International Journal of Multidisciplinary Research and Growth Evaluation.

Sign language recognition software

Mohammed Zurain Khan [1*], Mohd Sohail U Zama [2], Mohammed Sayeed [3], Ayesha Siddiqua 4, Dr. M Upendra Kumar [5]
[1-3] B.E. CSE, Department of (AI & ML) CS & AI, Muffakham Jah College of Engineering and Technology is an Engineering College, Hyderabad, Telangana, India
[4] Assistant Professor, Department of CS & AI, Muffakham Jah College of Engineering and Technology is an Engineering College, Hyderabad, Telangana, India
[5] Professor and Associate Head, Department of CS & AI, Muffakham Jah College of Engineering and Technology is an Engineering College, Hyderabad, Telangana, India

* Corresponding Author: **Mohammed Zurain Khan**

Article Info

ISSN (online): 2582-7138
Volume: 05
Issue: 02
March-April 2024
Received: 05-02-2024
Accepted: 09-03-2024
Page No: 510-519

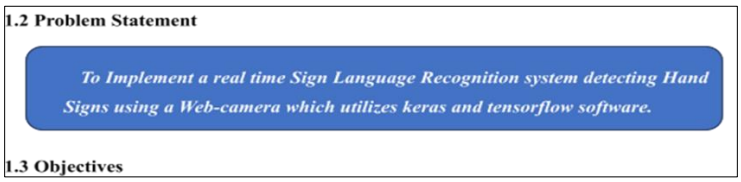
Abstract
Sign language recognition is a critical aspect of enhancing communication accessibility for individuals with hearing impairments. Recognizing the complexity and nuances of sign gestures can be challenging, and automating this process can significantly improve efficiency and accuracy. Leveraging deep learning techniques with frameworks like Tensor Flow, Keras, and Mediapipe, we propose an automated system for sign language recognition. The objective of this system is to expedite the recognition and classification of sign gestures, providing a quick and reliable non-invasive solution for users and facilitating communication. Our approach involves transfer learning using pre-trained models in conjunction with custom architectures tailored for sign language recognition.

DOI: https://doi.org/10.54660/.IJMRGE.2024.5.2.470-473

Keywords: Deep Learning, convolutional neural network, camera recognition, media pipe, sign recognition

1. Introduction

This project tackles the challenge faced by individuals with hearing and speech impairments in communication. We aim to bridge the gap by developing a real-time sign language recognition system using the power of deep learning.
This project tackles the challenge faced by individuals with hearing and speech impairments in communication. We aim to bridge the gap by developing a real-time sign language recognition system using the power of deep learning.
In this report, we provide a comprehensive overview of the project, detailing the methodology, implementation, experimentation, results, and conclusions. We begin by discussing the problem statement and motivation behind the project, followed by an explanation of the dataset used for training and evaluation. We then delve into the architecture of our model, describing its components and rationale behind the design choices.
Our approach utilizes the Keras library built on top of TensorFlow to construct a robust Convolutional Neural Network (CNN) model. This model will be trained on a dataset of hand gestures, enabling it to recognize specific signs and translate them into text language.
OpenCV2 and MediaPipe will be integrated to facilitate real-time video capture and hand detection from user input. This allows for seamless recognition of hand gestures, making communication more natural and efficient.
This project contributes to the field of assistive technology, aiming to empower individuals with hearing and speech difficulties through the power of deep learning and computer vision.



The objectives of this project are listed below:

- To develop a real-time sign language recognition system using deep learning: This system aims to accurately recognize hand gestures from a predefined vocabulary in real-time.
- Educational settings: Support deaf and hard-of-hearing students by providing real-time captioning for lectures and presentations, enhancing their learning experience.
- Public spaces: Integrate the system in kiosks or information booths to offer sign language translation for tourists or individuals with hearing impairments, improving accessibility in public areas.

1.4 Organization of Thesis

The following is the format of the thesis which is outlined in six chapters. A brief description of each chapter is as follows: The first chapter consists of the Introduction, Objectives, and Organization of Thesis. Second chapter contains the Literature Survey. This chapter focuses on the areas such as the Existing System, and the Problems with the Existing System. Third chapter explains about the Proposed System. Here, we talk about what the Proposed system can do, the Problem Statement and the System Architecture. Fourth chapter is about the Methodologies used. It explains about the various kinds of methods used in this project and about the various Technologies used. Fifth chapter is the Implementation. It tells us about the Requirements (Software Requirements), the Code Snippets explaining the main parts of the project with code, and the Execution part.
Finally, the Sixth chapter talks about the Result Analysis and Conclusion. Here, we can see the Result Analysis, Conclusion on the Results obtained and Future Enhancement which is followed by references.

