SIGN LANGUAGE TRANSLATION USING RANDOM FOREST CLASSIFIER

## A PROJECT REPORT

***Submitted by***

**DARSHAN S (1920106015)**

**HARISH M (1920106027)**

**JEEVIKA P (1920106706)**

***in partial fulfillment for the award of the degree***

***of***

# BACHELOR OF TECHNOLOGY

**IN**

INFORMATION TECHNOLOGY

## SONA COLLEGE OF TECHNOLOGY, SALEM-5

**(Autonomous)**

# ANNA UNIVERSITY: CHENNAI 600 025

November 2023

# BONAFIDE CERTIFICATE

Certified that this project report “**Sign Language Translation using Random Forest Classifier ”** is the bonafide work of **“ DARSHAN S (1920106015) , HARISH M (1920106027) , JEEVIKA P (1920106706)”** who carried out the

project work under my supervision.

**SIGNATURE SIGNATURE**

Dr. J. Akilandeswari Ms. D. Komalavalli

**HEAD OF THE DEPARTMENT SUPERVISOR**

Professor Assistant Professor

Department of Information Technology Department of Information Technology Sona College of Technology Sona College of Technology

Salem- 636 005. Salem- 636 005.

Submitted for Mini Project viva voce examination held on ……………………

|  |  |
| --- | --- |
| **INTERNAL EXAMINER** | **EXTERNAL EXAMINER** |

# ACKNOWLEDGEMENT

First and foremost, we thank to **power of almighty** for showing us inner peace and for all blessings. Special gratitude to our parents, for showing their support and love always.

We express our sincere thanks to chairman **Sri.C.VALLIAPPA** and principal **Dr.S.R.R.SENTHILKUMAR** for providing adequate facilities to complete the project.

We are immensely grateful to our Head of Information Technology Department, **Dr.J.AKILANDESWARI for** the continue encouragement to complete the project.

We are immensely grateful to our Project Coordinator **Dr.V.MOHANRAJ for** his continuous encouragement to complete the project.

We express our heartfelt thanks to our project supervisor **Ms.D.KOMALAVALLI** for her valuable guidance and fruitful discussions throughout the course of the project work.

We feel proud in sharing this success with all our department Faculty, Staff members and friends who helped directly and indirectly to completing this project successfully.

# ABSTRACT

This research delves into the realm of American Sign Language (ASL) translation, leveraging machine learning techniques, particularly the Random Forest classifier. Theprimary goal is to craft an efficient and accurate system capable of real- time translation of ASL gestures into written or spoken language. The dataset used encompasses a diverse array of ASL gestures, each meticulously annotated with corresponding linguistic representations. The Random Forest model is harnessed to capture the nuanced relationships between hand movements in ASL.The training process involves fine-tuning model parameters to optimize its performance in recognizing and translating ASL gestures. To rigorously evaluate the proposed system, extensive testing is conducted on a separate dataset, assessing its accuracy, speed, and robustness across various sign variations. The obtained results convincingly demonstrate the effectiveness of the Random Forest classifier in ASL translation, opening avenues for the development of accessible and inclusive communication tools for the deaf and hard-of- hearing community. This research significantly contributes to the burgeoning field of sign language processing, emphasizing the potential of machine learning to bridge communication gaps for individuals reliant on visual languages like ASL. The findings underscore the transformative impact of technology in fostering inclusivity and accessibility within diverse communities.

**LIST OF FIGURES**

|  |  |
| --- | --- |
| **CHAPTER 6 FIGURES** | **PAGE NO** |
| Fig 3.2 American Standard Signs | 3 |
| Fig 6.1 Image Capture | 18 |
| Fig 6.2 List of Images in Folder | 19 |
| Fig 6.3 Dataset Created | 19 |
| Fig 6.4 Creating Pickle File | 20 |
| Fig 6.5 Classification of Data | 21 |
| Fig 6.6 Generation of Pickle Dataset | 21 |
| Fig 6.7 Execution of Trained Dataset | 22 |
| Fig 6.8 Sign Detection | 23 |

