A PROJECT REPORT

on

"SIGN LANGUAGE DETECTION USING LANDMARKING(RANDOM FORESTS)"

Submitted to

KIIT Deemed to be University

In Partial Fulfilment of the Requirement for the Award of

**BACHELOR'S DEGREE IN
COMPUTER SCIENCE AND ENGINEERING**

BY

PARITOSH KUMAR	21051666
ANANYA MUKHERJEE	21051628
PRIYANSHU SINHA	21051673
VATSAL SAXENA	21051698

**UNDER THE GUIDANCE OF
PROF. JHALAK HOTA**

SCHOOL OF COMPUTER ENGINEERING

KALINGA INSTITUTE OF INDUSTRIAL TECHNOLOGY

BHUBANESWAR, ODISHA - 751024

May 2020

KIIT Deemed to be University

School of Computer Engineering

Bhubaneswar, ODISHA 751024

CERTIFICATE

This is certify that the project entitled

"SIGN LANGUAGE DETECTION USING LANDMARKING(RANDOM FORESTS)"

submitted by

PARITOSH KUMAR	21051666
ANANYA MUKHERJEE	21051628
PRIYANSHU SINHA	21051673
VATSAL SAXENA	21051698

is a record of bonafide work carried out by them, in the partial fulfilment of the requirement for the award of Degree of Bachelor of Engineering (Computer Science & Engineering) at KIIT Deemed to be university, Bhubaneswar. This work is done during year 2022-2023, under our guidance.

Date: / /

PROF. JHALAK HOTA

Acknowledgements

We are profoundly grateful to **PROF. JHALAK HOTA** of **Affiliation** for his expert guidance and continuous encouragement throughout to see that this project rights its target since its commencement to its completion. ....

ANANYA MUKHERJEE(21051628)
PARITOSH KUMAR(21051666)
PRIYANSHU SINHA(21051673)
VATSAL SAXENA(21051698)

ABSTRACT

This project delves into the confluence of machine learning methodologies and sign language detection, with the explicit objective of creating seamless communication channels for individuals with speech impediments. Sign language, as a visually expressive language contingent upon intricate hand gestures and movements, presents a multifaceted challenge for automated recognition systems. Beyond the foundational sign language recognition system, this project explores additional use cases involving integration with web and Android applications. For instance, the integration with web applications can enable interactive sign language tutorials and games, fostering learning and engagement among users. Similarly, integration with Android applications can facilitate real-world applications such as sign language translation apps, aiding in communication between individuals with and without speech disabilities.

Machine learning algorithms, notably the Random Forest classifier utilized in this project, are harnessed to discern and interpret the nuanced hand gestures inherent in sign language. By training the model on a meticulously curated dataset comprising hand gesture images sourced from webcam feeds, the system assimilates intricate correlations between specific hand landmarks and their corresponding sign language gestures. At the core of the project's methodology lies the utilization of the libraries like OpenCV and MediaPipe, which enable precise hand landmark detection and the extraction of salient features from dynamic hand movements. These extracted features serve as discriminative inputs for the Random Forest classifier, facilitating real-time prediction of sign language gestures. Through this sophisticated integration of machine learning paradigms, hand landmarking techniques, and application development, the project aims to bridge gaps in communication and foster inclusivity and accessibility in various digital domains. The potential for creating interactive sign language-based games and educational tools further underscores the versatility and societal impact of this integrated approach.

Keywords: ***Machine Learning, Sign Language, Random Forests, MediaPipe, OpenCV, Landmarking***.

Contents

List of Figures

Chapter 1

Introduction

This project addresses the critical need for an advanced sign language detection system, particularly for individuals with speech impairments who rely on sign language as their primary mode of communication. Current solutions in automated sign language recognition often lack accuracy, real-time responsiveness, and versatility, hindering effective communication and inclusivity.

The project leverages machine learning, specifically the Random Forest classifier, and advanced hand landmarking techniques using the MediaPipe library. By training the model on a diverse dataset of hand gesture images, captured through webcam feeds via OpenCV, the system can decipher subtle variations in hand movements that constitute different sign language gestures.

The primary objective of the project is to develop a straightforward and highly integrable sign recognition model that seamlessly integrates into various applications.

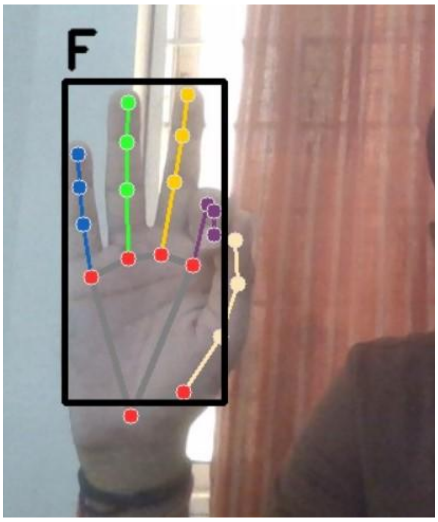


Figure 1.1: Desired and Obtained Output.

