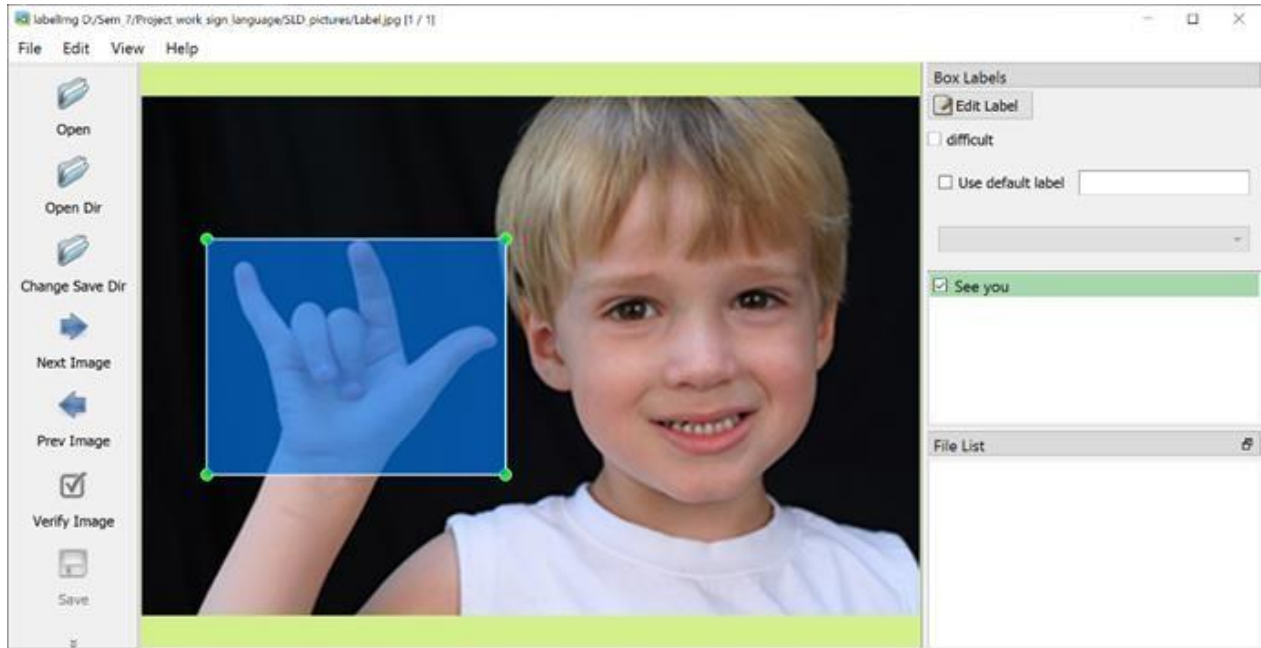


Figure 5: Thank you

2.2.2 Labelling Dataset

Image annotation is a necessary step in supervised learning to make our dataset a useful component for training our machine learning model. Labelling maps the dataset to its specific output that we expect as our detection result. We labelled our images with LabelImg, an open-source annotation tool. It creates an xml file that contains the orientation information for each labelled part of the image, as shown in fig.2.



```
<?xml version="1.0"?>
- <annotation>
  <folder>SLD_pictures</folder>
  <filename>Label.jpg</filename>
  <path>D:\Sem_7\Project_work_sign_language\SLD_pictures\Label.jpg</path>
  - <source>
    <database>Unknown</database>
  </source>
  - <size>
    <width>370</width>
    <height>247</height>
    <depth>3</depth>
  </size>
  <segmented>0</segmented>
</annotation>
```

Figure 6: Labelled Image and it's Xml file

2.2.3) Training Dataset

We trained our labelled dataset using Convolution Neural Network (CNN) for detecting hand gestures in real-time. Instead of building and training CNN from scratch we used a pre trained TensorFlow object detection Model. Through transfer learning, we trained the pre-trained TensorFlow object detection model Zoo to our smaller and specific purpose dataset for 5000 steps.

We used `ssd_mobilenet_v2_fpnlite_320x320_coco17_tpu-8` architecture as our training model. SSD is simple relative to methods that require object proposals because it completely eliminates proposal generation and subsequent pixel or feature resampling stages and encapsulates all computation in a single network. This makes SSD easy to train and straightforward to integrate into systems that require a detection component. It has a small convolutional filter to predict object categories and offsets in bounding box locations. It uses separate predictors (filters) for different aspect ratio detections, and applying these filters to multiple feature maps from the later stages of a network in order to perform detection at multiple scales.

2.2.4) Calculating Accuracy

We used confusion matrix to calculate of each detected result. A Confusion matrix is an $N \times N$ matrix used for evaluating the performance of a classification model, where N is the number of target classes. The matrix compares the actual target values with those predicted by the machine learning model. This gives us a holistic view of how well our classification model is performing and what kinds of errors it is making.

For a binary classification problem, we would have a 2×2 matrix as shown below with 4 values:

Table 1: Confusion Matrix

| | Actual Value: True/False | |
|---------------------------------------|--------------------------|----------------|
| Predicted Value: Positive/Negative | True Positive | False Positive |
| | False Negative | True Negative |

True Positive (TP)

- The predicted value matches the actual value
- The actual value was positive and the model predicted a positive value

True Negative (TN)

- The predicted value matches the actual value
- The actual value was negative and the model predicted a negative value

False Positive (FP) – Type 1 error

- The predicted value was falsely predicted
- The actual value was negative but the model predicted a positive value
- Also known as the **Type 1 error**

False Negative (FN) – Type 2 error

- The predicted value was falsely predicted
- The actual value was positive but the model predicted a negative value
- Also known as the **Type 2 error**

Using the confusion matrix values, Accuracy is calculated as in Eq.1.

$$Accuracy = \frac{TP+TN}{TP+FP+TN+FN}$$

Equation 1: Calculating Accuracy

To find how many results are correctly identified. We used the Concept of Precision and Recall.

Precision tells us how many of the correctly predicted cases actually turned out to be positive.

Here's how to calculate Precision in Eq.2.

$$Precision = \frac{TP}{TP + FP}$$

Equation 2: Calculating Precision

This would determine whether our model is reliable or not.

Recall tells us how many of the actual positive cases we were able to predict correctly with our model. And here's how we can calculate Recall in Eq.3.

$$Recall = \frac{TP}{TP + FN}$$

Equation 3: Calculating Recall

Chapter-3

Main Chapter

The domain analysis that we have done for the project mainly involved understanding the neural networks.

3.1 Technologies / Algorithms Used

3.1.1 Machine Learning

Machine learning is a branch of artificial intelligence (AI) and computer science which focuses on the use of data and algorithms to imitate the way that humans learn, gradually improving its accuracy.

IBM has a rich history with machine learning. One of its own, Arthur Samuel, is credited for coining the term, “machine learning” with his research (PDF, 481 KB) (link resides outside IBM) around the game of checkers. Robert Nealey, the self-proclaimed checkers master, played the game on an IBM 7094 computer in 1962, and he lost to the computer. Compared to what can be done today, this feat almost seems trivial, but it’s considered a major milestone within the field of artificial intelligence. Over the next couple of decades, the technological developments around storage and processing power will enable some innovative products that we know and love today, such as Netflix’s recommendation engine or self-driving cars.