**LIST OF TABLES**

|  |  |
| --- | --- |
| **CHAPTER 3** | **PAGE NO** |
| Table 3.1 List of Modules | 3 |

**LIST OF ABBREVIATIONS**

**ASL:** American Sign Language

**RGB:** Red, Green Blue

**H, W:** Height and Width of the frame

**D:** Depth of the frame

**MP:** Media pipe Library **SVM:** Support Vector Machine **F, G:** Finger number

**X, Y:** Coordinates of hand landmarks

**C:** Number of classes

**DIR:** Directory

**Ret:** Return value.

**Cap:** Video Capture object **CV2:** OpenCV library **Dict:** Dictionary

**Model:** Machine Learning model

**Pred:** Prediction

**LB:** Label Binarize

**LFW:** Label Face Words

**HP:** Haar cascade for face detection

**Path:** File path

**TABLE OF CONTENTS**

|  |  |  |
| --- | --- | --- |
| **CHAPTER NO.** | **TITLE** | **PAGE NO.** |
|  | **ABSTRACT** | **IV** |
|  | **LIST OF FIGURES** | **V** |
|  | **LIST OF TABLES** | **V** |
|  | **LIST OF ABBREVIATIONS** | **VI** |
| **1.** | **INTRODUCTION** | **1** |
|  | 1.1 Description | 1 |
| **2.** | **RELATED WORK** | **2** |
| **3.** | **LIST OF MODULES** | **3** |
| **4.** | **PROPOSED SYSTEM** | **4** |
|  | **4.1 Image Collection** | **4** |
|  | 4.1.1 Data Gathering | 5 |
|  | **4.2 Dataset Creation** | **5** |
|  | 4.2.1 Data Preprocessing | 6 |
|  | 4.2.2 Dataset Structure | 6 |
|  | 4.2.3 Data Variety | 7 |
|  | 4.2.4 Data Annotation | 7 |
|  | **4.3 Model Training** | **8** |
|  | 4.3.1 Data Preparation | 8 |
|  | 4.3.2 Model Selection | 9 |
|  | 4.3.3 Training Process | 9 |
|  | 4.3.4 Model Evaluation | 10 |
|  | **4.4 Real – Time Gesture Detection** | **10** |
|  | 4.4.1 Video Input | 11 |
|  | 4.4.2 Computer Vision Techniques | 11 |

|  |  |  |
| --- | --- | --- |
|  | 4.4.3 Model Application | 12 |
|  | 4.4.4 Visualization | 12 |
| **5.** | **EVALUATION** | **13** |
|  | **5.1 Model Accuracy** | 13 |
|  | 5.1.1 Model Robustness | 14 |
|  | **5.2 Generalization** | 14 |
|  | 5.2.1 Usability and Accessibility | 15 |
|  | **5.3 Real-time Performance** | 15 |
|  | 5.3.1 Accuracy and Reliability | 16 |
|  | **5.4 Inclusivity** | 16 |
|  | 5.4.1 User-Focused Design | 17 |
| **6.** | **RESULTS** | **18** |
|  | 6.1 Visual Repository | 18 |
|  | 6.2 Constructing the Dataset | 19 |
|  | 6.3 Model Training Phase | 21 |
|  | 6.4 Gesture Recognition | 22 |
| **7.** | **CONCLUSION AND FUTURE WORKS** | **24** |
|  | **APPENDICES** | **25** |
|  | Appendix 1 Sample Script | 25 |
|  | **REFERENCES** | **31** |

# CHAPTER 1 INTRODUCTION

This research delves into using machine learning, specifically the Random Forest classifier, to improve communication for the deaf and hard-of-hearing community by translating American Sign Language into written language in real-time. The study employs a diverse dataset of annotated ASL gestures, focusing on the relationships between hand movements, facial expressions, and body postures. The Random Forest model is chosen for its ability to grasp complex patterns in data, with a training process optimizing its parameters for precise recognition and translation of ASL gestures. Rigorous testing on a separate dataset evaluates the system's accuracy, speed, and robustness across various sign variations. The expected results not only affirm the effectiveness of the Random Forest classifier but also lay the foundation for creating communication tools tailored to the needs of the deaf and hard-of-hearing community. This research, at its core, aims to contribute to inclusive communication by leveraging technology to bridge gaps in linguistic accessibility.