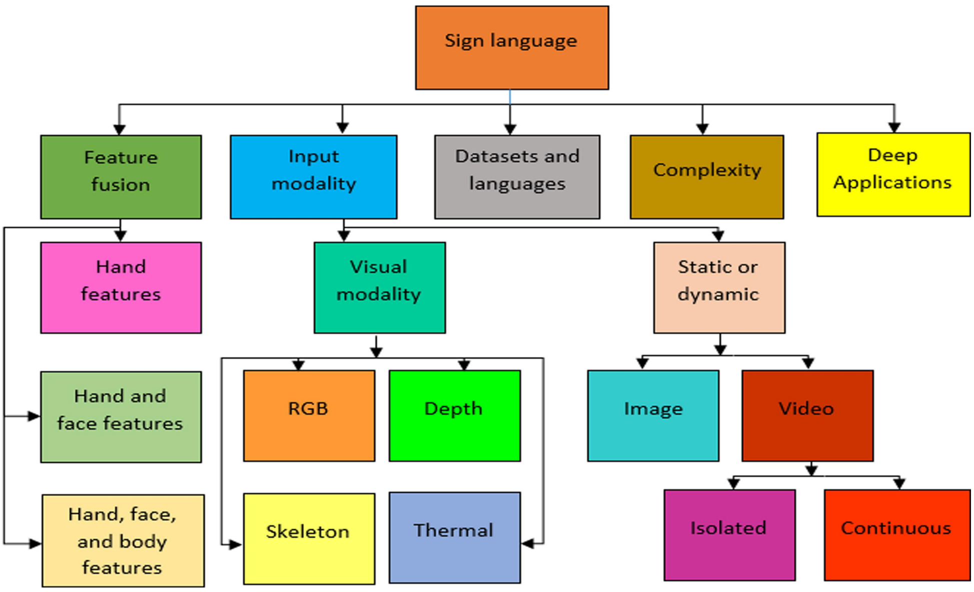
Chapter 2

Basic Concepts/ Literature Review

The project encapsulates the extensive literature and conceptual frameworks relevant to the project, drawing upon numerous scholarly references to provide a comprehensive understanding of the subject matter.

2.1 Sign Language Detection

[1]Sign language detection using machine learning or deep learning models can be approached in diverse ways, each tailored to the specific features, input types, complexity, and intended application of the model. The approach taken may vary based on the following considerations:



[1]Figure 2.1: Taxonomy Of Model Considerations.

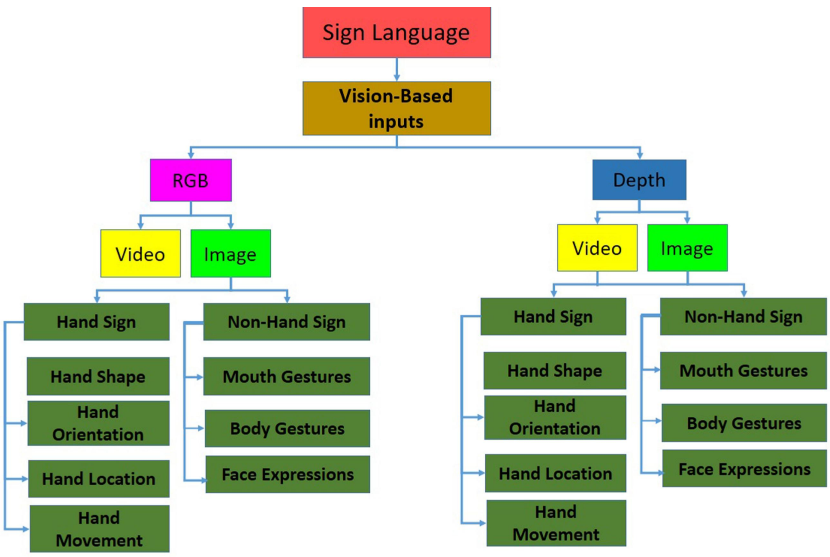
Features Selection: Different models may focus on distinct features of sign language, such as hand gestures, finger movements, facial expressions, or body posture. The selection of features depends on the granularity and specificity required for accurate detection and interpretation of sign language gestures.

Input Types: The type of input data can significantly impact the design of the model. Input types may include video streams, image frames, sensor data from wearable devices, or a combination of these inputs. The choice of input type influences the preprocessing steps, model architecture, and overall performance of the sign language detection system.

Model Complexity: The complexity of the ML/DL model can vary from simple classifiers to deep neural networks with multiple layers. The level of complexity is often determined by the intricacy of sign language gestures to be recognized, the size and diversity of the dataset, and the desired accuracy and efficiency of the model.

Application Context: The intended application of the sign language detection model also guides the approach. For instance, a real-time application for translating sign language into text or speech requires a fast and responsive model, while a research-oriented project may focus on exploring advanced architectures and techniques for improved accuracy.

Also Vision-based models for sign language detection can vary based on **input type (image or video)** and utilize techniques like CNNs for image-based models and RNNs, LSTMs, or TCNs for video-based models. They capture **hand signs, shapes, orientations, locations, non-hand signs, mouth gestures, body gestures, and facial expressions.** Depth-related inputs enhance spatial perception. Multi-modal fusion integrates different modalities, improving accuracy. Considerations include lighting, noise, hand variations for robustness. These models, with advanced techniques and neural network architectures, achieve accurate and versatile sign language detection.



[1]Figure 2.2: Vision Based Sign Language Models

2.2 Ensemble Learning And Random Forests

[2]Ensemble learning is a machine learning technique that combines multiple individual models to create a more robust and accurate predictive model. Random Forests, a popular ensemble learning method, consist of an ensemble of decision trees. Each decision tree in the Random Forest independently predicts the output, and the final prediction is determined through a voting mechanism (for classification) or averaging (for regression) of the individual tree predictions.

The following are the steps followed while implementing the random forests approach:

1. Decision Trees:

Decision trees are represented as a set of if-else conditions based on feature values. Given a feature vector $X=(X_1, X_2, \ldots, X_n)$, a decision tree predicts the output Y by traversing the tree based on the conditions at each node. For example, a simple decision tree rule might be:

$$\text{If } X_1>0.5 \text{ and } X_2<= 0.3 \text{ then } Y=1$$

Decision trees partition the feature space into regions based on these rules.

