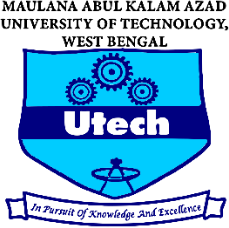
**Real-Time Hand Gesture Recognition**

**B.Tech in Computer Science and Engineering, 8th Semester, 2022-23**



**Prof. Dr. Shanta Phani**

Head of the Department of CSE & IT

Bengal Institute of Technology, Kolkata



**Group Members**

|  |  |
| --- | --- |
| **Name** | **MAKAUT Roll No.** |
| Abhik Chatterjee | 12100119043 |
| Adrija Mahato | 12100119070 |
| Angshuman Bhattacharya | 12100119045 |
| Ritankar Chakraborty | 12100119030 |
| Saraswata Jash | 12100119038 |

****

**BENGAL INSTITUTE OF TECHNOLOGY**

1 GOVT. COLONY, TECH TOWN, DHAPA MANIPUR, ON BASANTI

HIGHWAY, HADIA, KOLKATA-150

***Acknowledgement***

*This project would not been a successful without the sincere cooperation and guidance of our mentor Dr. Prof. Shanta Phani, Head of the Dept, Dept of CSE & IT, Bengal Institute of Technology, Kolkata, who has provided us with useful resources and motivation through the various phases of this project which has prove to be crucial for the completion of this documentation.*

**Signature of Mentor**

----------------------------------------------

Prof. Dr. Shanta Phani

Head of the Department

Dept. of CSE & IT

Bengal Institute of Technology

**Signature of Group Members:**

---------------------------------

Name: Abhik Chatterjee

Roll No. 12100119043

---------------------------------

Name: Adrija Mahato

Roll No. 12100119070

---------------------------------

Name: Angshuman Bhattacharya

Roll No. 12100119045

---------------------------------

Name: Ritankar Chakraborty

Roll No. 12100119030

---------------------------------

Name: Saraswata Jash

Roll No. 12100119038

**Signature of HOD**

---------------------------------------------

Dr. Shanta Phani

Head of the Department

Dept. of CSE & IT

Bengal Institute of Technology

**TABLE OF CONTENTS**

|  |  |
| --- | --- |
| Topic | Page Number |
| 1. Abstract | **4** |
| 1. Introduction   2.1 Random Forest  2.2 Convolutional Neural Network (CNN) | 4 |
| 1. Related Work | 5 |
| 1. Random Forest Classifier   4.1 Dataset  4.2 Data Preprocessing  43.Model Training using Random Forest Classifier  4.4 Final Output for Real-time hand gesture recognition  4.5 Sample Output | 5-8 |
| 1. Convolutional Neural Network (CNN)   5.1 Dataset  5.2 Data Preprocessing and Model Training  5.3 Final Output for Real-time hand gesture recognition | 9-11 |
| 1. Experimental Results and Discussions   6.1 Confusion Matrix  6.2 Deductions | 11 |
| 1. Conclusion | 12 |
| 1. References | 12 |

**1. Abstract:**

This documentation provides a detailed comparison of Random Forest versus CNN to recognize hand gestures in real-time. The objective of this project is to create a reliable and precise system that can recognize hand gestures and convert them into meaningful English subtitles. In order to create a real-time gesture recognition system, the offered project showcases the possibility of predicting the hand gestures using Random Forest Classifier and CNN. It displays the useful applications of this technology, including augmented reality interactions, sign language translation, and interactive interfaces. Developers, researchers, and hobbyists interested in investigating gesture recognition and its incorporation into various applications might benefit greatly from the material.

**Keywords:** Random Forest Classifier, Convolutional Neural Network (CNN), MediaPipe, OpenCV, Tensorflow, Sklearn, Keras, Deep Learning, Confusion Matrix

**2. Introduction:**

Real-time gesture recognition has emerged as a captivating field at the intersection of computer vision and human-computer interaction. It involves the ability to interpret and understand human gestures in real-time, enabling natural and intuitive interaction between humans and machines. Gesture recognition holds tremendous potential across a wide range of applications, including gaming, virtual reality, augmented reality, robotics, and smart environments.

The goal of real-time gesture recognition is to develop algorithms and systems capable of accurately and swiftly recognizing and interpreting human gestures in real-time scenarios. By capturing and analyzing hand movements, body postures, and facial expressions, these systems can extract meaningful information and translate it into actionable commands or responses.

Real-time gesture recognition systems often rely on computer vision techniques to capture and process visual input. They leverage sophisticated algorithms, machine learning models, and sensor technologies to detect, track, and classify gestures. Among the various computer vision libraries and frameworks available, OpenCV has emerged as a popular choice due to its extensive functionality for image and video processing, feature extraction, and object detection.

**2.1 Random Forest**

Random Forest is a popular machine learning algorithm that falls under the category of ensemble methods. It is commonly used for both classification and regression tasks. Random Forest combines the prediction of multiple decision trees to produce a final prediction.

**2.2 Convolutional Neural Network (CNN)**

It is a type of deep learning algorithm specifically designed for processing and analyzing data with a grid-like structure such as images or sequences of data.

A Convolutional Neural Network (CNN) is a network architecture for deep learning that learns directly from data. CNNs are particularly useful for finding patterns in images to recognize objects, classes, and categories. They can also be quite effective for classifying audio, time-series, and signal data.

**3. Related Works**

Various techniques can be used to implement the classification and recognitions of images using machine learning. Image identification is becoming a crucial step in modern world problem solving systems. Sharma[1], Mittal et.al., [1], aimed to identity hand gestures and classify them to their correct meaning with maximum accuracy. A novel approach for the same has been proposed and some other widely popular models have compared with it. The preprocessed data is passed through several classifiers such as Support Vector Machine (SVM), Random Forest, Logistic Regression, Naïve Bayes, etc. to draw effective results. Halder[2] and Tayade[2] have demonstrated a methodology that simplifies Sign Language Recognition on using MediaPipe’s opensource framework and machine learning algorithm.

The predictive model is lightweight and adaptable to smart devices.

**4. Random Forest Classifier**

**4.1 Dataset**

We are using a custom dataset of 10 class. It includes:

“hello”, “iloveyou”, “no”, “yes”, “victory” ”, “thumbs up”, “telephone”, “look up”, “live long”, “fine".