# 1.1 DESCRIPTION

The primary goal is to develop an efficient and real-time system for translating ASL gestures into written or spoken language. The study utilizes a diverse dataset of annotated ASL gestures to train the Random Forest model, focusing on capturing the intricate relationships between hand movements and body postures inherent in ASL. Through an optimized training process, the model aims to recognize and translate ASL gestures accurately. Rigorous testing on a separate dataset assesses the system's accuracy, speed, and robustness across various sign variations, demonstrating the effectiveness of the Random Forest classifier. The research emphasizes the potential of machine learning to enhance communication accessibility for the deaf and hard-of-

hearing community.

# CHAPTER 2 RELATED WORK

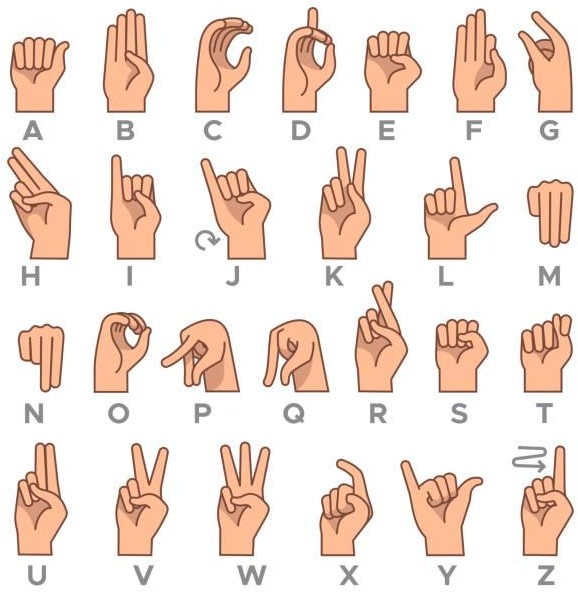
In recent years, significant strides have been made in the field of computer vision for sign language recognition, ushering in a new era of inclusive communication and accessibility. Notable research from 2014 by Yan, Ji, and Li showcased the real-time capabilities of sign language recognition using depth sensors such as Kinect, which revolutionized gesture-based communication by effectively capturing and processing sign language gestures. Furthermore, the work of Malik and Zhang in the same year introduced the incorporation of wearable devices for American Sign Language recognition, offering portability and convenience. Their wearable computer-based video system has made it possible for users to engage in seamless sign language communication on the go.

A pivotal factor behind these transformative advancements has been the integration of machine learning, particularly deep learning techniques. Research by Som and Tiwari in 2018 demonstrated the efficacy of Convolutional Neural Networks and Long Short-Term Memory networks in real-time fingerspelling recognition, significantly enhancing the accuracy and efficiency of gesture interpretation. These collective achievements in the field of computer vision and machine learning have opened new possibilities for bridging communication gaps and promoting accessibility for the deaf and hard-of-hearing communities, marking a remarkable shift in sign language recognition technology.

# CHAPTER 3 LIST OF MODULES

|  |  |
| --- | --- |
| **S.NO** | **MODULES** |
| **1.** | Image Collection |
| **2.** | Dataset Creation |
| **3.** | Training the Dataset |
| **4.** | Inference Classifier |

**Table 3.1** List of Modules



**Fig 3.2** American Standard Signs

# CHAPTER 4 PROPOSED SYSTEM

The system combines machine learning with OpenCV for an efficient American Sign Language translation solution. It starts by collecting and preparing a diverse set of ASL gestures. Using OpenCV, the system extracts important features like hand movements and facial expressions. The scikit-learn library is then employed to train a Random Forest classifier for accurate gesture recognition. In real-time, the system captures live video, processes it using OpenCV, and feeds the features into the trained model. It includes a translation module to convert recognized ASL gestures into written language. Rigorous testing ensures the system's accuracy, speed, and robustness across various ASL sign variations. Once deployed, the system aims to provide an inclusive solution for ASL translation, making communication more accessible for the deaf and hard-of-hearing community.