2. Bootstrapping:

Random Forests use bootstrapping to create multiple datasets from the original data. Let's denote the original dataset as $D=\{(X_1,Y_1),(X_2,Y_2), \ldots ,(X_m,Y_m)\}$, where X_i is the feature vector and Y_i is the corresponding label. Bootstrapping creates subsets of D by sampling with replacement, resulting in datasets $D_1, D_2, \ldots, D_k$.

3. Random Subset of Features:

At each node of a decision tree in a Random Forest, only a random subset of features is considered for splitting. Let's say we have n total features, and at each node, we randomly select m features $(m<n)$ for splitting. This random subset of features helps in decorrelating the trees and adding diversity.

4. Voting Mechanism:

In classification tasks, each decision tree in the Random Forest independently predicts the output class. Let C_i be the predicted class by the ***i-th tree***. The final prediction Y_{final} is determined by a majority vote:
For regression tasks, the final prediction is the average of the individual tree predictions:

$$Y_{final} = \text{mode}(C_1, C_2, \ldots, C_k)$$

For regression tasks, the final prediction is the average of the individual tree predictions.

5. Feature Importance:

Random Forests provide feature importance scores based on how much each feature reduces impurity (e.g., Gini impurity or entropy) across all the trees. Features with higher importance scores contribute more to the model's predictions.

2.3 About Mediapipe Hands

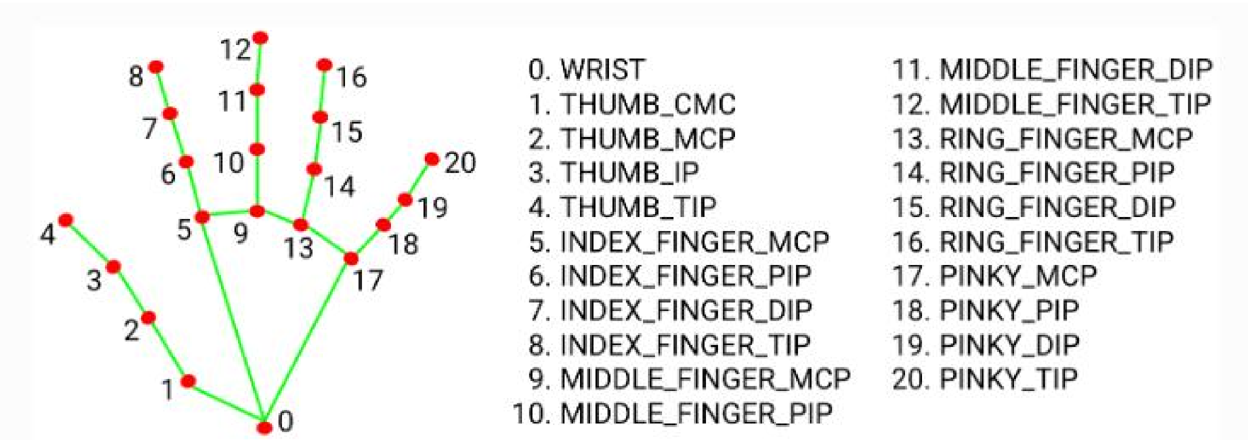
[3]MediaPipe Hands is an advanced hand and finger tracking solution designed to perceive the shape and motion of hands in real-time. It employs machine learning to infer 21 3D landmarks of a hand from a single frame, making it suitable for various technological applications such as sign language understanding, hand gesture control, and augmented reality. The underlying models of this library are:

Palm Detection Model:

MediaPipe employs a single-shot detector model optimized for mobile real-time use to detect initial hand locations.
A palm detector model is trained instead of a hand detector, as estimating bounding boxes of rigid objects like palms is simpler.
Techniques such as encoder-decoder feature extractor and minimizing focal loss during training are used to achieve high precision (95.7%) in palm detection.

Hand Landmark Model:

After palm detection, MediaPipe's hand landmark model performs precise keypoint localization of 21 3D hand-knuckle coordinates inside the detected hand regions via regression.
The model learns a consistent internal hand pose representation and is robust to partially visible hands and self-occlusions.
Ground truth data is obtained through manual annotation of real-world images and rendering a synthetic hand model over various backgrounds, providing comprehensive supervision on hand geometry.



[3]Figure 2.3: Landmarking Index For mp.hands

Chapter 3

Problem Statement / Requirement Specifications

The problem addressed by this project is the need for a straightforward, efficient, and accurate recognition system for sign language. This recognition system aims to provide a highly integrable solution that aids individuals who are deaf or hard of hearing in communicating through sign language. Furthermore, the system serves as a foundational model to facilitate the development of additional applications leveraging sign language recognition technology.

3.1 Project Planning

3.1.1 Defining Project Objectives:

The project objectives are as follows:
1. Develop an accurate and efficient sign language recognition system.
2. Create a user-friendly interface for seamless communication using sign language.
3. Ensure real-time recognition capabilities for smooth interaction.
4. Establish a flexible framework for future application development.
5. Provide comprehensive documentation and support.
6. Conduct rigorous testing to validate accuracy and effectiveness.
7. Foster collaboration with stakeholders for continuous improvement.

3.1.2 Establishing project timeline:

1. *Project Initiation and Approval :* Started: 18/1/24
   Approved by Mentor: 2/2/24

2. *Ideation Phase:* Ideation Period: Until 28/2/24

3. *Development Phase:* Start of Development: 8/3/24
   Prototype Completion: 20/3/24

4. *Refinement and Finalization:* Finalization of Project: 2/4/24

3.1.3 Resource Allocation:

1. *Human Resources:*
- Vatsal and Ananya: Model training and inference.
- Paritosh and Priyanshu: Dataset harvesting and image capturing.