Each class contains 100 images, size ranging from 86KB-92KB, dimension 640x480.

**4.2 Data Preprocessing**

We have used MediaPipe to detect the landmarks of hands. Each landmark is a single joint of our hand which is a total of 21 points. MediaPipe extracts 21 landmarks from each image of each class captured and stores it in a “pickle” file (a file which is a binary file format used in Python for serializing and deserializing Python objects. It allows you to store complex data structures, such as lists, dictionaries, and class instances). The name of the file in which the data points and labels are stored is named asl.pickle file.

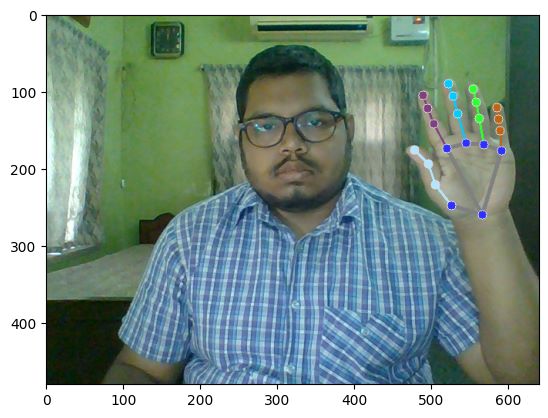


Fig 1.2: Landmark of hand sign “No”

Fig 1.1: Landmark of hand sign “Hello”

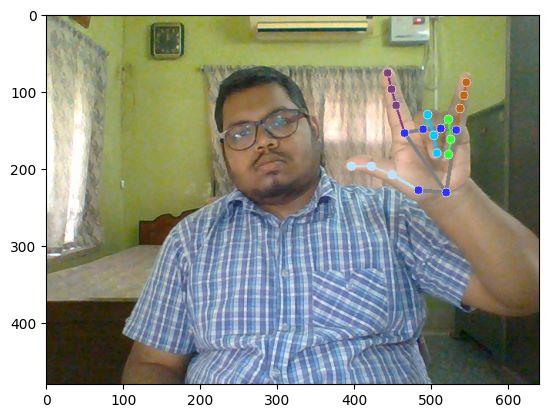
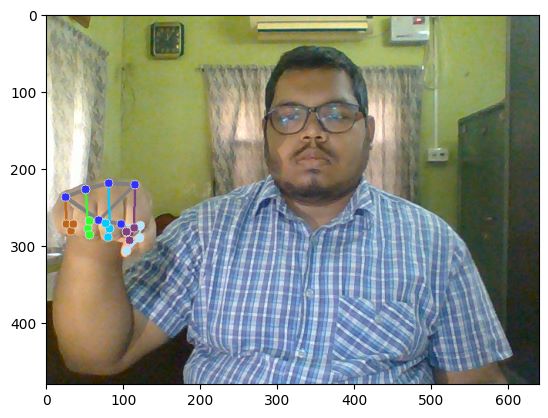


Fig 1.3: Landmark of hand sign “iloveyou”

’

Fig 1.4: Landmark of hand sign “Yes”

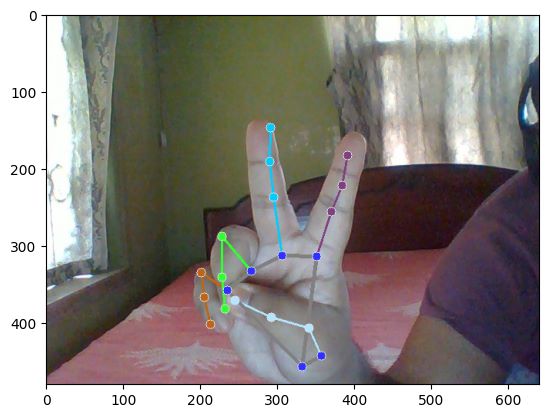
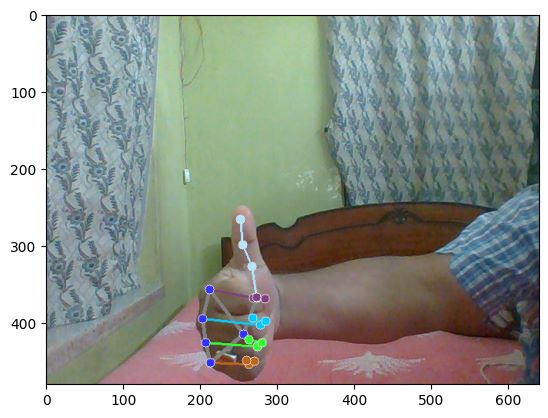
****

Fig1.6: Landmark for hand sign “thumbs up”

Fig 1.5: Landmark of hand sign ‘victory’

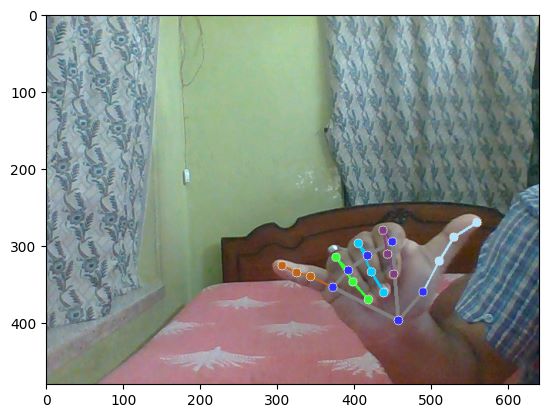
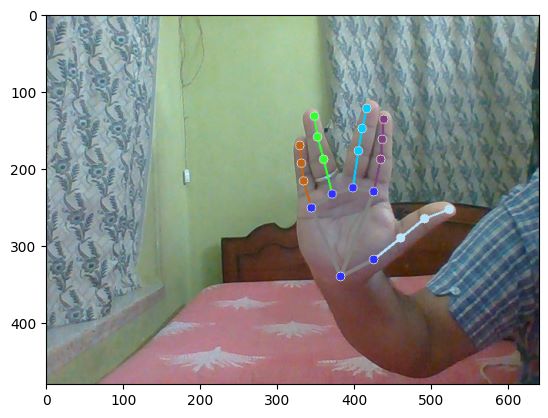
****