Machine learning is an important component of the growing field of data science. Through the use of statistical methods, algorithms are trained to make classifications or predictions, uncovering key insights within data mining projects. These insights subsequently drive decision making within applications and businesses, ideally impacting key growth metrics. As big data continues to expand and grow, the market demand for data scientists will increase, requiring them to assist in the identification of the most relevant business questions and subsequently the data to answer them.

Machine learning has developed based on the ability to use computers to probe the data for structure, even if we do not have a theory of what that structure looks like. The test for a machine learning model is a validation error on new data, not a theoretical test that proves a null hypothesis. Because machine learning often uses an iterative approach to learn from data, the learning can be easily automated. Passes are run through the data until a robust pattern is found.

A number of studies suggest of machine learning in terms of how an algorithm learns to be more accurate at making predictions. There are four main ways to learn: supervised,

unsupervised, semi-supervised, and reinforcement learning. When we did our project, we learned more about supervised learning.

3.1.2 Supervised Learning

Our Model is being trained by using a supervised learning dataset. Supervised learning, also known as supervised machine learning, is a subcategory of machine learning and artificial intelligence. It is defined by its use of labeled datasets to train algorithms that to classify data or predict outcomes accurately. As input data is fed into the model, it adjusts its weights until the model has been fitted appropriately, which occurs as part of the cross-validation process. Supervised learning helps organizations solve for a variety of real-world problems at scale, such as classifying spam in a separate folder from your inbox.

Supervised learning uses a training set to teach models to yield the desired output. This training dataset includes inputs and correct outputs, which allow the model to learn over time. The algorithm measures its accuracy through the loss function, adjusting until the error has been sufficiently minimized.

Supervised learning uses a training set to teach models to yield the desired output. This training dataset includes inputs and correct outputs, which allow the model to learn over time. The algorithm measures its accuracy through the loss function, adjusting until the error has been sufficiently minimized.

Supervised learning can be separated into two types of problems when data mining—classification and regression:

- Classification uses an algorithm to accurately assign test data into specific categories. It recognizes specific entities within the dataset and attempts to draw some conclusions on how those entities should be labeled or defined. Common classification algorithms are linear classifiers, support vector machines (SVM), decision trees, k-nearest neighbor, and random forest, which are described in more detail below.
- Regression is used to understand the relationship between dependent and independent variables. It is commonly used to make projections, such as for sales revenue for a given business. Linear regression, logistical regression, and polynomial regression are popular regression algorithms.

In supervised machine learning processes, a variety of algorithms and computation approaches are used. The following sections provide concise explanations of neural network-based learning algorithms that are commonly calculated using a programming language, such as Python.

3.1.3 Neural Networking

A neural network is a series of algorithms that endeavors to recognize underlying relationships in a set of data through a process that mimics the way the human brain operates. In this sense, neural networks refer to systems of neurons, either organic or artificial in nature. Neural networks can adapt to changing input; so the network generates the best possible result without needing to redesign the output criteria. The concept of neural networks, which has its roots in artificial intelligence, is swiftly gaining popularity in the development of trading systems.

A neural network works similarly to the human brain's neural network. A "neuron" in a neural network is a mathematical function that collects and classifies information according to a specific architecture. The network bears a strong resemblance to statistical methods such as curve fitting and regression analysis.

A neural network contains layers of interconnected nodes. Each node is a perceptron and is similar to a multiple linear regression. The perceptron feeds the signal produced by a multiple linear regression into an activation function that may be nonlinear.

In a multi-layered perceptron (MLP), perceptrons are arranged in -interconnected layers. The input layer collects input patterns. The output layer has classifications or output signals to which input patterns may map. Hidden layers fine-tune the input weightings until the neural network's margin of error is minimal. It is hypothesized that hidden layers extrapolate salient features in the input data that have predictive power regarding the outputs. This describes feature extraction, which accomplishes a utility similar to statistical techniques such as principal component analysis.

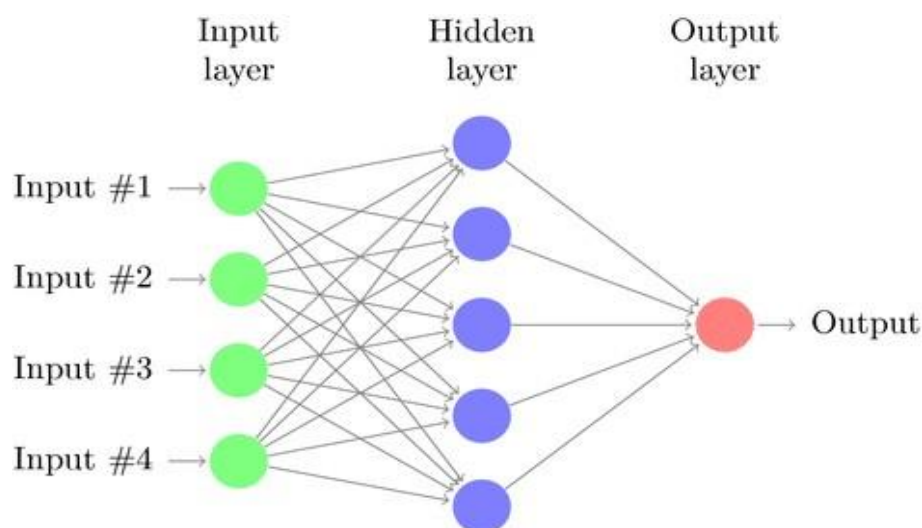


Figure 7: Neural Network

3.1.4 Convolutional Neural Network (CNN)

In neural networks, Convolutional neural network (ConvNets or CNNs) is one of the main categories to do images recognition, images classifications. Objects detections, recognition faces etc., are some of the areas where CNNs are widely used.

Convolutional neural networks (CNN/ConvNet) are a subclass of deep neural networks that are most frequently used for visual image analysis. A Convolutional Neural Network (ConvNet/CNN) is a type of Deep Learning algorithm that can take in an image as input, assign importance (learnable weights and biases) to various aspects/objects in the image, and distinguish between them.

Convolutional neural networks (CNN) are a type of artificial neural network design that was proposed by Yann LeCun in 1988. CNN makes use of some visual cortex characteristics. Image classification is one of the most common applications of this architecture. For instance, Facebook utilises CNN's automatic tagging algorithms, Amazon use CNN to provide product recommendations, and Google utilises CNN to search through users' photos.

CNN image classifications take an input image, process it and classify it under certain categories (Eg., Dog, Cat, Tiger, Lion). Computers sees an input image as array of pixels and it depends on the image resolution. Based on the image resolution, it will see $h \times w \times d$ (h = Height, w = Width, d = Dimension). Eg., An image of $6 \times 6 \times 3$ array of matrix of RGB (3 refers to RGB values) and an image of $4 \times 4 \times 1$ array of matrix of grayscale image.