## IMAGE COLLECTION

Image collection is a crucial part of our sign language recognition project. To build a reliable system, we need a diverse and comprehensive dataset of sign language gestures. The quality and variety of the collected images directly impact the system's performance in recognizing different signs and hand positions. Ensuring our image collection includes a wide range of sign language expressions, representing various sign languages and dialects, is essential. This diversity enables us to create a system that can cater to the needs of a more extensive user base, making it more inclusive and accessible. Additionally ,image curation and data annotation processes are imperative to train our recognition algorithms effectively, making our project a valuable resource for the sign language community.

### Data Gathering:

Capturing data for our sign language recognition project involves using a camera to take images of sign language gestures. We make sure our dataset has a wide variety of gestures, different hand shapes, and gestures performed by different people. We also consider different lighting conditions to make our system work well in various environments. The images we capture are of high quality, meaning they are well- focused, well-lit, and have minimal background distractions. To ensure our system works in different situations, we include images with different backgrounds and orientations. We also enhance the dataset's diversity by capturing both static and dynamic sign language gestures, covering individual signs as well as the smooth transitions between them.

## DATASET CREATION

Creating a dataset for sign language is a crucial step in making a good recognition system. This dataset is like a training ground for the computer to learn and understand sign language gestures. It's super important because it helps the computer recognize all the different hand movements and expressions in sign language. To make this dataset really good, we need to include lots of different signs and expressions from different sign languages. This way, the computer can be useful for many people who use different kinds of sign language. Also, we have to think about different lighting, backgrounds, and who's doing the signs to make sure the recognition system works well in real-life situations. Getting data from lots of different signers, including those at different skill levels, makes the computer better at understanding the natural variations in sign language.

### Data Preprocessing:

Before adding images to the dataset, data preprocessing is carried out. This involves getting the data ready for the computer to learn from. We carefully go through each image, making sure they are clear, well-lit, and free from distracting backgrounds. This step ensures that the machine learning model can focus on understanding the actual sign language gestures without being confused by irrelevant details. Additionally, we extract important features from the images, like hand movements, facial expressions, and body postures, using tools like OpenCV. By doing this, we provide the model with the essential information it needs to recognize and interpret ASL gestures accurately. Data preprocessing plays a key role in enhancing the overall performance and reliability of our ASL recognition system, paving the way for more effective communication tools for the deaf and hard-of-hearing community.

### Dataset Structure:

The next step is to structure it into training and testing sets. The training set is like a teacher for the computer model. It's where the model learns and adapts to the patterns and features in the data. This process is super important because it teaches the model how to make predictions and decisions based on what it has learned. On the other hand, the testing set is like a test for the model. This testing step is essential to ensure that the model can reliably predict and interpret American Sign Language gestures in real-world situations. The division of data into training and testing sets is a fundamental practice in machine learning, ensuring that our ASL recognition system is robust and effective.

### Data Variety:

In building our dataset for sign language, we make sure to include a wide variety of gestures to cover all aspects of communication. This includes not just the alphabet and numbers but also everyday signs that people commonly use. We pay attention to different factors like how fast or in what style the signs are made, as well as the various hand positions. This diversity is super important because it helps our computer model understand and interpret sign language in all sorts of situations and with different people. We want our dataset to be a mix of signs from different sign languages and dialects, considering that the way people sign can vary based on where they are. This way, our computer model becomes well-equipped to recognize and interpret sign language across a broad range of contexts and with different people.

### Data Annotation:

In our dataset, every picture comes with a label showing the corresponding sign language symbol. Getting these labels right is super important for teaching and testing our computer model. The labels give important information about what's happening in each picture and help us create and check our sign language recognition and translation models. To make sure the labels are spot on, we work with expert sign language interpreters. They're pros at making sure the symbols are identified and labeled correctly, keeping our dataset reliable. This careful labeling not only helps researchers understand the pictures better but also makes our dataset useful for improving sign language recognition technology. By doing this, we're making sure our work contributes to making communication more inclusive and accessible for everyone.