Fig 1.8: Landmark for “live long”

Fig 1.7: Landmark for “telephone”

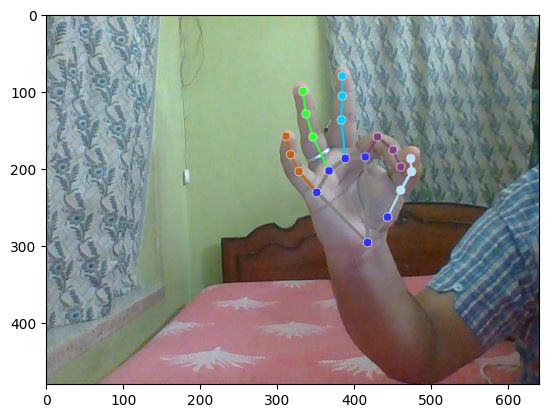
****

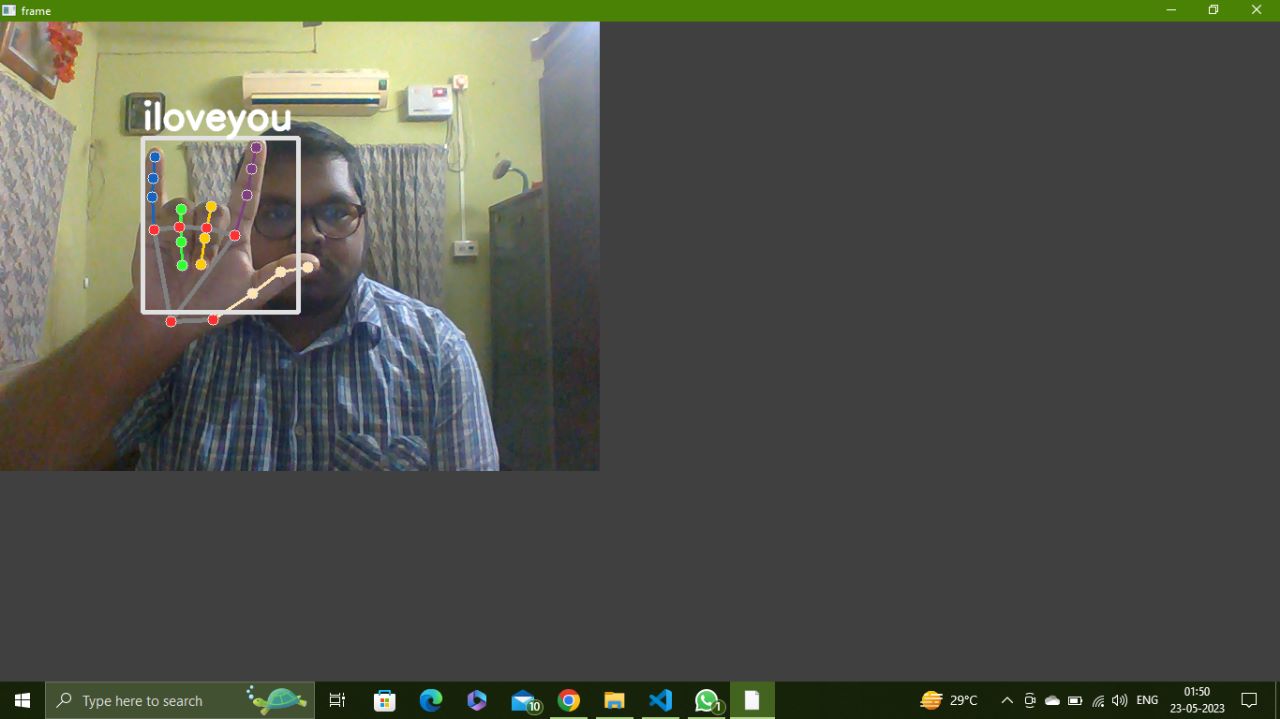
Fig 1.9: Landmark for “fine”

**4.3. Model Training using Random Forest Classifier**

First, the asl.pickle file is loaded, the contents of the file is divided into two lists: 1. Data and 2. Label. These two lists convert the contents of ASL.pickle to numpy array. For training and testing the entire dataset is divided in a percentage ratio of 80:20, where 80% is training dataset and 20% is testing dataset. Random Forest Classifier is called from sklearn module (The sklearn module, also known as scikit-learn, is a popular machine learning library in Python. It provides a wide range of tools and algorithms for tasks such as classification, regression, clustering, dimensionality reduction, and model selection.), which is used to train the model. In our model the accuracy is 100%. The model is saved as model\_ASL.pickle.

**4.4 Final Output for Real-Time Hand Gesture Recognition**

The model\_ASL.pickle file will be loaded after which the camera is initialized using OpenCV. The different classes are further labeled with their respective subtitles like “0” means “hello”, “1” means “iloveyou”, and so on. The camera which was initialized captures images whose landmarks are then detected by MediaPipe. Then a function model.predict() is called which predicts the hand gestures. Finally, the output is on the screen, in which our hand gestures are surrounded by a bounding box which has a tag (name of the gesture/subtitle) is present.