2. Literature Survey

Abdullah Qassim Baktash [1] had proposed a multi sign language glove based hand talking system using hardware components.

Methods

This paper introduces a technique to realize multiple sign language translation using a sensor-based glove and an Android smartphone for speech impaired people to communicate normally with people. The design of the hand talking system (HTS) was implemented with a minimum possible number of sensors and a capable sewing controller (Lilypad). The proposed HTS includes flex sensors, Arduino, smartphone, and accelerometer. The HTS uses an Android application programmed to store multi-language in the SQLite database and enables the user to interact with the system. The system provides talking with a letter formed words, or using the most frequently used words in daily communication by hand gesture.

Results

The HTS has achieved high accuracy obtained for American Sign Language and Arabic Sign Language which are about 98.26% and 99.33% respectively with an average accuracy of 98.795 for both Sign Languages.

Drawbacks

While the study demonstrated promising results, it may have some limitations. For example, the device was not portable and the cost was relatively. Limited vocabulary, not ideal for complex sentences.
Lakshman Karthik Ramkumar *et al.* [2] had proposed a new model, an intel real sense technology which captures the users depth images of gestures.

Methods

The authors use wavelet families, computed on edge images, as features to train a Neural Network for 24 sign classification. Haarlet-like features, computed on grey-scale images and on silhouettes were used for classification of 10 hand shapes. Principal Component Analysis (PCA) was applied directly on images to derive a subspace of hand poses, which is then used to classify the hand poses.
A modification of HOG -descriptors is employed to recognize static signs of the British Sign Language.

Results

The accuracy for the alphabets J & Z is less compared to other alphabets, as they involve motion parameters such as swing and need more time to be recognized. The test accuracies are tested by F1 scores for which precision and recall values are considered Though, CNN proves to be a faster and effective classifier in terms of speed and accuracy. Almost every gesture with support values greater than 370 is predicted correctly.

Drawbacks

However, the study lacked thorough discussion on potential drawbacks or limitations of the proposed framework. Based on our observation it requires a depth sensor and is limited by the sensor quality.
Kamal Preet kaur *et al.* [3] had proposed a Sign Language Recognition software using Image Processing techniques
Methods:
This study uses a vision-based approach used for sign language recognition in which a video camera is used to record hand movements, and the input video is partitioned into frames, for each frame, a set of features are extracted. The system aims at recognizing 26 alphabets using Image processing technique in which feature detection and feature extraction of hand gesture is done with the help of SURF (Speed Up Robust Features) algorithm.

Results

Results for two alphabets ie. For A and B are shown. The system is implemented in MATLAB. The system resolves the difficulties of disabled persons to learn sign language and identify the characters. There is no clear accuracy for the prediction provided as it just proposes a technique of implementation.

Drawbacks
The Study lacked any discussions on the potential drawbacks or limitations upon its implementation, However from our perspectives it lacks accuracy.
Kaustub jadav [4] created a Sign Language recognition software using a neural network applying Gabor Filters.

Method
For hand representation or feature extraction, the system uses Gabor filters as the Gabor filters are known to resemble the receptive fields of the visual cortex. Upon obtaining the Gabor features, they performed a similarity matching technique for gesture recognition called Elastic Bunch-Graph
Matching (EBMS) technique. For experimentation they have considered a subset of 10 gestures from American sign language and the dataset consists of 657 images captured from 24 people in three different backgrounds (Complex, uniform light and uniform dark).

Result
The system has achieved an accuracy of 86.2% under complex background conditions and overall accuracy of 91% under all background conditions.

Drawbacks
The above project doesn't contain a large dataset to train the model as it uses only 645 images in total. This may result in lower prediction accuracy of the signs.
Ang Zi *et al*. [5] developed a Data-Glove for Sign language recognition of people with hearing and speech impairment via wearable inertial sensors

Methods
This paper proposes a solution to this problem by designing a low-cost data glove that utilizes multiple inertial sensors with the purpose of achieving efficient and accurate sign language recognition. In this study, four machine learning models—decision tree (DT), support vector machine (SVM), K-nearest neighbor method (KNN), and random forest (RF)—were employed to recognize 20 different types of dynamic sign language data used by deaf individuals. Additionally, a proposed attention-based mechanism of long and short-term memory neural networks (Attention-BiLSTM) was utilized in the process.