2. *Material Resources:*
- Equipment: Computers for development and testing and Cameras for image capturing.
- Software: PyCharm(for development environment) MediaPipe, OpenCV (cv2), pickle, numpy, scikit-learn (for model implementation.)
- External Resources: MediaPipe library(for hand detection and landmark estimation), Python libraries(for machine learning and data manipulation)

3. *Success criteria:*
   The success criteria for the project entail achieving high accuracy and correct classification alongside fast processing capabilities for real-time sign language interpretation.

3.2 Project Analysis

3.2.1 Gathering Requirements:

The requirements of this model include:
1. A clear visual interpretation window to show real time sign detection.
2. Easy to integrate in other applications.
3. Fast and lightweight processing.

3.2.2 Analyzing Feasibility: The requirements mentions are feasible and very much attainable with existing technology.

3.2.3 Identify Risks: No such risks were found during the implementation of this project.

3.3 System Design

3.3.1 Design Constraints:

The Sign Language Detection system using Landmarking and Random Forests Classifier will operate within the following design constraints:

Software Requirements:

- ✧ The system will be coded in Python, utilizing key libraries like OpenCV, Matplotlib, Scikit Learn, and MediaPipe.
- ✧ These libraries will be employed for tasks such as image capturing, landmarking, and real-time image detection.

Hardware Requirements:

- ✧ The system will operate on standard computing hardware with sufficient processing power and memory capabilities.
- ✧ This ensures efficient performance of the system during runtime.

Experimental Setup:

- ✧ No specific constraints or environmental requirements are foreseen for the operation of the system.
- ✧ The system is expected to function effectively under diverse experimental conditions without specific environmental limitations.

3.3.2 System Architecture Or Block Diagram

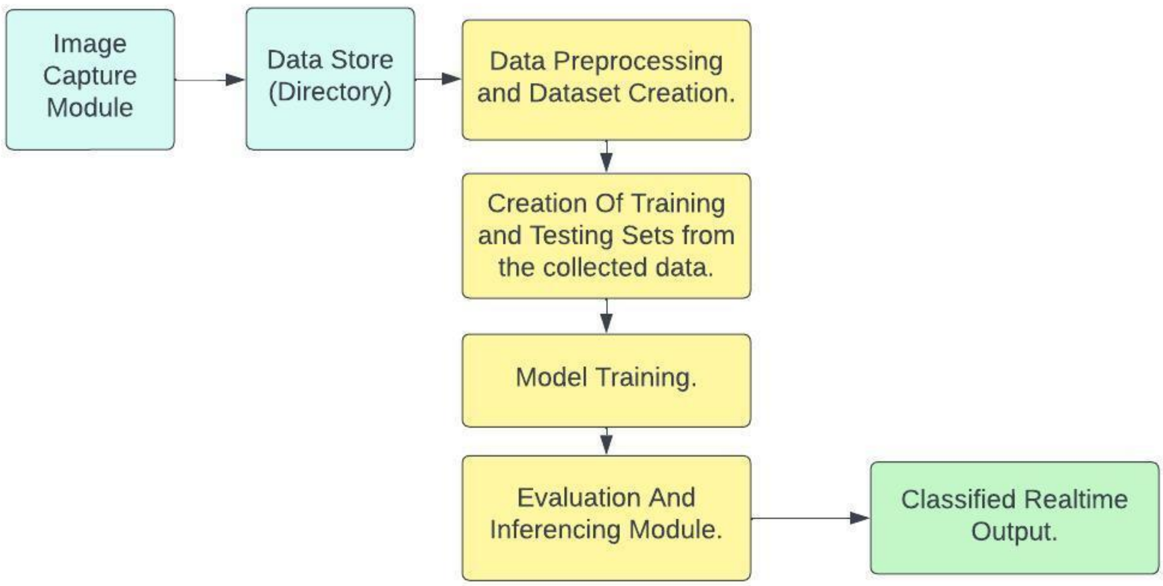


Figure 3.1: ML Design Pipeline Of The Project

Chapter 4
Implementation

The project aims to create a robust system for accurately recognizing sign language, ensuring it is efficient, straightforward, and easily integrated into various applications. This technology will serve as a foundational tool for developing a range of applications tailored to assist the deaf and mute community, empowering them with enhanced communication and accessibility features.

4.1 Methodology

4.1.1 Machine Learning Algorithm:
For this project, Random Forest Classifier is used as the machine learning algorithm. Random Forest is an ensemble learning method that operates by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes (classification) or mean prediction (regression) of the individual trees. It generally provides higher accuracy compared to individual decision trees and is robust to overfitting.

4.1.2 Landmark Detection:
Landmark detection is performed using MediaPipe, which is a popular library for various tasks in computer vision, including hand landmark detection. MediaPipe provides pre-trained models that can detect landmarks on hands in real-time. These landmarks are key points on the hand, such as fingertips, knuckles, etc., and they serve as features for training the machine learning model.

4.1.3. Individual Steps:
The project is divided into four Python files which are later integrated to implement the project.

collect_images.py file (DATA COLLECTION PHASE)

- Objective: This script is responsible for collecting images of sign language gestures using the webcam.

- Implementation: It utilizes OpenCV to access the webcam and capture images. It provides instructions for the user to perform different sign language gestures and captures images accordingly.

- Reasoning: Separating the image collection process into its own script allows for focused development and testing of this specific functionality. It also makes it easier to modify or expand this part of the project without affecting other components.

- Libraries Used: OpenCV (cv2).

- Objects Created: cap: An instance of cv2.VideoCapture() representing the webcam, used to capture images.