**4.5 Sample Output**

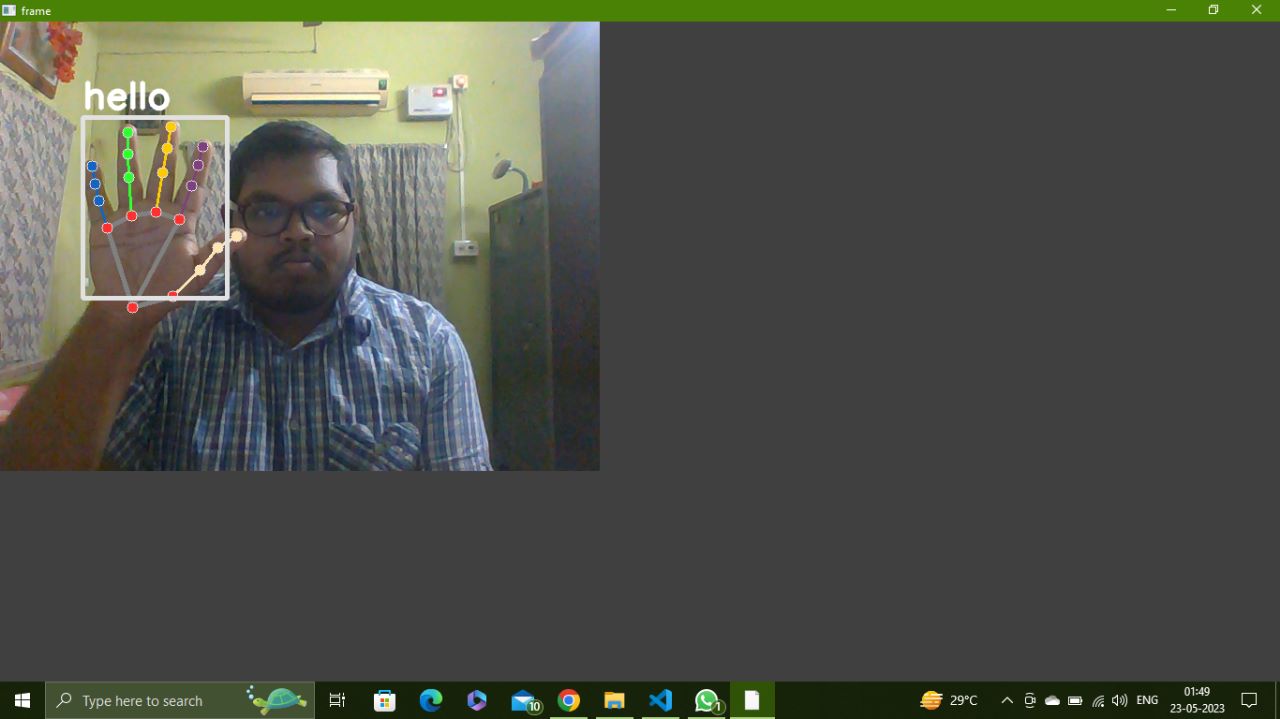


Fig 2.1: Output for hand sign “Hello”