Results
The evaluation of the proposed methodology yielded promising outcomes. The results indicate that both the Attention-BiLSTM and RF algorithms have the highest performance in recognizing the twenty dynamic sign language gestures, with an accuracy of 98.85% and 97.58%, respectively.

Drawbacks
In practical engineering fields, traditional machine learning methods tend to have significantly lower computational costs than deep learning methods. Wearable devices for sign language recognition must consider factors such as portability, power consumption, cost, and comfort. Given these considerations, it can be challenging to install the necessary computational units for deep learning,

Table 2.1: Summary of Literature Survey

SNO	Title of paper	Methodology	Results	Advantages	Drawbacks
1	Multi sign language glove-based hand talking system [2021]	Introduced a technique to realize multiple sign language translation using a sensors-based glove and an Android smartphone for speech impaired people.	The HTS has achieved a high accuracy obtained from American sign language and Arabic sign language which are about 98.26% and 99.33% respectively.	Reached an accuracy of 98.26% and 99.33% respectively with an average accuracy of 98.795 for both Sign Languages	The device was not portable and the cost was relatively high. Limited vocabulary, not ideal for complex sentences.
2	Sign Language Recognition using Depth Data and CNN [2019].	The Gesture Recognition is performed using Convolutional Neural Networks, by classifying the gestures. Principle Component Analysis (PCA) was applied directly on images to derive a subspace of hand poses, which is then used to classify the hand poses.	The test accuracies are tested by F1 scores for which precision and recall values are considered. Here CNN proves to be a faster and effective classifier in terms of speed and accuracy.	CNN proves to be a faster and effective classifier in terms of speed and accuracy. Almost every gesture with support values greater than 370 is predicted correctly.	It requires a depth sensor and is limited by the sensor quality. The accuracy for the alphabets J & Z is less compared to other alphabets, as they involve motion parameters such as swing and need more time to be recognized.
3	Sign Language Recognition software using Image Processing techniques [2017].	The system aims at recognizing 26 alphabets using Image processing technique in which feature detection and feature extraction of hand gesture is done with the help of SURF (Speed Up Robust Features) algorithm.	Results for two alphabets ie. For A and B are shown. The system is implemented in MATLAB.	The system resolves the difficulties of disabled persons to learn sign language and identify the characters.	May provide lower accuracies of the predicted values.

4	Sign Language	Upon obtaining the	The system has achieved an accuracy of 86.2% under complex background conditions and overall accuracy of 91% under all background conditions. The results indicate that both the Attention-Bi-LS TM and RF algorithms have the highest performance in recognizing the twenty dynamic sign language gestures, with an accuracy of 98.85% and 97.58%, respectively	It provides us	May require a
5	recognition software using Gabor filters. A Data-Glove for Sign language recognition of people with hearing and speech impairment via wearable inertial sensors	Gabor features, they performed a similarity matching technique for gesture recognition called Elastic Bunch-Graph Matching (EBMS) technique. A CNN architecture is then implemented. A data glove that utilizes multiple inertial sensors and the following algorithms DT, SVM, KNN, RF and Attention-Bi-LSTM was utilized in the process.		with a higher percentage of accuracy of an average of 91%. Easier for detection of signs with higher accuracy.	greater dataset as it may result in lower predictions. Compared to vision based technology, it is not portable, it is not as cost effective and is more hardware dependent.

3 Existing System

Some of the existing systems for recognizing Sign Language are based on machine learning, deep learning. These systems can perform tasks such as Feature classification, feature extraction and image classification as well. For example, some systems use convolutional neural networks (CNNs) like SVM, k- means clustering, other systems use deep learning models such as 3D CNNs and Bi- LSTM to detect the signs. Several studies have explored the efficacy of various machine learning and deep learning models in the domain of sign language recognition. Feature classification plays a crucial role in extracting relevant information from the input data .The following summaries provide insights into different approaches.

Kaustab Jadav *et al*. (2017) [4]

- Created a sign language recognition system using Gabor filters.
- Integrated convolutional neural networks with Gabor filters techniques.

Abdullah Qassim Baktash *et al*. (2021) [1]

- Proposed a Multi-Sign Language Glove-Based Hand Talking System.
- Multiple sign language translation using a sensors-based glove and an Android smartphone for speech impaired people.
- Achieved a high and stable prediction of 98.79%.