Technically, deep learning CNN models to train and test, each input image will pass it through a series of convolution layers with filters (Kernels), Pooling, fully connected layers (FC) and apply Softmax function to classify an object with probabilistic values between 0 and 1. The below figure is a complete flow of CNN to process an input image and classifies the objects based on values.

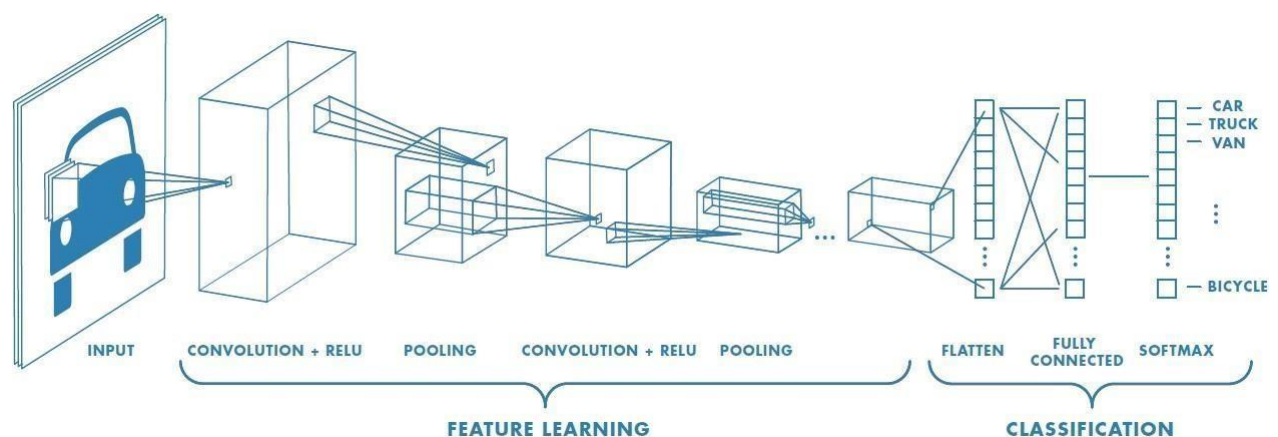


Figure 8: Neural network with many convolutional layers

Applications of convolution neural network:

- Decoding Facial Recognition: Facial recognition is broken down by a convolutional neural network into the following major components - Identifying every face in the picture focusing on each face despite external factors, such as light, angle, pose, etc. Identifying unique features.
- Comparing all the collected data with already existing data in the database to match a face with a name.
- A similar process is followed for scene labeling as well.
- Analyzing Documents: Convolutional neural networks can also be used for document analysis. This is not just useful for handwriting analysis, but also has a major stake in recognizers. For a machine to be able to scan an individual's writing, and then compare that to the wide database it has, it must execute almost a million commands a minute. It is said with the use of CNNs and newer models and algorithms, the error rate has been brought down to a minimum of 0.4% at a character level, though it's complete testing is yet to be widely seen.

3.1.5 SSD Object Detection Model

SSD, a single-shot detector for multiple categories that is faster than the previous state-of-the-art for single shot detectors (YOLO), and significantly more accurate, in fact as accurate as slower techniques that perform explicit region proposals and pooling (including Faster R-CNN). The core of SSD is predicting category scores and box offsets for a fixed set of default bounding boxes using small convolutional filters applied to feature maps. To achieve high detection accuracy we produce predictions of different scales from feature maps of different scales, and explicitly separate predictions by aspect ratio. These design features lead to simple end-to-end training and high accuracy, even on low resolution input images, further improving the speed vs accuracy trade-off[13].

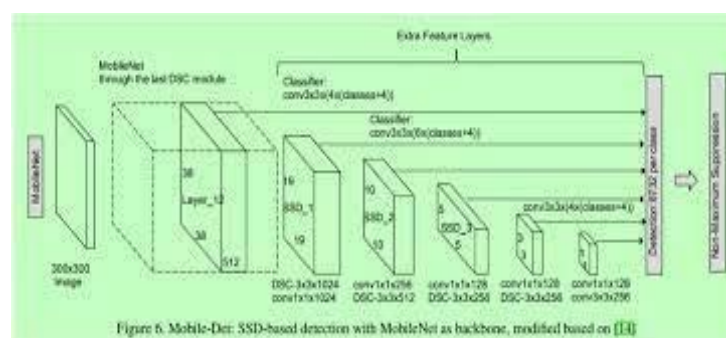


Figure 9: SSD_v2_mobilenet architecture

The SSD approach is based on a feed-forward convolutional network that produces a fixed-size collection of bounding boxes and scores for the presence of object class instances in those boxes, followed by a non-maximum suppression step to produce the final detections. The early network layers are based on a standard architecture used for high quality image classification (truncated before any classification layers), which is called the base network. The convolutional feature layers are at the end of the truncated base network. These layers decrease in size progressively and allow predictions of detections at multiple scales.

3.1.6 OpenCV

Computer Vision

Computer vision is the field of computer science that focuses on replicating parts of the complexity of the human vision system and enabling computers to identify and process objects in images and videos in the same way that humans do. Until recently, computer vision only worked in limited capacity. Computer vision is a process by which we can understand the images and videos how they are stored and how we can manipulate and retrieve data from them. Computer Vision is the base or mostly used for Artificial Intelligence. Computer-Vision is playing a major role in self-driving cars, robotics as well as in photo correction apps.



OpenCV

Open-CV is an open-source computer vision library developed by Intel for real-time image & video analysis and processing. Primarily written in C++, This library has bindings for Python, Java, Matlab, Octave etc. Open-CV combined with python makes image/video analysis and processing astonishingly simple and for many, it can also be the first step in the world of Computer Vision.

Open cv is the most popular library in computer vision. It is originally written in C and C++. It's now available in python also. It is originally developed by intel. The library is a cross-platform open-source library. It is free to use. Open cv library is a highly optimized library with its main focus on real-time applications.

The Open cv Library is a combination of more than 2500 optimized algorithms. which can be used to detect and recognize different faces, identifying objects in images or in real-time, classifying different human actions using videos and webcam, tracking camera movements, tracking moving objects like car, humans, etc., counting objects in real-time, stitch images together to produce a high-resolution image, find similar images from an image database, remove red eyes from images taken using flash and increasing the quality of image, follow eye movements, tracking faces, etc.

OpenCV is the huge library for computer vision, machine learning, and image processing and now it plays a major role in real-time operation which is very important in today's systems. By using it, one can process images and videos to identify objects, faces, or even handwriting of a human.

When it is integrated with various libraries, such as NumPy, Python is capable of processing the OpenCV array structure for analysis. To identify image pattern and its various features we use vector space and perform mathematical operations on these features.

Using OpenCV, we have captured frames for real-time detection. For this, we have collected 16 pictures of each selected sign. The collected images were taken considering variety of angles and orientations to increase the detection accuracy.

3.1.7 TensorFlow

Open-source end-to-end platform for creating Machine Learning applications. Symbolic math library that uses dataflow and differentiable programming to perform various tasks focused on training and inference of deep neural networks. It allows developers to create machine learning applications using various tools, libraries, and community resources.