- Objects Used:

  - cv2.VideoCapture(): Object to access the webcam.
  - cv2.imshow(): Object to display captured images.
  - cv2.imwrite(): Object to save captured images.

- Expected Result: Images of sign language gestures are captured and saved.

create_dataset.py file (DATA PRE-PROCESSING PHASE)

- Objective: This script processes the collected images to extract landmark features using MediaPipe and creates a dataset for training the machine learning model.

- Implementation: It uses MediaPipe for landmark detection and OpenCV for image processing. It iterates over the collected images, extracts hand landmarks using MediaPipe, and saves the extracted features along with their corresponding labels.

- Reasoning: By separating dataset creation from image collection, it allows for flexibility in data preprocessing and feature extraction techniques. It also ensures that the dataset creation process can be easily reused or modified for different projects.

- Libraries Used: MediaPipe (mediapipe), OpenCV (cv2), pickle.

- Objects Created:

  - hands: An instance of mp_hands.Hands() representing the MediaPipe hands model, used for landmark detection.

- Objects Used:

  - mp_hands.Hands(): Object for hand landmark detection.
  - cv2.imread(): Object to read images from files.
  - cv2.cvtColor(): Object to convert images to RGB format.
  - pickle.dump(): Object to save data as a dataset.

- Expected Result: Landmark features are extracted from images and saved as a dataset.

train_classifier.py file (TRAINING PHASE OF THE MODEL)

- Objective: This script trains a machine learning classifier using the extracted landmark features and labels.

- Implementation: It employs scikit-learn to train a Random Forest classifier using the dataset created in the previous step. It splits the dataset into training and testing sets, trains the classifier, and evaluates its performance.

- Reasoning: Training the classifier in a separate script promotes modularity and allows for experimentation with different machine learning algorithms or hyperparameters without altering other parts of the project. It also facilitates the reuse of the trained model for inference.

- Machine Learning Algorithm used: Random Forest Classifier.

- Libraries Used: scikit-learn (sklearn), numpy (np), pickle.

- Objects Used:
    - RandomForestClassifier(): Object to initialize the Random Forest classifier.
    - train_test_split(): Object to split the dataset into training and testing sets.
    - model.fit(): Object to train the classifier.
    - accuracy_score(): Object to compute the accuracy of the classifier

- Expected Result: A trained Random Forest classifier model is saved.

inference_classifier.py file (PREDICTION PHASE)

- Objective: This script uses the trained classifier to make real-time predictions on sign language gestures captured by the webcam.

- Implementation: It utilizes MediaPipe for landmark detection and OpenCV for webcam access and real-time video processing. It continuously captures frames from the webcam, extracts hand landmarks using MediaPipe, and uses the trained classifier to predict the corresponding sign language gestures.

- Reasoning: Separating the inference process into its own script ensures that real-time predictions can be made independently of the training process. It also facilitates the deployment of the trained model for practical applications, such as gesture recognition systems.

- Machine Learning Algorithm: Random Forest Classifier (for inference).

- Libraries Used: MediaPipe (mediapipe), OpenCV (cv2), numpy (np), pickle.

- Objects Created:
  - hands: An instance of mp_hands.Hands() representing the MediaPipe hands model, used for landmark detection.
  - model: An instance of the trained Random Forest classifier.

- Objects Used:
  - cv2.VideoCapture(): Object to access the webcam.
  - mp_hands.Hands(): Object for hand landmark detection.
  - model.predict(): Object to make predictions using the trained classifier.
  - cv2.imshow(): Object to display images with predicted gestures.

- Expected Result: Real-time predictions of sign language gestures are made using the webcam.

4.1.4. Overall Model Integration:

The overall model is integrated by sequentially executing the four Python scripts: **collect_images.py -> create_dataset.py -> train_classifier.py -> inference_classifier.py.**

4.2 Testing OR Verification Plan

The overall testing plan covers different aspects of testing, including unit testing, integration testing, system testing, and acceptance testing, ensuring the correctness and reliability of the sign language detection system.

Following are the libraries used for testing:

OpenCV (cv2): Used for accessing the webcam, capturing frames, image processing, and displaying images.

MediaPipe (mediapipe): Used for hand landmark detection in real-time images.

scikit-learn (sklearn): Used for training machine learning models, such as Random Forest classifiers.

numpy (np): Used for numerical computations and data manipulation.
pickle: Used for serializing and deserializing Python objects, such as saving and loading trained machine learning models.

Test ID	**Test Case Title**	**Test Condition**	**System Behavior**	**Expected Result**
T01	Image Collection Test	Ensuring that images of sign language gestures are collected properly using the webcam.	The system captures images when prompted and saved them in the specified directory.	Images of sign language gestures are captured and saved without any errors.
T02	Landmark Extraction Test	Verifying that hand landmarks are correctly extracted from the collected images.	The system use MediaPipe to detect hand landmarks in each image and use matplotlib to plot them.	Landmarks representing key points on the hand (e.g., fingertips, knuckles) are accurately extracted from the images.
T03	Model Training Test	Ensuring that the machine learning model is trained properly using the extracted landmarks.	The system trains a Random Forest classifier using the extracted landmark features and corresponding labels.	The trained model achieves reasonable accuracy on the training data set and generalizes well to unseen data.
T04	Real-time Prediction Test	Verify that the system can make real-time predictions on sign language gestures captured by the webcam.	The system continuously captures images from the webcam, extract hand landmarks, and predict the corresponding gestures using the trained model.	The system accurately detects and classifies sign language gestures in real-time, displaying the predicted gestures overlaid on the webcam feed.