Ang Zi *et al*. (2023) [5]

- Developed a Data-Glove for Sign language recognition of people with hearing and speech impairment via wearable inertial sensor.
- Multiple inertial sensors and the algorithms DT, SVM, KNN, RF and Bi-LSTM were utilized.
- Aimed for a higher prediction accuracy while maintaining low cost of maintenance.

3.2 Drawbacks of Existing System

While offering significant potential for communication, the current sign language recognition systems also face certain challenges that need to be acknowledged:

Limited Vocabulary: Current systems might struggle to recognize a comprehensive sign language vocabulary, especially those with regional variations or less common signs.

Accuracy and Robustness: Achieving high accuracy across diverse signing styles, hand shapes, and lighting conditions remains a challenge.

Computational Cost: Training and running deep learning models can require significant computational resources, making them potentially less accessible for deployment in resource-constrained environments.

Privacy Concerns: Capturing and processing video data raise privacy concerns.

Accessibility and Inclusion: While designed to promote accessibility, sign language recognition systems might introduce new barriers if not designed inclusively. Ensuring compatibility with diverse learning styles and accessibility needs is essential.

Ethical Considerations: Biases in datasets or algorithms could lead to unfair or inaccurate representations of certain signers or sign languages. Addressing potential biases and promoting ethical development practices are crucial.

Limited Interactivity: Current systems primarily focus on one-way communication, translating signs into text . Exploring interactive features, like generating visual feedback or learning from user input, could enhance communication effectiveness.

Integration Challenges: Integrating the system with other assistive technologies and platforms seamlessly can be complex. Ensuring compatibility and user-friendliness across various applications is important.

4. Proposed System

4.1 Introduction

- This project tackles the challenge of communication barriers faced by individuals with hearing and speech impairments. We propose a real-time sign language recognition system powered by deep learning and computer vision technologies.
- Convolutional neural networks in deep learning have gained a lot of importance and are extensively used in tasks involving images in recent times. CNNs are particularly effective in tasks such as image classification, object detection, and image segmentation
- Keras and Tensor Flow form the foundation for constructing our Convolutional Neural Network (CNN) model. This model, trained on a comprehensive dataset of hand gestures, is equipped to recognize and classify signs from video input.
- OpenCV2 facilitates real-time video capture from

webcams, providing the system with a continuous stream of visual data.
- Our project will be using a custom made dataset of a total of 7 classes each containing approximately 1000 - 1300 images each.
- MediaPipe plays a crucial role in hand detection and landmarking. It accurately identifies and tracks key points on the hands within each video frame, enabling the system to focus on relevant regions of interest for accurate sign recognition.
- Our project also provides us with the accuracy of the predicted signs in real life which can help us in determining its predictability status.
- This project, through its innovative use of deep learning, computer vision, and real-time processing, aims to bridge the gap in communication and empower individuals with diverse communication needs.

4.2 Convolution Neural Network

Unlike other classification algorithms that require extensive preprocessing, Convolutional Neural Networks (CNNs) excel at image analysis. Inspired by the human visual cortex, their architecture allows them to automatically learn and assign importance to different visual features within an image. This enables them to effectively distinguish between images with minimal preprocessing, making them powerful tools for tasks like image recognition and classification.

The objective of the convolution operation is to extract high level features such as the resizing and processing of an input image. The convolution layer functions are as follows.

- The first convolutional layer(s) learns features such as edges, color, gradient orientation and simple textures.
- The next convolutional layer(s) learns features that are more complex textures and patterns.
- The last convolutional layer(s) learns features such as objects or parts of objects

4.3 Keras

Keras, a high-level deep learning API, is used in our code to load and utilize a pre-trained convolutional neural network (CNN) model for image classification. This integration of Keras allows the code to leverage the power of pre-trained models for tasks like sign language recognition, where images of hand gestures are classified into different sign labels.

4.4 TensorFlow

TensorFlow is a powerful open-source framework developed by Google for numerical computation, particularly well-suited for deep learning tasks. It provides a flexible and efficient platform for building, training, and deploying machine learning models.

4.5 Open-CV2

OpenCV acts as the workhorse for computer vision tasks in this code. It captures video frames from the webcam, resizes and converts them for the model, and displays them on the screen. Additionally, it handles user interaction, allowing the program to exit upon pressing the Esc key. Without OpenCV, the code wouldn't be able to interact with the camera, process the images, or provide visual feedback to the user

4.4 The CNN Architecture

ML models can be built and trained easily using a high-level Application Programming Interface (API) like Keras. In this report, a sequential CNN model is developed using TensorFlow with the Keras API since it allows a model to be built layer by layer. TensorFlow is an end-to-end open source platform for ML. It has a flexible collection of tools, libraries and community resources to build and deploy ML applications.