TensorFlow is a free and open-source software library for dataflow and differentiable programming across a range of tasks. It is a symbolic math library, and is also used for machine learning applications such as neural networks. It is used for both research and production at Google.

Features:

- TensorFlow provides stable Python and C APIs; and without API backwards compatibility guarantee: C++, Go, Java, JavaScript and Swift (early release).
- Third-party packages are available for C#, Haskell, Julia, MATLAB, R, Scala, Rust, OCaml, and Crystal. "New language support should be built on top of the C API. However, not all functionality is available in C yet." Some more functionality is provided by the Python API.

Application: Among the applications for which TensorFlow is the foundation, are automated image-captioning software, such as DeepDream.

3.1.8 Keras

Keras is an Open-Source Neural Network library written in Python designed to be modular, fast and easy to use. Keras is high-level API wrapper for the low-level API, capable of running on top of TensorFlow, CNTK, or Theano.

Keras also compiles our model with loss and optimizer functions, training process with fit function.

It is used for creating deep models which can be productized on smartphones. Keras is also used for distributed training of deep learning models. It wraps the efficient numerical computation libraries Theano and TensorFlow and allows you to define and train neural network models in just a few lines of code.

3.1.9 NumPy

NumPy, which stands for Numerical Python, is a library consisting of multidimensional array objects and a collection of routines for processing those arrays. Using NumPy, mathematical and logical operations on arrays can be performed. Confidence matrix of the detection is calculated using this.

It is the fundamental package for scientific computing with Python. It contains various features including these important ones:

A powerful N-dimensional array object

- Sophisticated (broadcasting) functions.
- Useful linear algebra, Fourier transform, and random number capabilities
- NumPy targets the Cpython reference implementation of Python, which is a non-optimizing bytecode interpreter. Mathematical algorithms written for this version of Python often run much slower than compiled equivalents due to the absence of compiler optimization. NumPy addresses the slowness problem partly by providing multidimensional arrays and functions and operators that operate efficiently on arrays; using these requires rewriting some code, mostly inner loops, using NumPy.
- Using NumPy in Python gives functionality comparable to MATLAB since they are both interpreted, and they both allow the user to write fast programs as long as most operations work on arrays or matrices instead of scalars. In comparison, MATLAB boasts a large number of additional toolboxes, notably Simulink, whereas NumPy is intrinsically integrated with Python, a more modern and complete programming language. Moreover, complementary Python packages are available; SciPy is a library that adds more MATLAB-like functionality and Matplotlib is a plotting package that provides MATLAB-like plotting functionality. Internally, both MATLAB and NumPy rely on BLAS and LAPACK for efficient linear algebra computations.

- Python bindings of the widely used computer vision library OpenCV utilize NumPy arrays to store and operate on data. Since images with multiple channels are simply represented as three-dimensional arrays, indexing, slicing or masking with other arrays are very efficient ways to access specific pixels of an image. The NumPy array as universal data structure in OpenCV for images, extracted feature points, filter kernels and many more vastly simplifies the programming workflow and debugging.

3.1.10 ReactJS

Managing our frontend component. React (also known as React.js or ReactJS) is a free and open-source front-end JavaScript library for building user interfaces based on UI components. React can be used as a base in the development of single-page, mobile, or server-rendered applications with frameworks like Next.js.



ReactJS is a declarative, efficient, and flexible JavaScript library for building reusable UI components. It is an open-source, component-based front end library which is responsible only for the view layer of the application. It was initially developed and maintained by Facebook and later used in its products like WhatsApp & Instagram.

However, React is only concerned with state management and rendering that state to the DOM, so creating React applications usually requires the use of additional libraries for routing, as well as certain client-side functionality.

The main objective of ReactJS is to develop User Interfaces (UI) that improves the speed of the apps. It uses virtual DOM (JavaScript object), which improves the performance of the app. The JavaScript virtual DOM is faster than the regular DOM. We can use ReactJS on the client and server-side as well as with other frameworks. It uses components and data patterns that improve readability and helps to maintain larger apps.

To create React app, we write React components that correspond to various elements. We organize these components inside higher level components which define the application structure. For example, we take a form that consists of many elements like input fields, labels, or buttons. We can write each element of the form as React components, and then we combine it into a higher-level component, i.e., the form component itself. The form components would specify the structure of the form along with elements inside of it.

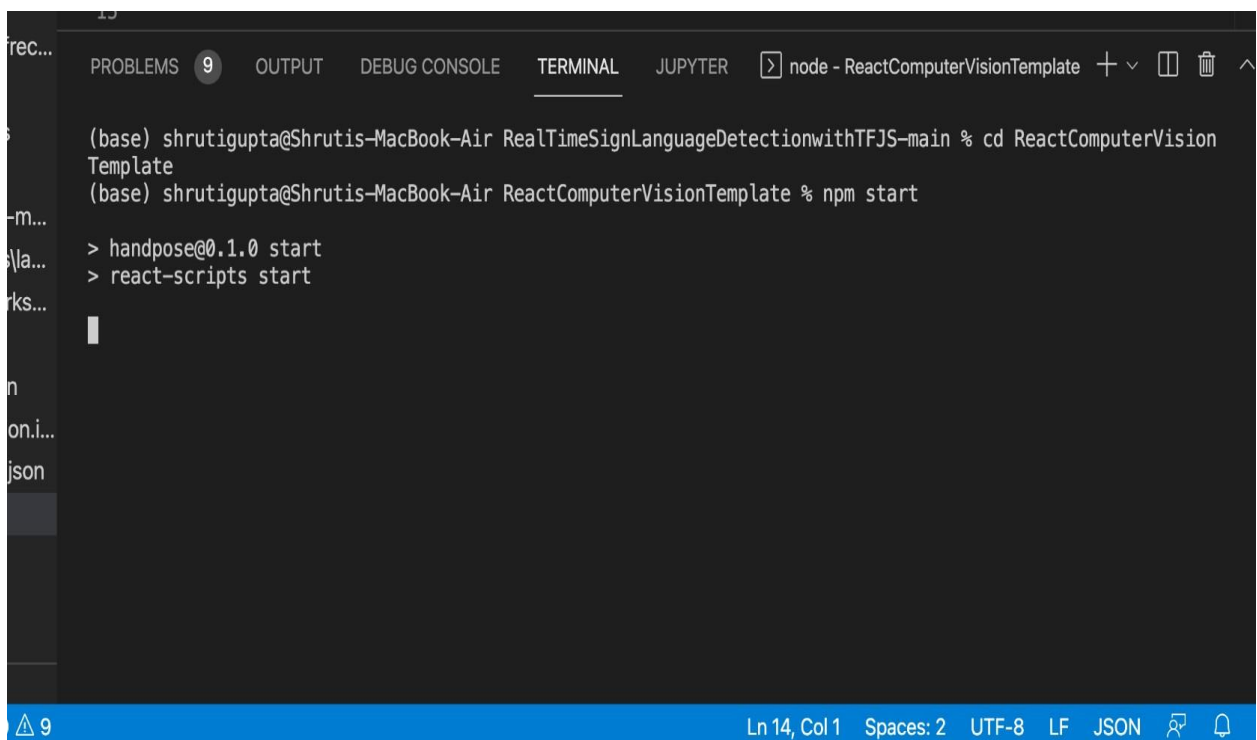
The react package contains only the functionality necessary to define React components. It is typically used together with a React renderer like react-dom for the web, or react-native for the native environments.