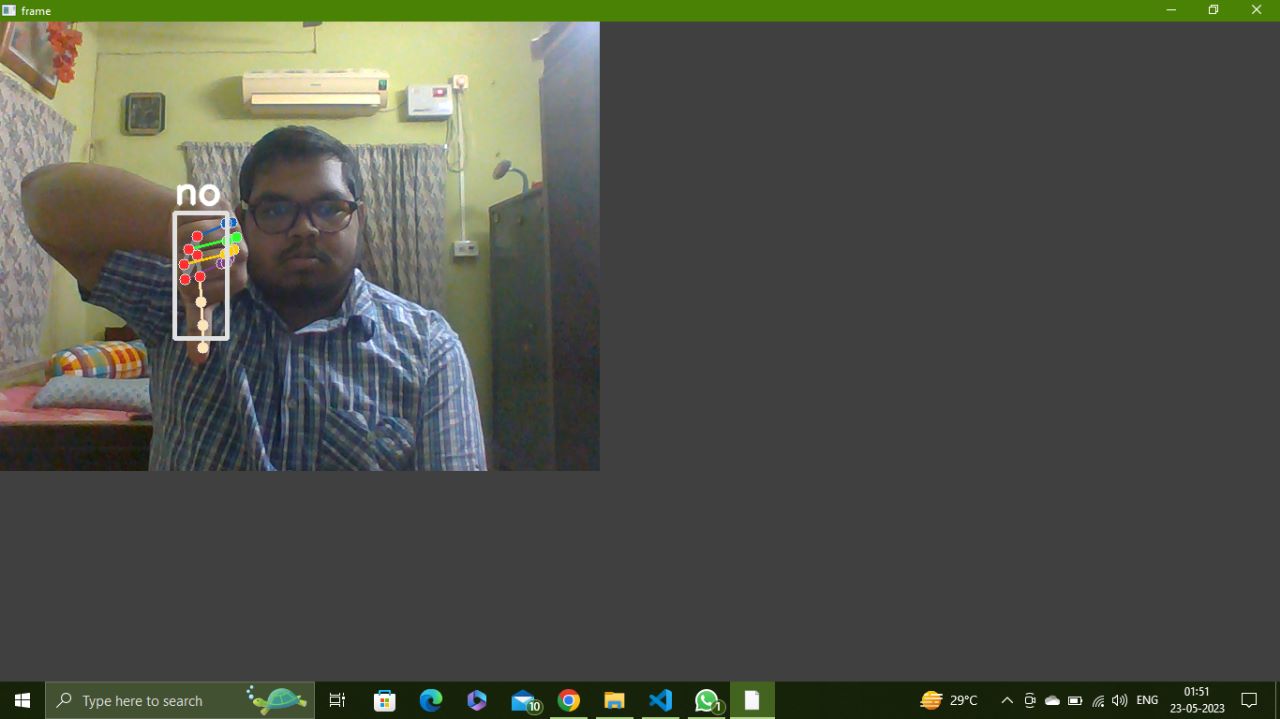
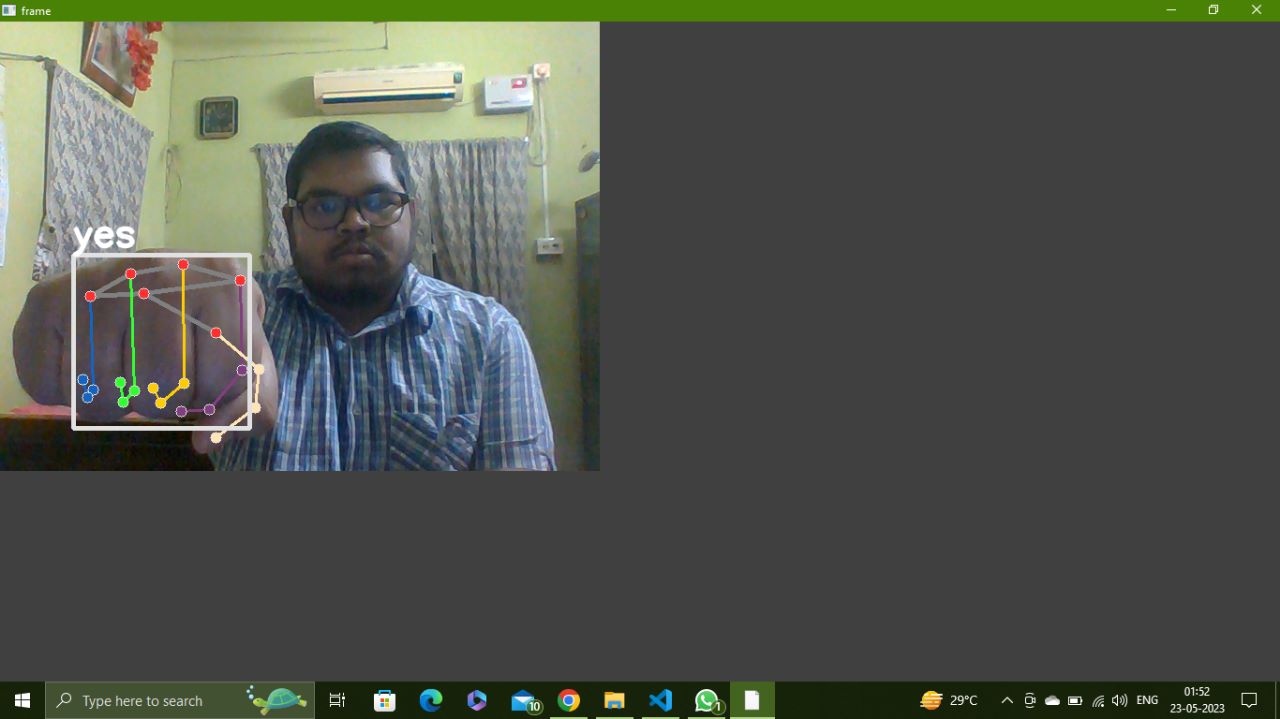
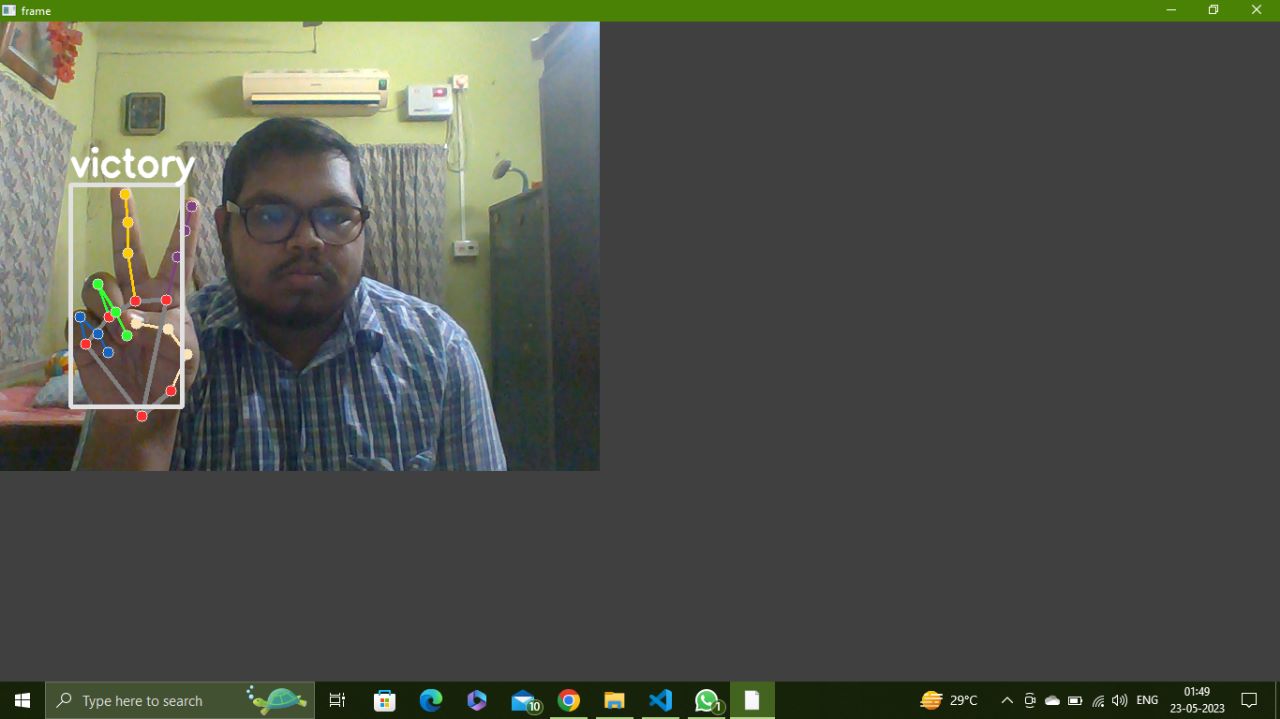
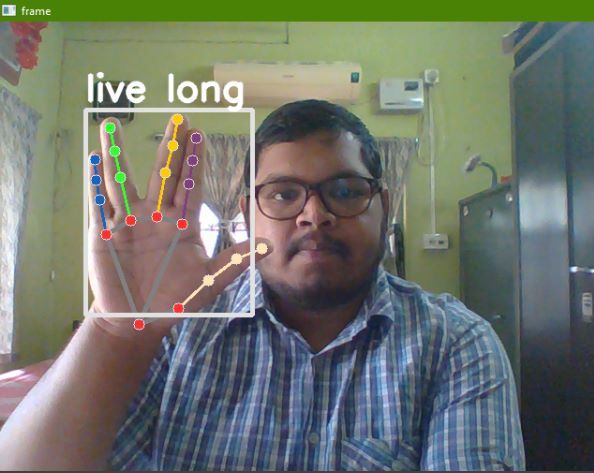


Fig 2.6: Output for hand sign “live long”

Fig 2.2: Output for hand sign “iloveyou”

Fig 2.4: Output for hand sign “yes”

Fig 2.3: Output for hand sign “no”

Fig 2.5: Output for hand sign “victory”

**5. Convolutional Neural Network (CNN)**

**5.1 Dataset**

We are using a custom dataset of 5 class. It includes:

“hello”, “iloveyou”, “no”, “yes”, “victory”.

Each class contains 1000 images, size ranging from 118KB-122KB, dimension 640x480.