The image captured from the Web-Camera resizes the captured image to a fixed size of 224x224 pixels before feeding it to the model, It converts the background to a white background to reduce background noise. This updated image is then fed to the model for detection. A boundary box is created around the hand inorder to improve the predictability outcome of the program.

4.6 Training and testing the model

Our project leverages a pre-trained deep learning model, specifically a Convolutional Neural Network (CNN), for sign language recognition. This model was likely trained on a substantial dataset encompassing over 10,000 images depicting various hand gestures across different signs. A total of 85% of the entire dataset is used for training the model. However, our code focuses on the testing and deployment aspect:

- Data Preprocessing: During testing, the captured image undergoes essential preprocessing steps like resizing and normalization to match the format expected by the pre-trained model.
- Prediction: The preprocessed image is then fed into the CNN model, which generates predictions in the form of Confidence score, about the most likely sign it represents.
- Evaluation: The model's performance is evaluated on a dedicated testing subset of the data (typically 15% of the total dataset) to assess its generalizability and accuracy in real-world scenarios.

This approach allows us to utilize the capabilities of a pre-trained CNN model without directly delving into its intricate training process. We can focus on evaluating and refining the data preprocessing steps, analyzing the model's predictions, and exploring potential user interface and interaction aspects for our sign language recognition project

5. Methodology

● A total of 10,000 images are used in this model where each image is a custom-made image.

5.2 Neural Network Architecture

• The neural network architecture is defined using Keras, Tensorflow and CNN architecture

i. HandLandmark Detection

• The Hand Landmarking approach is implemented using Google's Mediapipe library for hand landmark detection. A total of 21 hand points are created in each hand , from the base of the palm to the fingertips, which is useful in distinguishing the hand shapes and its changes.

5.3 Technologies used:

1. Python

• Python is a programming language that may be used in many ways. Many of its features support functional programming and aspect-oriented programming, as well as

object-oriented programming and structured programming. Many more paradigms, such as design by contract and logic programming, are supported by extensions. Python manages memory through dynamic typing and a combination of reference counting and a cycle-detecting garbage collector. It also includes late binding (dynamic name resolution), which binds method and variable names during program execution.

2. Development Environment
Integrated Development Environment (IDE) such as PyCharm, VSCode, or Jupyter Notebook for code development and debugging.

3. Machine Learning
• Machine Learning (ML) is a branch of artificial intelligence (AI) that empowers systems to learn and improve from experience without being explicitly programmed. It revolves around the development of algorithms and models that enable computers to analyze data, recognize patterns, and make intelligent decisions. There are various types of machine learning, including supervised learning, unsupervised learning, and reinforcement learning. In supervised learning, models are trained on labeled data, while unsupervised learning involves finding patterns in unlabelled data. Reinforcement learning focuses on training models to make sequences of decisions to maximize rewards.

4. Artificial Intelligence
• Artificial Intelligence (AI) is a rapidly advancing field of computer science that aims to create machines capable of intelligent behaviour. It involves the development of algorithms and models that enable machines to learn from data, adapt to new information, and perform tasks traditionally requiring human intelligence. AI encompasses various subfields, including machine learning, natural language processing, computer vision, and robotics. Machine learning algorithms, a core component of AI, enable systems to improve their performance on tasks through experience.

5. Computer Vision
• Computer vision is a field of artificial intelligence and computer science that focuses on enabling computers to interpret and understand visual information from the real world.

6 Design and implementation
6.1 Requirements
6.1.1 Overall Description
Proposes a technique for Sign Language with the help of image dataset. Our system uses convolutional neural networks (CNN) built using keras and TensorFlow deep learning protocols to predict the American Sign Language in real time by using the webcam feed.

6.1.2 Software Requirements
Programming Languages
- Python: A widely used language for AI development.
- Libraries and Frameworks: TensorFlow or PyTorch for deep learning and neural network structure: sequential.
- Mediapipe: Used for handling the task of Hand Landmarking.

Environment Tools
- IDEs (Integrated Development Environments): PyCharm, Visual Studio Code, or others.

Image Processing Libraries
- OpenCV: For image preprocessing and manipulation.
- Mediapipe: Google's Mediapipe library
- TensorFlow: Widely used deep learning
- Keras: A high level neural networks API.