There are two ways to set up an environment for successful ReactJS application. They are given below.

1. Using the npm command
2. Using the create-react-app command

Create React App provides a familiar environment for working with React and is the recommended method for getting started with building a new single-page application in React.

It sets up your development environment so that you can use the latest JavaScript features, provides a nice developer experience, and optimizes your app for production. You'll need to have Node $\geq 14.0.0$ and npm ≥ 5.6 on your machine. To create a project, run:

A screenshot of a terminal window within a code editor. The terminal shows a sequence of commands and their outputs. The first command is `cd ReactComputerVisionTemplate`, which changes the directory. The second command is `npm start`, which triggers a series of messages from the development server, including the version of handpose and react-scripts. The terminal interface includes tabs for PROBLEMS, OUTPUT, DEBUG CONSOLE, and TERMINAL. The status bar at the bottom indicates the current file is at line 14, column 1, with 2 spaces, using UTF-8 encoding and LF line endings.

```
(base) shrutigupta@Shrutis-MacBook-Air RealTimeSignLanguageDetectionwithTFJS-main % cd ReactComputerVisionTemplate
(base) shrutigupta@Shrutis-MacBook-Air ReactComputerVisionTemplate % npm start

> handpose@0.1.0 start
> react-scripts start
```

Figure 10: Command for starting the Development (React) Server

3.2 Development Tool

3.2.1 Microsoft Visual Studio

Visual Studio Code is a streamlined code editor with support for development operations like debugging, task running, and version control. It aims to provide just the tools a developer

needs for a quick code-build-debug cycle and leaves more complex workflows to fuller featured IDEs, such as Visual Studio IDE.

Visual Studio has a code editor supporting IntelliSense (the code completion component) as well as code refactoring. The integrated debugger functions both as a source-level debugger and a machine-level debugger. Other built-in tools include a code profiler, designer for designing GUI applications, web designer, class designer, and database schema designer. It accepts plug-ins that expand the functionality at almost every level—including adding support for source control systems (like Subversion and Git) and adding new toolsets like editors and visual designers for domain-specific languages or toolsets for other aspects of the software development lifecycle (like the Azure DevOps client: Team Explorer) (like the Azure DevOps client: Team Explorer).

Visual Studio supports 36 different programming languages and allows the code editor and debugger to support (to varying degrees) practically any programming language, providing a language-specific service exists. Built-in languages include C, C++, Visual Basic .NET, C#, JavaScript, XML, HTML, and CSS. Support for other languages such as Python, Ruby, Node.js, and M among others is accessible via plug-ins. Java w

The most basic edition of Visual Studio, the Community edition, is offered free of charge. The slogan for Visual Studio Community version is "Free, fully-featured IDE for students, open-source and individual developers".

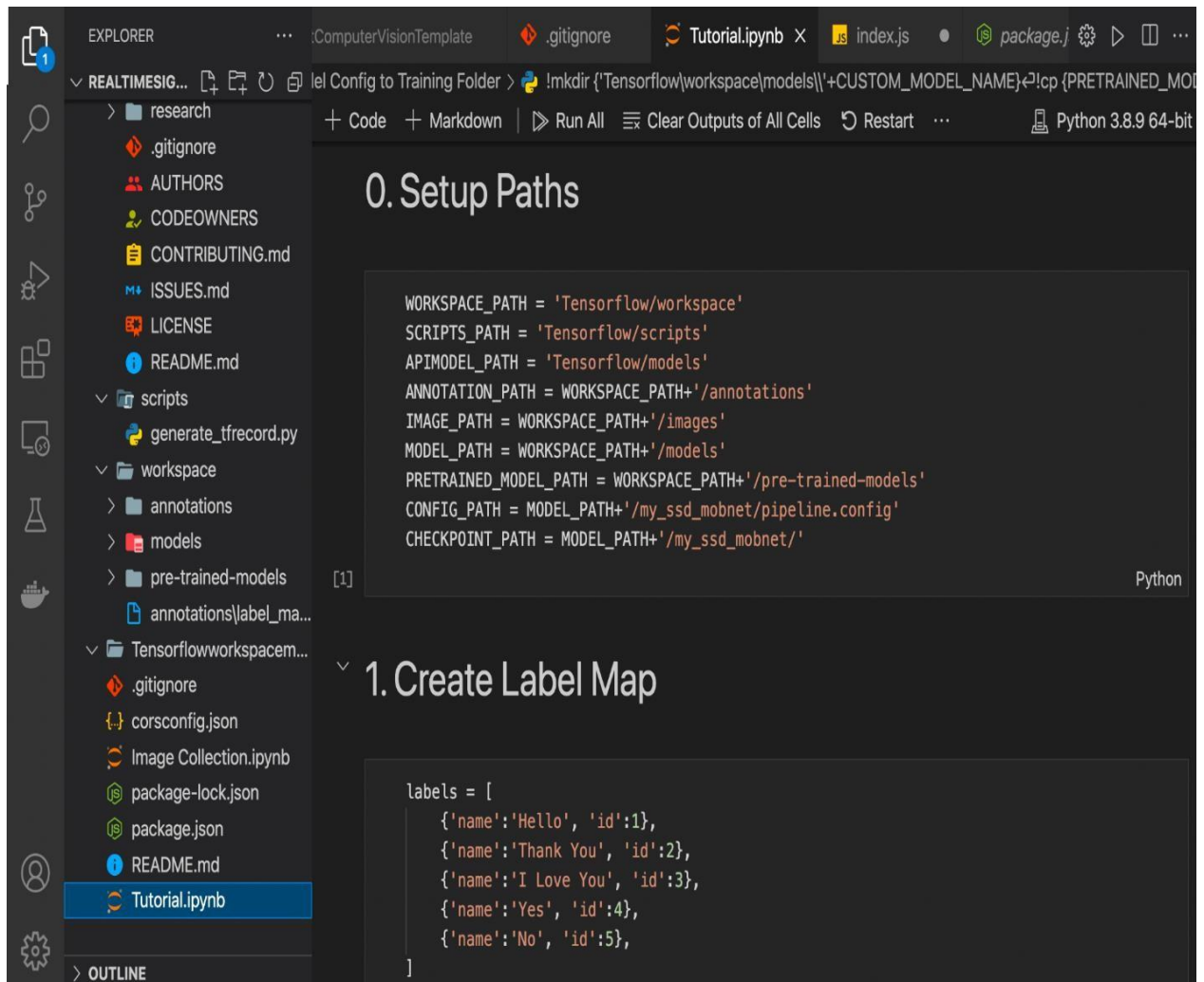


Figure 11: Interface of Visual Studio Code

3.2.2 LabelImg: Image Annotation Tool

LabelImg is a graphical image annotation tool which provides images with bounding boxes after labeling. It is written in Python and uses Qt (one of the most common GUI for python) for its graphical interface. The output labeled data are saved as XML files .

Significance of LabelImg

- It is open-source then it is free to use.