4.3 Result Analysis OR Screenshots

The project fulfils all the requirements specified.

The random forest model classifies the images very well with an accuracy of over 99.6%.

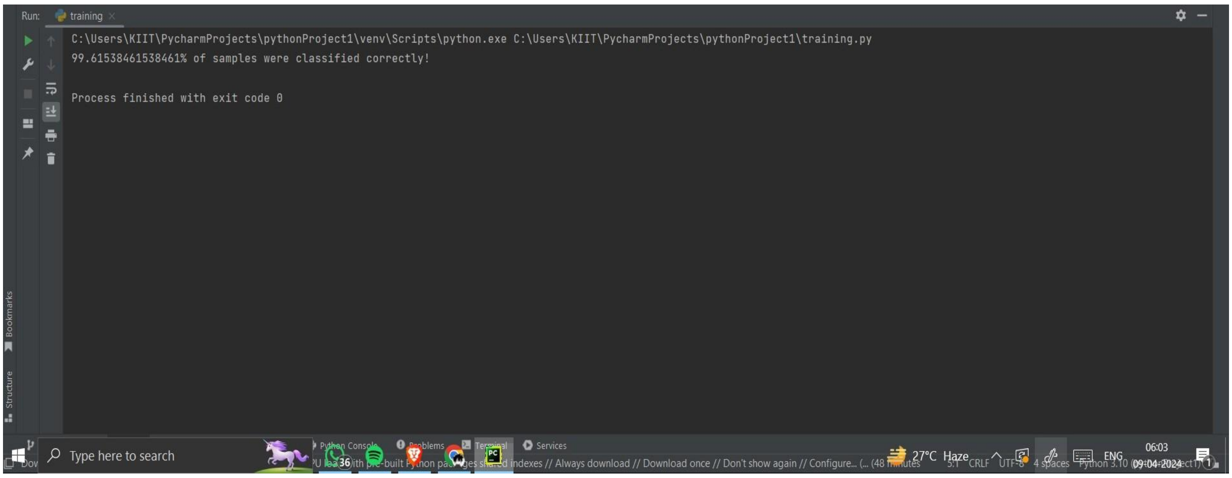


Figure 4.1: Obtained Accuracy

The land marking is done perfectly and the plots are well recognized by the classifier in real-time.

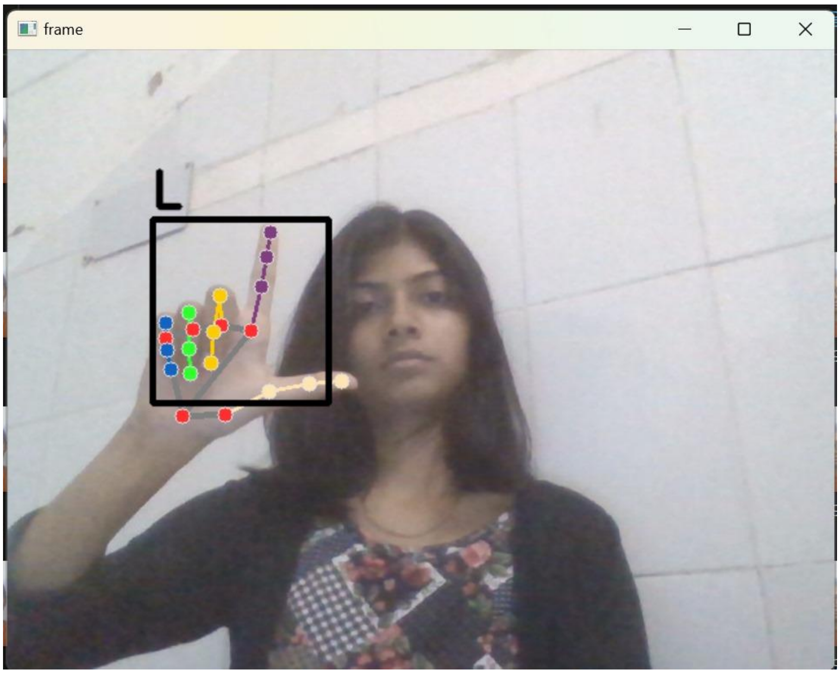


Figure 4.2 : Plotted Landmarkings using MediaPipe

Chapter 5

Standards Adopted

5.1 Design Standards

ISO/IEC 9126: Software Engineering - Product Quality: Used for defining quality characteristics and metrics, such as correctness, reliability, usability, efficiency, maintainability, and portability.

IEEE 1016: Software Design Descriptions: Followed for creating structured design descriptions, including architectural design and defining the design constraints.

5.2 Coding Standards

ISO/IEC/IEEE 12207: Software Life Cycle Processes: Referenced for defining coding standards within the software development life cycle, covering coding conventions, naming conventions, commenting guidelines, and documentation standards.

ISO/IEC 9126: Software Engineering - Product Quality: Ensured adherence to coding standards for aspects like readability, maintainability, portability, and compatibility across platforms.

IEEE 1028: Software Reviews and Audits: Conducted code reviews and audits to verify compliance with coding standards and identify areas for improvement.

5.3 Testing Standards

ISO/IEC/IEEE 29119: Software Testing Standards: Followed for defining testing processes, techniques, and documentation standards throughout the testing life cycle, including test planning, test design, test execution, and test reporting.

Chapter 6

Conclusion and Future Scope

6.1 Conclusion

In conclusion, this project has explored the feasibility and efficacy of sign language detection using computer vision techniques: computer vision techniques. The developed system has achieved an accuracy of 99.62% in recognizing various signs from the vocabulary. This accomplishment paves the way for further advancements in sign language communication and accessibility.