**5.2 Data Preprocessing and Model Training**

In the Keras library, the 'to\_categorical' function is used to convert a class vector into binary class matrix representation (the function converts each element into a binary vector, where the index corresponding to the class label is set to 1 and all other indices are set to 0. The resulting ‘class\_matrix’ is a 2D NumPy array representing the binary class matrix.). The name of the file in which the data points and labels are stored is named asl.pickle file.



Fig 3.2: Dataset for hand sing “no”

Fig 3.1: Dataset for hand sign “hello”





Fig 3.3: Dataset for hand sign “victory”

Fig 3.4: Dataset for hand sign “yes”

The contents of the file are divided onto two lists: 1. data and 2. label. These two lists convert the contents of asl.pickle to numpy array. For training and testing the entire dataset is divided in a percentage ratio of 80:20, where 80% is training dataset and 20% is testing dataset. CNN is called from sklearn module, which is used to train the model. The accuracy being 100%.

**5.3 Final Output for Real-Time Hand Gesture Recognition**

The model\_ASL.pickle file will be loaded after which the camera is initialized using OpenCV. The different classes are further labeled with their respective subtitles like “0” means “yes”, “1” means “hello”, and so on. The camera which was initialized captures images using OpenCV. Then a function model.predict() is called, which predicts the hand gestures. Finally, the output is on the screen, in which our hand gestures are surrounded by a bounding box which has a tag (name of the gesture/subtitle) is present.

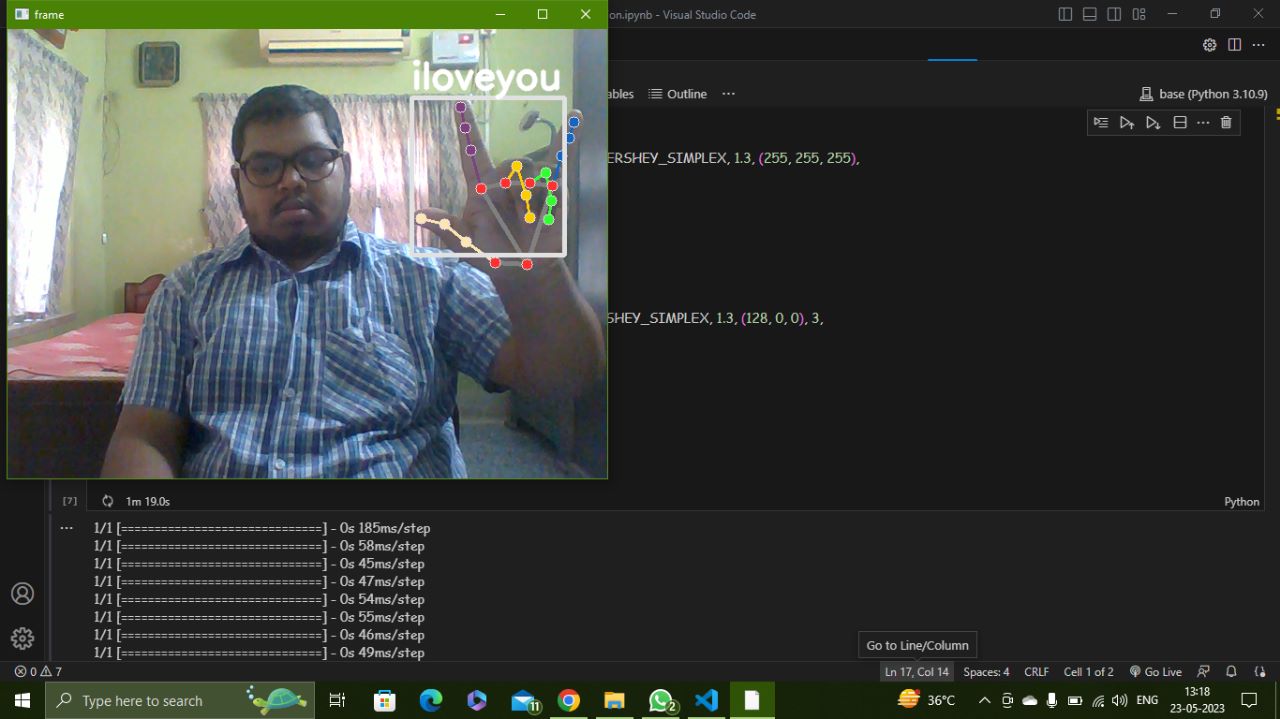


Fig 4.2: Output for “no”

Fig 4.1: Output for “iloveyou”