- It has both online and offline versions. The offline interface is user-friendly and allows the user to make fast annotations.
- The output annotation images are saved separately as XML files along with Pascal VOC format or YOLO format. These are common formats supported by many ML packages.
- After installation, it is easy to use with simple instructions.
- It can process a large number of images when compared to other free apps such as **Cloud Vision** or **Label box**
- It can be run in multiple operating systems such as Linux, Mac or Windows.
- It is written in Python then it can interact with other python packages and is easy to be used in python applications for further development.



Figure 12: Image Annotation of Hello Sign

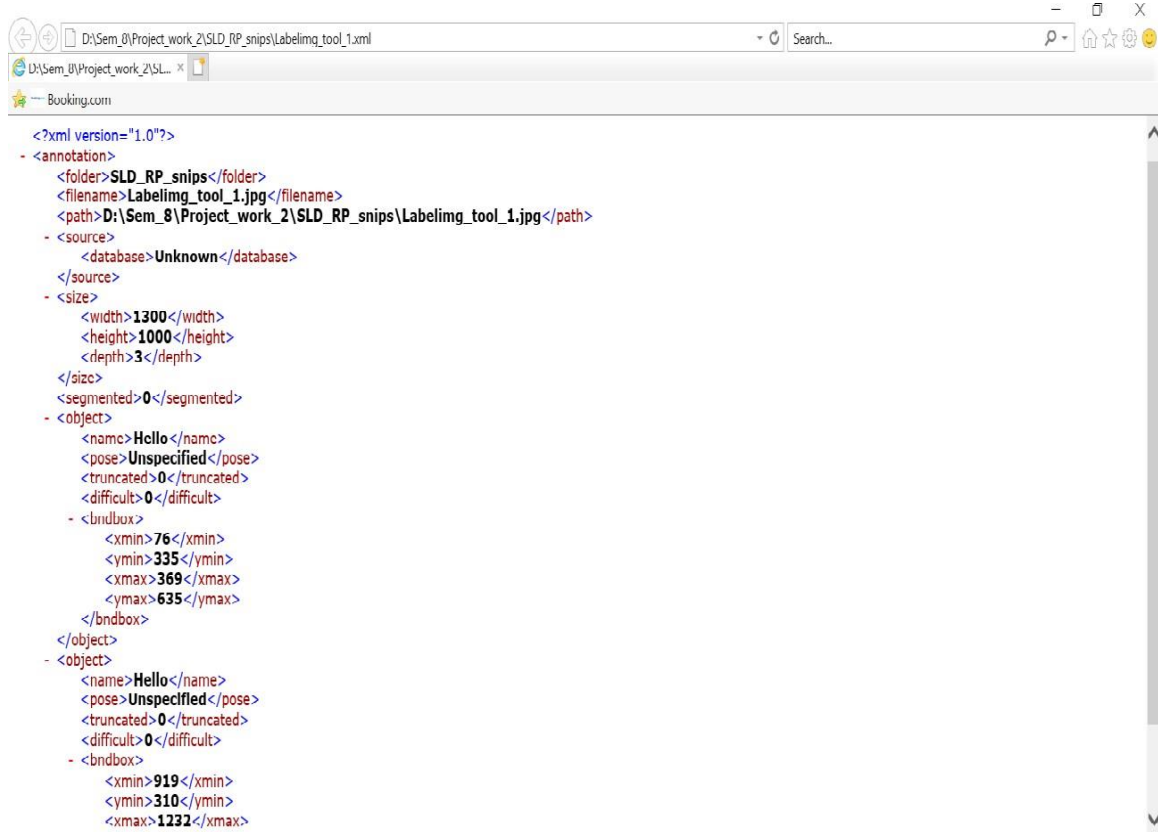


Figure 13 : XML file of Hello Sign

3.3 Working of or project

3.3.1 Detection in real time

Real time video can be thought as sequence of multiple frames. For real time detection, the system will take hand gesture as input data. It would detect and track hands and fingers placed within its field of view and presents motion tracking data as a series of snapshots called frames. Each frame is resized to dimension of 800x600. Each frame of tracking data contains the measured positions, orientations and other information about each entity detected in that snapshot. Using this information, it will try to match the detected frame to it correct sign and display the label as output as shown in Fig.3.

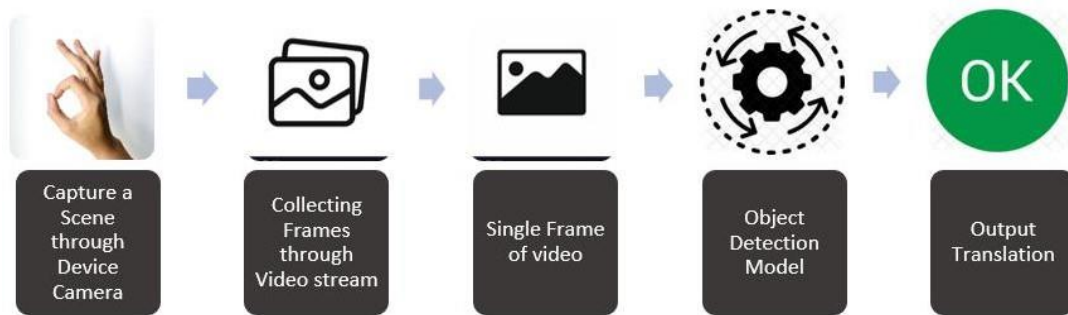


Figure 14: Workflow of Sign language Detection System

3.4 Testing

The purpose of testing is to discover errors. Testing is a process of trying to discover every conceivable fault or weakness in a work product.

It provides a way to check the functionality of components, sub assemblies, assemblies and/or a finished product. It is the process of exercising software with the intent of ensuring that the software system meets its requirements and user expectations and does not fail in an unacceptable manner.

Software testing is an important element of the software quality assurance and represents the ultimate review of specification, design and coding. The increasing feasibility of software as a system and the cost associated with the software failures are motivated forces for well planned through testing.

3.5.1 Testing Objectives:

There are several rules that can serve as testing objectives they are:

- Testing is a process of executing a program with the intent of finding an error.
- A good test case is the one that has a high probability of finding an undiscovered error.

Types of Testing:

In order to make sure that the system does not have errors, the different levels of testing strategies that are applied at different phases of software development are:

Unit Testing:

Unit testing is done on individual models as they are completed and becomes executable. It is confined only to the designer's requirements.

- o Unit testing is different from and should be preceded by other techniques, including:
Inform Debugging
- o Code Inspection

Black Box testing

In this strategy some test cases are generated as input conditions that fully execute all functional requirements for the program.

This testing has been used to find error in the following categories:

- Incorrect or missing functions
- Interface errors
- Errors in data structures are external database access
- Performance error
- Initialisation and termination of errors
- In this testing only the output is checked for correctness
- The logical flow of data is not checked

White Box testing

In this the test cases are generated on the logic of each module by drawing flow graphs of that module and logical decisions are tested on all the cases.

It has been used to generate the test cases in the following cases:

- Guarantee that all independent paths have been executed .
- Execute all loops at their boundaries and within their operational bounds.
- Execute internal data structures to ensure their validity.