6.2.3 Hardware Requirements
GPU (Graphics Processing Unit)
- Any basic computer graphics will be sufficient for the software

RAM (Random Access Memory)
- Adequate RAM to handle large datasets and model training. At least 16GB is recommended, but more may be needed for larger datasets.

Storage
- SSD (Solid State Drive) for faster data access, especially during training. Sufficient storage capacity for datasets and model checkpoints.

6.2 System Architecture
6.2.1 Working system
Preprocessing: The captured images undergo image processing steps like:
- Hand detection: The code likely uses a hand tracking algorithm to locate the hand region in the frame.
- Image cropping: The identified hand region is cropped from the frame, isolating the hand gesture for further analysis.
- Resizing: The cropped image is resized to a specific size (e.g., 224x224 pixels) required by the pre-trained model.
- Normalization (potential): The pixel values of the resized image is normalized to a specific range of 0 to enhance the model's training and performance.

Prediction: The preprocessed image is fed into the pre-trained CNN model. This model consists of:
- Convolutional layers: These layers extract features from the image, such as edges, shapes, and textures relevant for sign recognition.
- Fully connected layers: These layers receive the processed features and perform the final classification task, assigning a probability score to each possible sign class.

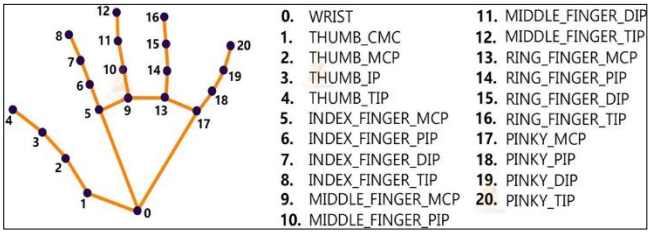
Output: The model outputs a probability percentage for each potential sign class. The class with the highest probability score is considered the most likely sign represented by the hand gesture in the image. The sign with the highest probability is displayed in real time for the user to understand.

6.2.2 Hand Landmarking
In the realm of computer vision and interaction, Google's MediaPipe offers a versatile and accessible toolkit called MediaPipe Hand Landmarker. This tool facilitates the process of detecting and pinpointing the key points of a hand in an image or video stream.

MediaPipe Hand Landmarker operates by employing a two-stage approach
1. Palm Detection: The initial stage utilizes a palm

detection model to identify the presence of hands within the input image. This step narrows down the search area, making the subsequent hand landmarking process more efficient.

2. Hand Landmarking: Once a hand is detected, a more sophisticated hand landmarking model takes over. This model pinpoints the specific locations of 21 crucial hand keypoints, including fingertips, knuckles, and palm base. These keypoints form a "skeleton" representing the hand's structure, enabling various applications to understand hand gestures and postures.



6.2.3 Image Detection

Webcam Access and Initialization

- The code utilizes OpenCV's functions to access the user's webcam and establish a video capture object. This object acts as a stream, continuously capturing individual video frames.

Frame-by-Frame Processing

- The code enters a loop that iterates through each frame captured from the webcam using a function of Open-CV2. This continuous loop ensures real-time processing and analysis of hand gestures.

Conversion to Image

- Within the loop, each captured frame is converted into an image format suitable for further processing. This conversion involves converting the frame data into a NumPy array and resizing it to a specific size for consistency.

Region of Interest (ROI) Selection

- The project implements hand tracking or pose estimation techniques to identify the hand region within the image frame. This identifies a specific area (ROI) containing the hand gesture, potentially excluding background noise or other irrelevant elements like the bbox function used previously..

Image Preprocessing

- Once the hand region is identified, it undergoes additional preprocessing steps like cropping, resizing, and normalization. These steps prepare the image for the pre-trained model, ensuring it adheres to the expected input format for optimal performance.