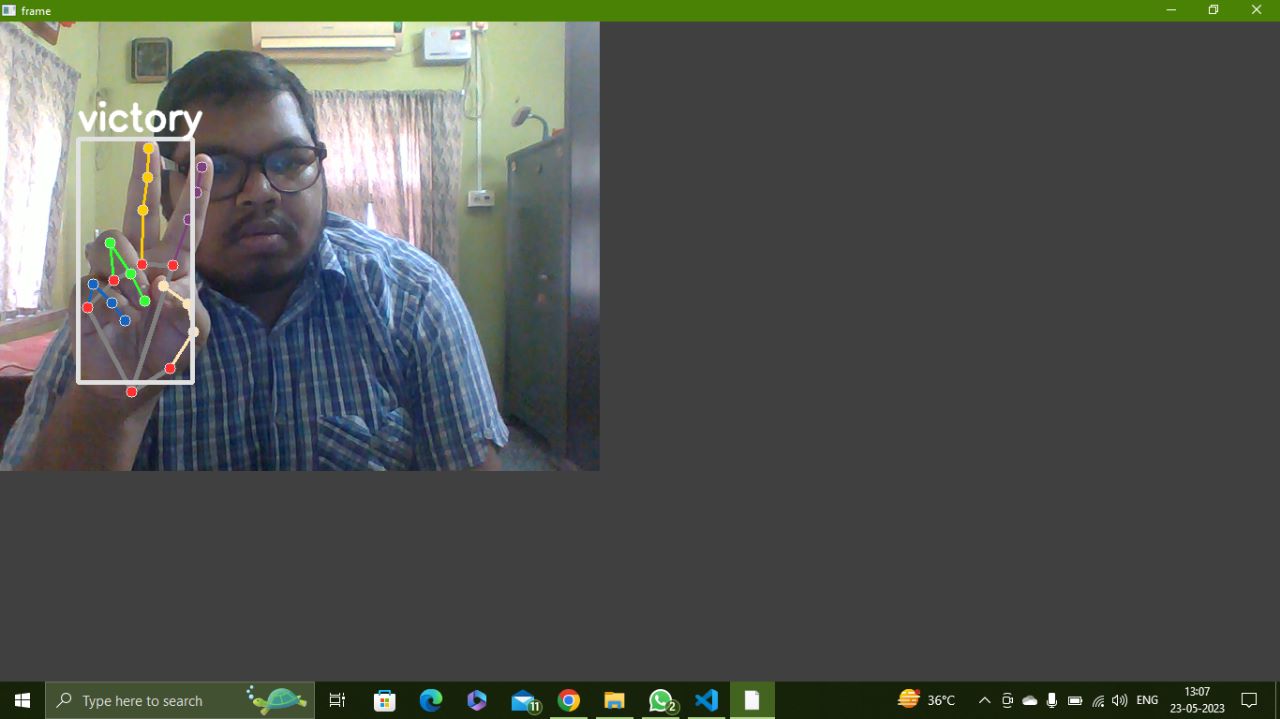
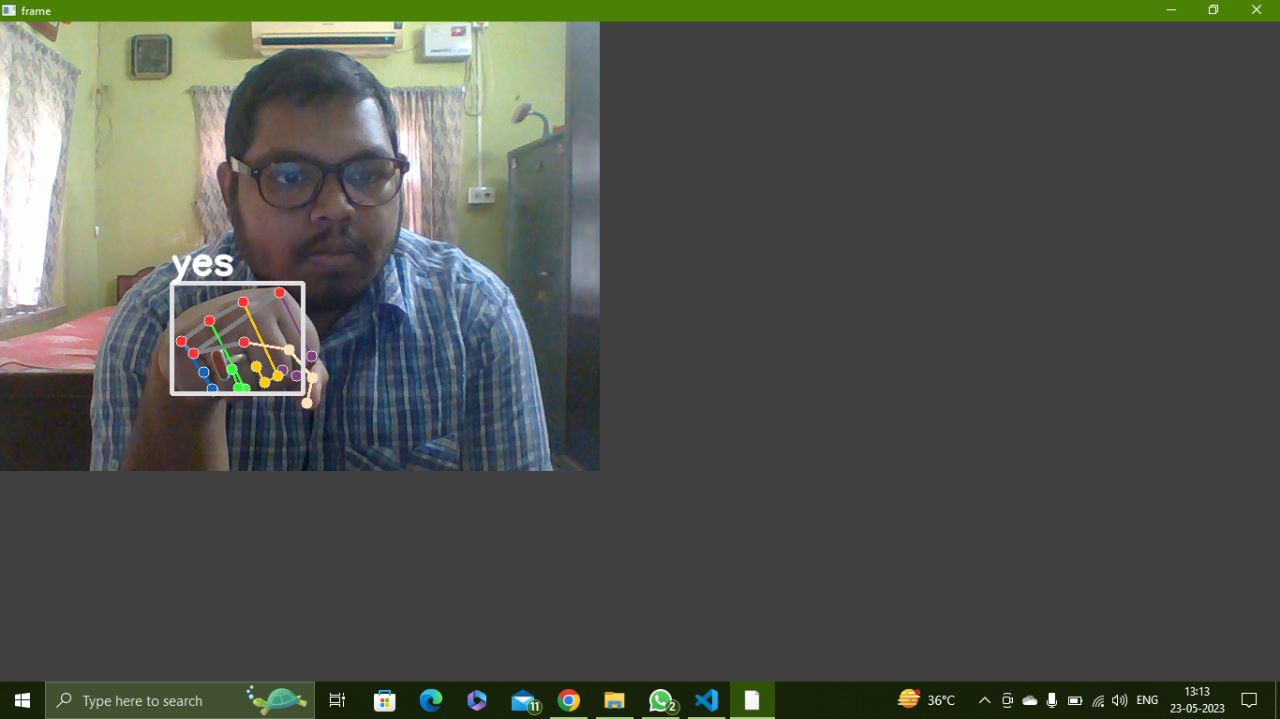


Fig 4.5: Output for “hello”

Fig 4.4: Output for “victory”

Fig 4.3: Output for “yes”