Integration Testing

Integration testing ensures that software and subsystems work together as a whole. It tests the interface of all the modules to make sure that the modules behave properly when integrated together. It is typically performed by developers, especially at the lower, module to module level.

Testers become involved in higher levels of system testing. In-house testing of the entire system before delivery to the user. The aim is to satisfy the user that the system meets all requirements

of the client's specifications. It is conducted by the testing organization if a company has one. Test data may range from and generated to production.

- Requires test scheduling to plan and organize:
- Inclusion of changes/fixes.
- Test data to use One common approach is graduated testing: as system testing progresses and (hopefully) fewer and fewer defects are found, the code is frozen for testing for increasingly longer time periods.

Acceptance Testing

It is a pre-delivery testing in which the entire system is tested at the client's site on real world data to find errors.

User Acceptance Test (UAT)

Beta testing: Acceptance testing in the customer environment.

Requirements traceability

Match requirements to test cases.

- Every requirement has to be cleared by at least one test case.
- Display in a matrix of requirements vs. test cases.

Test Cases

Table 2: Test Cases

| Sr.no | Test Case | Input Description | Expected Output | Result |
|-------|----------------------------|---|-------------------------------|--|
| 1 | Application Loading | Deploying the model and run the app | Loading without error | Pass |
| 2 | Noise Removal | Captured frames | Detection of only hands | Pass But sometimes detecting surrounding object |
| 3 | Recognize the hand gesture | Live Gestures in Dim Light | Gesture Detected Successfully | Detecting Rarely |
| | | Live Gesture in very bright environment | | Detecting to its correct Label |
| | | Live Gesture in proper Light | | Detecting to its Correct Label |

Chapter 4

Results And Discussion

Each sign for which the model was trained is detected with high accuracy for both hands as shown in Fig 4 and summarized results can be seen in Table1.

Table 3: Experimental Results

| Sr.No | Gestures | Accuracy Achieved |
|------------------|-----------|-------------------|
| 1. | Hello | 98% |
| 2. | Yes | 99% |
| 3. | No | 99% |
| 4. | Thank You | 94% |
| 5. | See You | 84% |
| Average accuracy | | 94.8% |

We calculated accuracy of each sign and finally found the average accuracy of sign gestures that is 94.8 %.