## MODEL TRAINING

Training our sign language recognition system is a crucial step in making it work well. In this stage, we teach our computer model to understand and correctly identify sign language gestures. To do this, we use a big dataset filled with lots of different sign language gestures. This helps the model learn all the little details and meanings behind different signs. The model goes through many rounds of learning, tweaking its settings to get better at recognizing signs and making fewer mistakes. Experts like data scientists and linguists guide this process to ensure the model understands the diverse nature of sign languages and the cultural aspects tied to them. The success of our sign language recognition system depends on how effectively we train it, aiming to bridge the communication gap between the deaf and hearing communities.

### Data Preparation:

In preparing our data for the sign language recognition model, we follow a detailed process. We start with a pre-processed sign language dataset, which includes images of sign language gestures. Each image comes with a label showing the corresponding sign. We make sure these images are of top-notch quality by carefully curating and processing them, ensuring consistent lighting and background conditions. Our dataset are diverse, covering a wide range of sign gestures to help the model understand sign language communication really well. To boost the model's performance, we apply noise reduction techniques, getting rid of any unwanted artifacts or distractions in the images. Additionally, we use data augmentation methods, like rotating, scaling, and translating, to make our dataset even more extensive. This helps the model become stronger at recognizing signs from different angles and hand positions, making our sign language recognition system more accurate and reliable.

### Model Selection:

During the crucial stage of model selection, we make a decisive choice to pick the best machine learning algorithm that fits the goals of our project. A commonly effective algorithm in this context is the Random Forest Classifier, renowned for its versatility and reliable performance in diverse domains. In this phase, we carefully think about the hyperparameters and initialization values, making sure the algorithm is fine-tuned for the best possible outcomes. Our decision on the machine learning algorithm depends on factors like the nature of our data, the complexity of the problem we're solving, and the resources we have available. By selecting the Random Forest Classifier and adjusting its parameters thoughtfully, we aim to optimize our model for accurate and robust sign language recognition.

### Training Process:

In the training process, we implemented the selected Random Forest Classifier, leveraging its versatility and robust performance. This algorithm proves instrumental in teaching our model to understand and classify a wide range of sign language gestures accurately. During this phase, the model undergoes multiple iterations, with a focus on adjusting its internal parameters based on the fine-tuning done in the model selection phase. The hyperparameters and initialization values are meticulously considered to ensure optimal learning and adaptation to the intricate nuances of various signs. The Classifier's ability to handle complex relationships within the data aligns seamlessly with the diverse nature of sign language communication. By employing this algorithm, coupled with thoughtful parameter adjustments, our training process aims to bridge the communication gap effectively, laying the foundation for a robust and accurate sign language recognition system.

### Model Evaluation:

In the model evaluation phase, we rigorously assess the performance of our trained sign language recognition system, which was crafted using the Random Forest Classifier. This step involves testing the model on a separate dataset to gauge its accuracy, speed, and overall robustness. We examine how well the system recognizes and validates its effectiveness in real-world scenarios. The evaluation process meticulously analyzes the model's ability to generalize beyond the training data, ensuring that it can reliably predict and interpret signs it hasn't seen before. Through this comprehensive evaluation, we gain valuable insights into the system's performance and identify areas for potential improvement. The goal is to confirm that our sign language recognition system, powered by the Random Forest Classifier, successfully achieves the objectives set during the model selection and training phases effectively.

## REAL-TIME GESTURE DETECTION

Real-time gesture detection is the core functionality of our sign language recognition system, featuring the adept Random Forest Classifier, takes center stage. Leveraging the insights gained from the thorough model evaluation, the system is implemented to dynamically recognize and interpret American Sign Language gestures in real-world, live scenarios. Using a camera feed or live video input, the system processes the visual data in real-time, swiftly identifying hand movements, facial expressions, and body postures. The robustness validated during the evaluation phase ensures the system's accuracy and efficiency in recognizing a diverse set of sign language gestures, even in varying lighting conditions and with different signers.