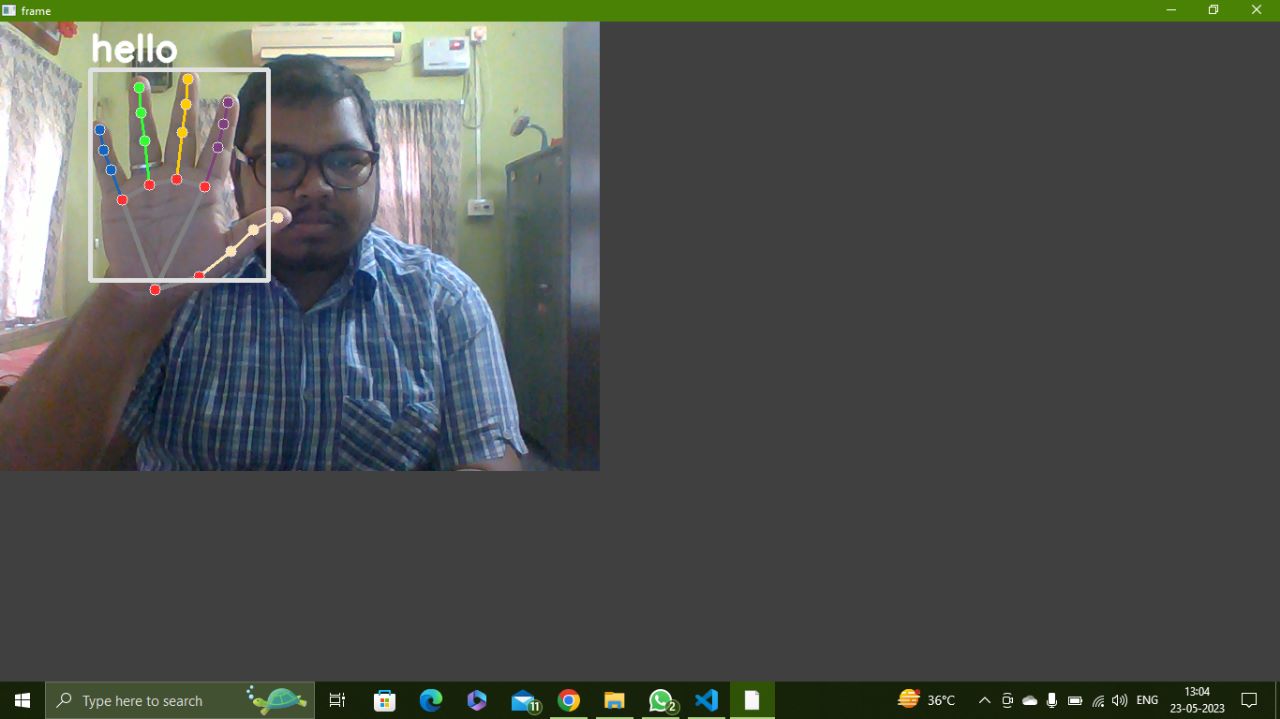


Fig 4.5: Output for “hello”

**6. Experimental Results and Discussions**

**6.1 Confusion Matrix**

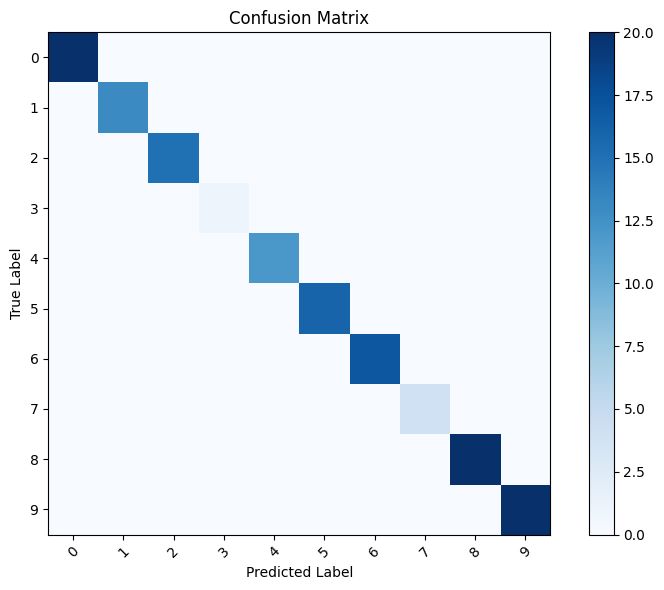
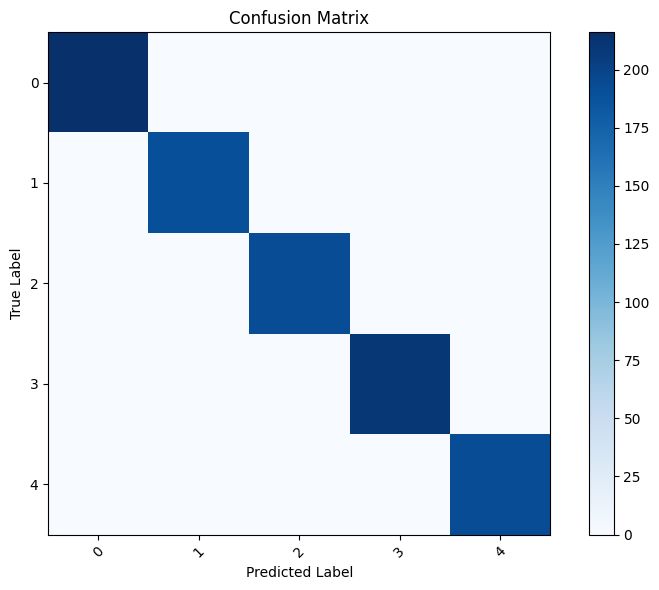
A confusion matrix, also known as an error matrix, is a table that summarizes the performance of a classification model. It is commonly used in machine learning to evaluate the accuracy of a model’s predictions by comparing them to the actual ground truth tables. The confusion matrix provides a detailed breakdown of the correct and incorrect predictions made by the model.

Fig 5.2: Confusion Matrix for CNN

Fig 5.1: Confusion Matrix for Random Forest Classifier

0 – yes

1 – hello

2 – iloveyou

3 – victory

4 - no

0 – hello

1 – iloveyou

2 – no

3 – yes

4 – victory

5 – thumbs up

6 – telephone

7 – look up

8 – live long

9 - fine

**6.2 Deduction**

From our project we have observed that:

While training the model using Random Forest Classifier we are using 100 images per class and we are successfully training the model and acquiring the desired output very quickly.

In the case of CNN, we are using 1000 dataset per class for model training, but despite having a 100% accuracy we are not acquiring the desired output.

Hence, we have deduced that Random Forest which is a machine learning algorithm is a better classifier than Convolutional Neural Network (CNN) which is a deep learning algorithm.

**7. Conclusion**

We tried to do a comparative study between Random Forest Classifier and Convolutional Neural Network (CNN) to check which one gives better results for Real-Time hand gesture recognition. We have thus concluded that Random Forest Classifier is a better alternative than CNN, as it gives the desired output more accurately.

**8. References:**

[1] Sharma, Mittal, Singh and Awatramani: Hand Gesture Recognition using Image Processing and Feature Extraction Techniques (2020)

<https://www.sciencedirect.com/science/article/pii/S187705092031526X>

[2] Arpita Halder and Akshit Tayade: Real-Time Vernacular Sign Language Recognition using MediaPipe and Machine Learning (2021)

<https://github.com/arpita739/Real-time-Vernacular-Sign-Language-Recognition-using-MediaPipe-and-Machine-Learning>

[3] Computer vision engineer YouTube Channel: Sign language detection with Python and Scikit Learn | Landmark detection | Computer vision tutorial

<https://www.youtube.com/watch?v=MJCSjXepaAm>

[4] Nicholas Renotte Youtube Channel: Real Time Sign Language Detection with Tensorflow Object Detection and Python | Deep Learning SSD

<https://www.youtube.com/watch?v=pDXdlXlaCc0>