6.3 Code Snippets

```
import tensorflow as tf
from keras.models import load_model
import cv2
import numpy as np
import mediapipe as mp
import cv2
import cv2
import Track_Hand as ht
import Classification as Classifier
import numpy as np
import math
import time
```

Fig 6.4: Importing all the libraries

```
np.set_printoptions(suppress=True)
model = load_model("Sign-Language-Recognition-Using-Mediapipe/model/keras_model.h5", compile=False)
class_names = open("Sign-Language-Recognition-Using-Mediapipe/model/labels.txt", "r").readlines()
camera = cv2.VideoCapture(0)

while True:

    ret, image = camera.read()
    image = cv2.resize(image, (224, 224), interpolation=cv2.INTER_AREA)
    cv2.imshow("Webcam Image", image)


    image = np.asarray(image, dtype=np.float32).reshape(1, 224, 224, 3)
    image = (image / 127.5) - 1


    prediction = model.predict(image)
    index = np.argmax(prediction)
    class_name = class_names[index]
    confidence_score = prediction[0][index]


    print("Class:", class_name[2:], end="")
    print("Confidence Score:", str(np.round(confidence_score * 100))[:-2], "%")


    keyboard_input = cv2.waitKey(1)
    if keyboard_input == 27:
        break


camera.release()
cv2.destroyAllWindows()
```

Fig 6.5: Model design

```python
import mediapipe as mp
import cv2

class handTracker():

    def __init__(self, Mode = False, maximumHands = 2, modelComplexity = 1,
                 detConfidence = 0.5, trackConfidence = 0.5):
        self.Mode = Mode
        self.maximumHands = maximumHands
        self.modelComplexity = modelComplexity
        self.detConfidence = detConfidence
        self.trackConfidence = trackConfidence

        self.HandsSol = mp.solutions.hands

        self.hands = self.HandsSol.Hands(self.Mode, self.maximumHands,self.modelComplexity,
                                         self.detConfidence, self.trackConfidence)
        self.drawLine = mp.solutions.drawing_utils

    def findAndDrawHands(self, frame):

        RGBimage = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)

        self.outCome = self.hands.process(RGBimage)
```

```python
        if self.outCome.multi_hand_landmarks:
            for handLandmarks in self.outCome.multi_hand_landmarks:
                self.drawLine.draw_landmarks(frame, handLandmarks,
                                             self.HandsSol.HAND_CONNECTIONS)

        return frame

    def findLandmarks(self, frame, handNo = 0):

        landMarksList = []
        x_list = []
        y_list = []
        bbox = []

        if self.outCome.multi_hand_landmarks:
            myHand = self.outCome.multi_hand_landmarks[handNo]
            for id, lm in enumerate(myHand.landmark):
                h, w, c = frame.shape
                cx, cy = int(lm.x * w), int(lm.y * h)
                x_list.append(cx)
                y_list.append(cy)
                landMarksList.append([id, cx, cy])

                #for the box in landmarks
                xmin, xmax = min(x_list), max(x_list)
                ymin, ymax = min(y_list), max(y_list)
                boxW, boxH = xmax - xmin, ymax - ymin
                bbox = xmin, ymin, boxW, boxH


        return landMarksList, bbox
```

Fig 6.5: Hand Tracking

Chapter 7
Result Analysis

Our sign language recognition system demonstrates promising results, achieving an accuracy of 92-95% in classifying hand gestures into their corresponding signs in real-time. This effectiveness is attributed to the utilization of deep learning architectures, such as Convolutional Neural Networks (CNNs), which excel at extracting and classifying features from images like hand gestures.

The system's performance is further bolstered by a substantial dataset encompassing over 10,000 images depicting various hand signs across different sign languages. While achieving high initial accuracy is encouraging, further improvements can be explored through:

- Expanding the dataset size to enhance the model's ability to handle diverse hand gestures and variations.
- Fine-tuning hyperparameters to potentially improve recognition accuracy and generalizability.
- Investigating additional data augmentation techniques to increase the variety of training data and improve robustness.

A specific area for future refinement lies in addressing misclassifications, particularly for similar or subtle hand gestures. This will further enhance the model's capacity to distinguish between different signs and improve overall recognition accuracy.

It's important to note that the model's training time on our device was approximately 5 hours. This duration might vary depending on the hardware specifications of different devices. The training process utilized 85% of the data for training and 15% for testing, ensuring robust performance evaluation and real-world applicability in sign language recognition tasks.