Future endeavours can focus on expanding the system's sign recognition repertoire and incorporating real-time translation capabilities. Additionally, integrating this technology into mobile applications or wearable devices could foster more inclusive communication bridges between the signing and non-signing communities.

As sign language detection technology continues to evolve, it holds the promise of revolutionizing how we interact with the deaf and hard-of-hearing population, fostering a more inclusive and equitable society.

6.2 Future Scope

Sign language detection technology can potentially bridge communication gaps in various real-world scenarios. Here are some ground breaking possibilities for the implementation of this model with related technologies:

Real-time translation: In lectures, conferences, or even casual conversations where sign language is being used. A sign language detection system could translate the signs into spoken or text in real-time, fostering inclusion and participation for deaf and hard-of-hearing individuals.
Educational tools: Interactive learning applications that incorporate sign language detection can provide a more engaging and accessible learning experience for deaf and hard-of-hearing students.
Improved accessibility: Public spaces like museums, transportation hubs, and hospitals can leverage sign language detection systems to provide information and assistance to deaf and hard-of-hearing visitors in their preferred communication method.
Virtual Assistant Compatibility: Integrate sign language detection into virtual assistants like Siri or Alexa. This allows deaf users to interact with technology using sign language, opening doors to information access and smart home control.
Augmented Reality (AR) Applications: Imagine AR glasses that overlay translated text or speech on top of sign language gestures. This can empower real-time communication between deaf and hearing people, even in noisy environments.

References

[1] “Sign Language Recognition: A Deep Survey” by Razieh Rastgoo, Kourosh Kiani, Sergio Escalera

[2] “Random Forests” by LEO BREIMAN, Statistics Department, University of California, Berkeley, CA 94720 on Springer.

[3] “MediaPipe Hands” on MediaPipe Read the docs

[4] OpenCV Documentation

INDIVIDUAL CONTRIBUTION REPORT:

SIGN LANGUAGE DETECTOR

ANANYA MUKHERJEE
21051628

Abstract:
Our project focuses on developing a system for sign language detection using machine learning, landmark detection, and the Random Forest algorithm. The aim is to create a robust solution that accurately recognizes sign language gestures in real-time, contributing to enhanced communication and accessibility for the deaf and mute community.

Individual contribution and findings:
I played a significant role in developing the training_classifier.py file for our group project on sign language detection. My responsibilities included data preprocessing, model training, and performance evaluation and optimization. I addressed challenges like handling noisy data, optimizing computational efficiency, and addressing class imbalances, and worked collaboratively with my peers to achieve promising results.

Overall, my contribution involved fine-tuning the Random Forest classifier's hyperparameters, evaluating its performance using appropriate metrics, and ensuring fair representation of all sign language gestures through preprocessing techniques like handling missing data, standardizing features, and addressing outliers. My technical findings underscored the importance of meticulous data preprocessing, the significance of hyperparameter tuning, and the synergistic benefits of collaborative problem-solving.

Individual contribution to project report preparation:
In the project report preparation, my contribution focused on "*Chapter 4: Implementation*". It includes the following:

1. Methodology
   A. Machine Learning Algorithm Used
   B. Landmark Detection Process
   C. Individual Steps Involved Data Collection, Preprocessing, Training and Inference
   D. Overall Model Integration
2. Testing or Verification Plan
3. Result Analysis and Screenshots

Individual contribution for project presentation and demonstration:
For the project presentation and demonstration, my individual contribution centered on elucidating the intricacies of the train_classifier.py file, which orchestrates the training of our sign language detection model.

Full Signature of Supervisor: ……………………………

Full signature of the Student: Ananya Mukherjee

SIGN LANGUAGE DETECTOR

PARITOSH KUMAR
21051666

Abstract:

Our project focuses on developing a system for sign language detection using machine learning, landmark detection, and the Random Forest algorithm. The aim is to create a robust solution that accurately recognizes sign language gestures in real-time, contributing to enhanced communication and accessibility for the deaf and mute community.

Individual contribution and findings:

I played a significant role in selecting the best algorithm for the project. Additionally, in my role within the project group, I spearheaded the development of the collect_images.py script, which forms a crucial component in creating the dataset for training the classifier. This involved orchestrating the webcam-based data collection process, meticulously organizing the captured images into corresponding class directories, and ensuring the dataset's readiness for subsequent training phases. Throughout this endeavor, I encountered challenges related to data consistency and image quality, which were mitigated through iterative refinement of the data collection process. My technical insights include the significance of robust dataset curation and the practical implications of real-time data acquisition in machine learning projects.

Individual contribution to project report preparation:

For the project report, my contribution includes the following sections:

1. Abstract
2. Chapter 1: Introduction
   A. Setting context for the objectives of the project
3. Chapter 2: Basic Concepts/Literature Review
   A. Sign Language Detection Methodologies
   B. Ensemble Learning and Random Forest Techniques
   C. About Mediapipe Hands Landmarking Capabilities

Individual contribution for project presentation and demonstration:

During the project presentation, I introduced the collect_images.py script that I developed for the data collection phase. I explained how it facilitates the collection of hand gesture images through webcam feeds and organizes them into class-specific directories. Additionally, I discussed the broader objectives of our project, which involves integrating machine learning, landmark detection, and random forests to create an advanced sign language detection system. By providing a comprehensive understanding of our work's scope and significance, I aimed to captivate the audience's interest and set the stage for a detailed exploration of our project's implementation and outcomes.

Full Signature of Supervisor: ……………………………

Full signature of the Student: Paritosh Kumar

SIGN LANGUAGE DETECTOR

PRIYANSHU KUMAR SINHA
21051673

Abstract:
Our project focuses on developing a system for sign language detection using machine learning, landmark detection, and the Random Forest algorithm. The aim is to create a robust solution that accurately recognizes sign language gestures in real-time, contributing to enhanced communication and accessibility for the deaf and mute community.