As you can see in the figure below

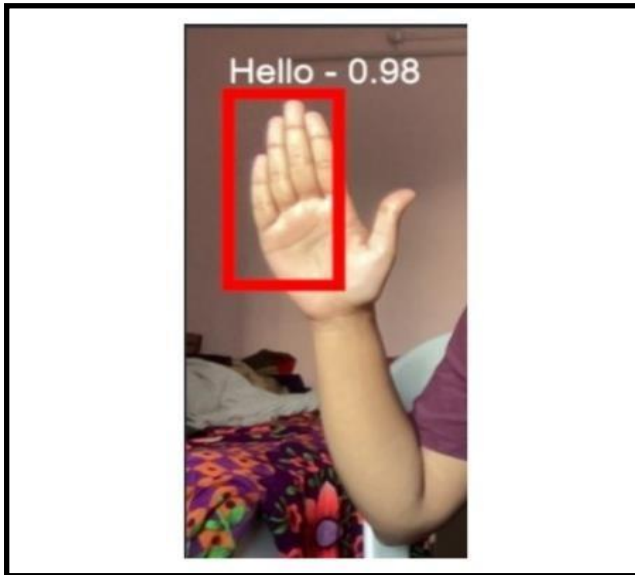


Figure 15 : Result Of Hello

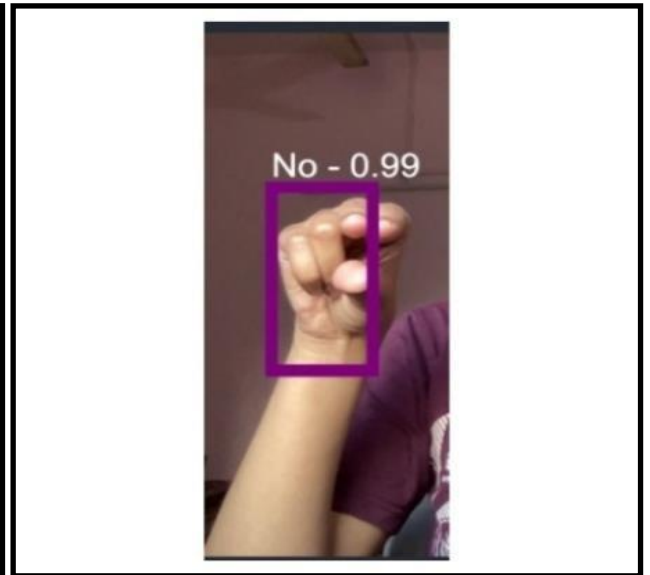


Figure 16: Result of No



Figure 17: Result of Yes



Figure 18: Result of Thank You

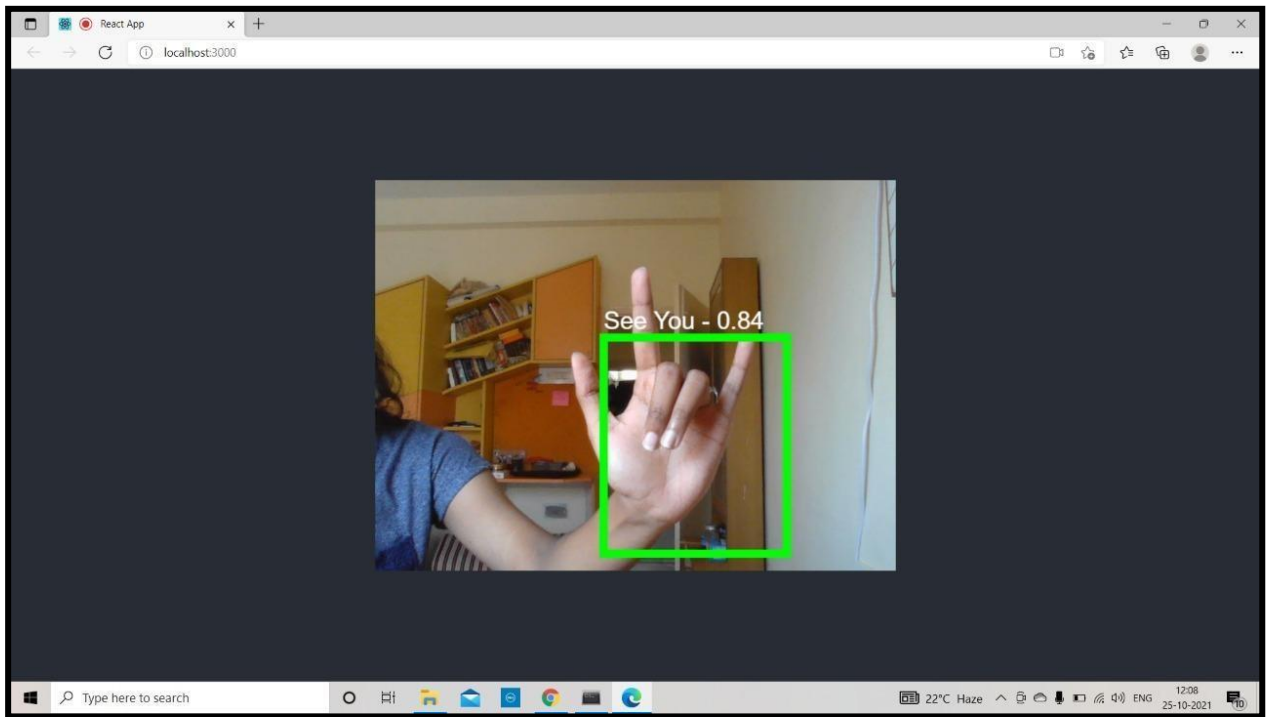


Figure 19: Result of See You

Chapter 5

Summary and Conclusion

Summary

Sign language recognition is a problem that has been addressed in research for years. However, we are still far from finding a complete solution available in our society. Among the works developed to address this problem, the majority of them have been based on basically two approaches: contact-based systems, such as sensor gloves; or vision-based systems, using only cameras. The latter is way cheaper and the boom of deep learning makes it more appealing. When the latter approach is scaled to work over real time detection, the system can prove itself as an aid to society.

Sign language recognition system is developed to facilitate the communication between signers and non-signers. Overall, the system uses CNN and OpenCV as the foundation technologies. Through transfer learning, our object detection is trained and is able to detect gestures in real-time and managed to obtain a satisfactory result.

Conclusion

Technology cannot be considered "accessible" until and unless it is easily operable by every user, regardless of disability. Around a billion individuals on the planet are unable to access applications in the same manner that we do. Making an application universally accessible is not just the law, but it is also quite simple if we put our minds to it. Inclusion is about comprehending, respecting, and ensuring that everyone's voice and opinion are heard and considered seriously. It is critical to foster an atmosphere of mutual respect.

The primary goal of this effort is to eliminate communication obstacles between the hearing impaired, the deaf, and the general public. If sign language detection can be integrated as a feature, a broader audience can benefit from video conferencing services.

We trained our model with photos in a variety of directions and orientations; as a result, our system is capable of identifying precise and exact gestures with an average accuracy of 94.8 percent in real time from every viewpoint. Our solution does not require any special tools such as gloves, sensors, or other specialized equipment; all that is required is a web camera.

The system is simple, quick, adaptable, and fast for any user, with an intuitive interface. Due to society's lack of fluency in sign language, a sign language identification technology integrated into video conferencing platforms will enable these differently abled individuals to integrate more easily.

Chapter 6

Future Scope

1. This software can be scaled over applications like Zoom, Microsoft teams etc, for the ease of every individual.
2. In future AI assistants like Alexa, google duo can be deployed to respond to sign language using AI.
3. Just like WhatsApp and Google where we can use the feature of Voice Search (Speech-to-text translation), this project may be further developed to convert the sign Language to text using Natural Language Processing.

Bibliography

- [1] A. Moryossef, I. Tsochantaridis, R. Aharoni, S. Ebling, and S. Narayanan, "Real-Time Sign Language Detection using Human Pose Estimation," Aug. 2020, [Online]. Available: <http://arxiv.org/abs/2008.04637>
- [2] S. Shrenika and M. Madhu Bala, "Sign Language Recognition Using Template Matching Technique," in *2020 International Conference on Computer Science, Engineering and Applications (ICCSEA)*, Mar. 2020, pp. 1–5. doi: 10.1109/ICCSEA49143.2020.9132899.
- [3] Z. Zhang, Z. Su, and G. Yang, "Real-Time Chinese Sign Language Recognition Based on Artificial Neural Networks *," in *2019 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, Dec. 2019, pp. 1413–1417. doi: 10.1109/ROBIO49542.2019.8961641.
- [4] R. Bhadra and S. Kar, "Sign Language Detection from Hand Gesture Images using Deep Multi-layered Convolution Neural Network," in *2021 IEEE Second International Conference on Control, Measurement and Instrumentation (CMI)*, Jan. 2021, pp. 196–200. doi: 10.1109/CMI50323.2021.9362897.
- [5] J. Singha and K. Das, "Recognition of Indian Sign Language in Live Video," *International Journal of Computer Applications*, vol. 70, no. 19, pp. 17–22, May 2013, doi: 10.5120/12174-7306.
- [6] R. Y. Wang and J. Popović, "Real-time hand-tracking with a color glove," *ACM Transactions on Graphics*, vol. 28, no. 3, pp. 1–8, Jul. 2009, doi: 10.1145/1531326.1531369.
- [7] D. Deora and N. Bajaj, "Indian sign language recognition," in *2012 1st International Conference on Emerging Technology Trends in Electronics, Communication & Networking*, Dec. 2012, pp. 1–5. doi: 10.1109/ET2ECN.2012.6470093.
- [8] Y. 'Raushan, A. 'Shirpurkar, V. 'Mudholkar, S. 'Walke, T. 'Makde, and P. 'Wahane, "SIGN LANGUAGE DETECTION FOR DEAF AND DUMB USING FLEX SENSORS," *International Research Journal of Engineering and Technology (IRJET)*, vol. 04, pp. 1–3, Mar. 2017.
- [9] F. Nasir, U. Farooq, Z. Jamil, M. Sana, and K. Zafar, "Automated Sign Language to Speech Interpreter," in *2014 12th International Conference on Frontiers of Information Technology*, Dec. 2014, pp. 307–312. doi: 10.1109/FIT.2014.64.
- [10] T. Aadit, J. Deepak, P. Janhavi, and D. Jyoti, "Video Chat Application for Mutes," in *2021 International Conference on Emerging Smart Computing and Informatics (ESCI)*, Mar. 2021, pp. 181–185. doi: 10.1109/ESCI50559.2021.9397044.

- [11] D. Talukder and F. Jahara, “Real-Time Bangla Sign Language Detection with Sentence and Speech Generation,” in *2020 23rd International Conference on Computer and Information Technology (ICCIT)*, Dec. 2020, pp. 1–6. doi: 10.1109/ICCIT51783.2020.9392693.
- [12] D. Bragg *et al.*, “Sign Language Recognition, Generation, and Translation,” in *The 21st International ACM SIGACCESS Conference on Computers and Accessibility*, Oct. 2019, pp. 16–31. doi: 10.1145/3308561.3353774.
- [13] W. Liu *et al.*, “SSD: Single Shot MultiBox Detector,” 2016, pp. 21–37. doi: 10.1007/978-3-319-46448-0_2.

List of Publications: (Under Review)

- ICTACT Journal Section
Journal Name: Journal on Soft Computing
Acknowledgement Date: 28th Feb 2022
Review Time: 12 to 16 weeks from the day of acknowledgement
- IEEE International Conference on Signal Processing and Communication (SPCOM 2022)
Under Review: 24th April 2022
Acceptance Date: 20th May 2022 (likely to be)
- FICTA 2022 (10th International Conference on Frontiers of Intelligent Computing: Theory and Applications)
Under Review: 20th April 2022
Acceptance Date: 20th May 2022 (likely to be)