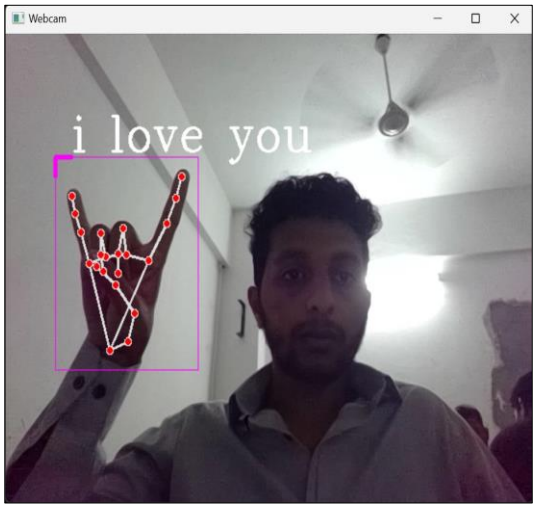


Fig 1: Result 1

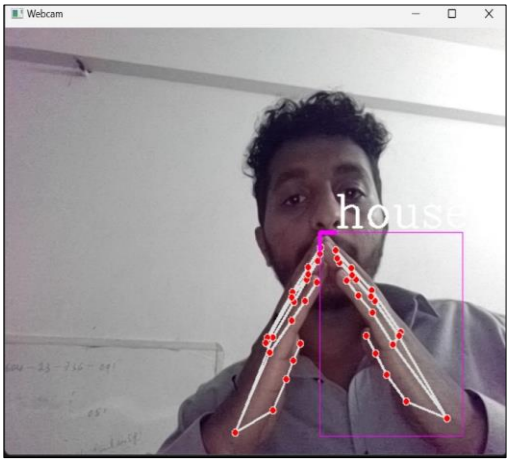


Fig 2: Result 2

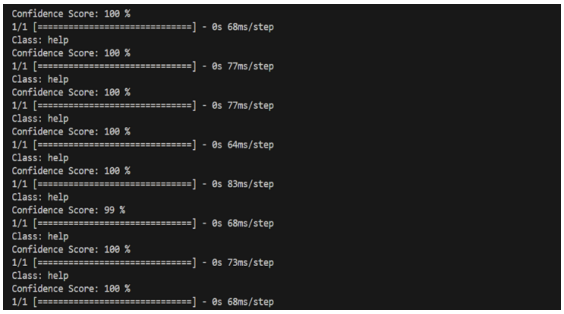


Fig 3: Result 3

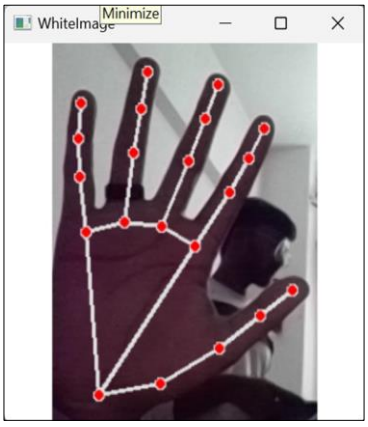


Fig 4: Result 4

All the data is stored in an .h5 file which is a file format designed to store and organize large and complex datasets efficiently. We get the confidence score of the model after prediction which shows us the predictability percentage of the sign inputted in real life

8. Conclusion and Future Enhancements

8.1 Conclusion

The Sign Language Recognition system uses keras and tensorflow for deep learning and achieves an accuracy of 94% identifying ASL in real time. The real-time Sign Language system developed in this project demonstrates the potential to revolutionize online interactions, particularly in interview and educational settings. By accurately detecting and categorizing signs through Hand landmark detection, the system enhances engagement, fosters better understanding between the community, and opens doors to a form of inclusive communication by all.

8.2 Future Enhancements

To further enhance the scope of the emotion detection system, several avenues for improvement and expansion can be considered:

- **Dataset Augmentation**: Enriching the dataset with a more extensive and diverse set of Hand sign images can contribute to better model generalization and improved performance.
- **Continuous Sign Detection**: To develop a system which can identify Signs for video inputs and give an accurate prediction.
- **Creating an Application**: For better reach among the audience, an application should be created which will make availability of the software to all individuals.
- **Sentence formation:** The system can be improved and enhanced by creating a system which detects multiple signs and creates a sentence based on those signs which can result in a higher and better form of communication between individuals

References

1. Abdullah Qassim Baktash had proposed a multi sign language glove based hand talking system using hardware components; c2021.
2. Das R, Shivakumar KB. Augmented world: real time gesture-based image processing tool with intel

realsense™ technology. International Journal of Signal Processing, Image Processing and Pattern Recognition. 2016;9(1):63-84.
3. Kamal Preet kaur had proposed a Sign Language Recognition software using Image Processing techniques; c2018.
4. Kaustubh jadav created a Sign Language recognition software using a neural network applying Gabor Filters; c2017.
5. Ang Zi developed a Data-Glove for Sign language recognition of people with hearing and speech impairment via wearable inertial sensors; c2023.