Individual contribution and findings:
I designed and implemented the create_dataset.py script, which iterates through the dataset directory, processes each image using the MediaPipe Hands module to detect hand landmarks, and extracts feature vectors representing the hand gestures. Additionally, I ensured that the dataset creation process was efficient and scalable to accommodate a large volume of images. Throughout the implementation, I encountered challenges related to optimizing the feature extraction process and managing memory resources efficiently. However, with careful planning and experimentation, I was able to overcome these challenges and successfully generate the dataset for training.

Individual contribution to project report preparation:
I contributed to the following sections of the project report:
- Problem Statement / Requirement Specifications
- Project Planning
- Project Analysis
- System Design
- Design Constraints
- System Architecture (ML Pipeline)

Individual contribution for project presentation and demonstration:
I assisted in preparing the project presentation and demonstrated the dataset creation process during the project demonstration. Additionally, I provided insights into the dataset generation methodology and highlighted its importance in training the sign language interpreter model effectively.

Full Signature of Supervisor: ……………………………

Full signature of the Student: Priyanshu Kumar Sinha

SIGN LANGUAGE DETECTOR

VATSAL SAXENA
21051698

Abstract:
Our project focuses on developing a system for sign language detection using machine learning, landmark detection, and the Random Forest algorithm. The aim is to create a robust solution that accurately recognizes sign language gestures in real-time, contributing to enhanced communication and accessibility for the deaf and mute community.

Individual contribution and findings:
In the project group, my role primarily focused on developing the inference classifier component of the system. This involved implementing the real-time inference of sign language gestures using landmark detection data. My contribution included:

- Designing and implementing the logic for extracting hand landmarks from video frames.
- Integrating the trained machine learning model for sign language classification.
- Fine-tuning the system for optimal performance and accuracy in real-time inference.
- My technical findings revolved around optimizing the performance of the inference classifier, ensuring smooth and accurate interpretation of sign language gestures in varying lighting and hand orientation conditions.

Individual contribution to project report preparation:
I contributed to Chapter 6 of the project report, specifically focusing on defining the future scope and conclusion of the project. This involved outlining potential advancements and applications of sign language detection technology, as well as summarizing the project's accomplishments and impact.

Individual contribution for project presentation and demonstration:
For the project presentation and demonstration, I played a significant role in preparing the visual aids and content for showcasing the functionality of the sign language interpreter system. I also actively participated in demonstrating the real-time interpretation of sign language gestures during the presentation.

Full Signature of Supervisor: ……………………………

Full signature of the Student: Vatsal Saxena

SIGN LANGUAGE DETECTION USING LANDMARKING(RANDOM FORESTS

ORIGINALITY REPORT

18%	15%	6%	14%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Submitted to KIIT University Student Paper	3%
2	Yerragudipadu Subbarayudu, Gurram Vijendar Reddy, Masuram Raj Kumar, Mudavath Aravind Naik, R.V.S.S. Nagini, Balpreet Singh. "An efficient novel approach for glaucoma classification on retinal fundus images through machine learning paradigm", MATEC Web of Conferences, 2024 Publication	1%
3	www.coursehero.com Internet Source	1%
4	www.mdpi.com Internet Source	1%
5	Submitted to University of West London Student Paper	1%
6	Submitted to National Institute of Technology, Silchar Student Paper	1%

7	www.worldleadershipacademy.live Internet Source	1 %
8	Submitted to National Institute of Technology Warangal Student Paper	1 %
9	Submitted to WHU - Otto Beisheim School of Management Student Paper	1 %
10	Submitted to University of Surrey Student Paper	<1 %
11	dspace5.zcu.cz Internet Source	<1 %
12	epdf.tips Internet Source	<1 %
13	Submitted to Carnegie Mellon University Student Paper	<1 %
14	Submitted to International Islamic University Malaysia Student Paper	<1 %
15	Submitted to University of Birmingham Student Paper	<1 %
16	webstore.iec.ch Internet Source	<1 %
17	Submitted to Liverpool John Moores University	<1 %

	Student Paper	
18	google.github.io Internet Source	<1 %
19	Submitted to Banaras Hindu University Student Paper	<1 %
20	www.lambdatest.com Internet Source	<1 %
21	www.mobilewirelessjobs.com Internet Source	<1 %
22	Submitted to Brunel University Student Paper	<1 %
23	Submitted to University of Minnesota System Student Paper	<1 %
24	scholar.ppu.edu Internet Source	<1 %
25	Submitted to University of Makati Student Paper	<1 %
26	cdn-cms.f-static.net Internet Source	<1 %
27	www.cs.cmu.edu Internet Source	<1 %
28	www.jstage.jst.go.jp Internet Source	<1 %
29	Submitted to SRM University	

Student Paper

<1%

30	fastercapital.com Internet Source	<1%
31	kirksvillekiwanis.org Internet Source	<1%
32	rdrr.io Internet Source	<1%
33	Online Information Review, Volume 38, Issue 4 (2014-09-16) Publication	<1%
34	Shahzad Ahmad Qureshi, Lal Hussain, Muhammad Rafique, Huniya Sohail et al. "EML-PSP: A novel ensemble machine learning-based physical security paradigm using cross-domain ultra-fused feature extraction with hybrid data augmentation scheme", Expert Systems with Applications, 2024 Publication	<1%
35	www.researchgate.net Internet Source	<1%

Exclude quotes	On	Exclude matches	< 10 words
Exclude bibliography	On		