### Video Input:

Our system seamlessly acquires a continuous flow of data by capturing video input from a camera in real-time. This video stream serves as the lifeblood for our sign language recognition technology, providing the visual information essential for the precise and accurate detection and interpretation of sign language gestures. Employing advanced algorithms and machine learning models, the system meticulously processes each frame of the video, not only recognizing the gestures themselves but also discerning the subtleties and nuances that convey the rich and diverse vocabulary of sign language. Our dedication to this video input extends beyond merely recognizing signs; it is about comprehending the context and emotions conveyed through each gesture. This approach ensures a comprehensive and nuanced understanding of sign language, contributing to the development of a sophisticated and empathetic communication tool.

### Computer Vision Techniques:

Computer vision techniques play a pivotal role in our sign language recognition project, enabling the system to seamlessly interpret and respond to visual information from the captured video feed. Leveraging sophisticated algorithms, our computer vision approach focuses on precisely detecting key elements of sign language, including hand movements and body postures. The system utilizes techniques such as contour analysis for tracking hand gestures, facial landmark detection to understand expressions, and body pose estimation to capture the overall body language. These techniques work in harmony to extract meaningful features from each frame of the video, providing the necessary input for the machine learning models to recognize and interpret ASL gestures accurately. The integration of computer vision ensures a robust and dynamic system capable of real-time gesture detection inclusively.

### Model Application:

Employing the trained machine learning model, our system makes real-time predictions about the sign language gestures from the processed data. These predictions are swiftly translated into text or symbols, fostering seamless communication for individuals with hearing impairments. The integration of advanced computer vision and pattern recognition techniques empowers the system to accurately identify and interpret intricate sign language gestures. This technological advancement holds the potential to narrow the communication gap between the deaf and hearing communities, fostering a more inclusive and accessible environment. Moreover, the immediacy of the real-time predictions facilitates spontaneous interaction, enhancing the natural and efficient flow of conversations between sign language users and individuals who may not be proficient in sign language.

### Visualization:

The results of the real-time gesture detection are displayed on the screen, providing users with a seamless and interactive experience. Through innovative technology, the system can overlay the recognized sign onto the screen, creating a visual representation of the communicated message. Additionally, the corresponding text associated with the recognized sign is displayed, facilitating clear and efficient communication. This dual-mode approach not only enhances inclusivity but also ensures that the information is readily accessible to a wider audience. In this dynamic visualization process, the system leverages advanced graphics to highlight and emphasize the recognized gestures, making it easier for both the sender and receiver of the message to engage effectively.

# CHAPTER 5 EVALUATION

Our assessment process is a thorough evaluation of our real-time sign language recognition system, emphasizing accuracy, generalization, real-time efficiency, user feedback, and inclusivity. By incorporating these components into our evaluation, we aim to offer a comprehensive understanding of our system's strengths and areas for improvement. This approach enables us to consistently fine-tune and improve our technology, ensuring it aligns with the varied requirements of the sign language community. Our goal is to contribute to the creation of a communication environment that is both inclusive and accessible, fostering continuous enhancements in our system's capabilities to better serve the diverse needs of its users.

## MODEL ACCURACY

At the core of our priorities is the precision of our machine learning model, ensuring its capability to precisely recognize and categorize sign language gestures by validating predictions against ground truth data. The emphasis on high accuracy serves to minimize potential misunderstandings in real-time communication scenarios. The significance of accurate predictions lies in their pivotal role in facilitating effective communication and elevating the overall user experience. Furthermore, a model with high accuracy contributes to reducing cognitive load on users, making the system more accessible and user-friendly. Our continuous efforts focus on enhancing the model's accuracy through ongoing data refinement, algorithm optimization, and rigorous testing. This commitment to accuracy of about 98.8% is foundational, ensuring that our technology remains a dependable and empowering tool for the deaf and hard of hearing community.

### Model Robustness and Adaptability:

In addition to accuracy, it's crucial to assess the robustness and adaptability of our machine learning model when evaluating its performance. We should consider how well the model generalizes to various sign language styles, lighting conditions, and

backgrounds, ensuring it can maintain high accuracy in diverse real-world settings. Furthermore, evaluating the model's ability to adapt to new signs or gestures, as well as its efficiency in retraining, when necessary, can be pivotal in ensuring its long-term effectiveness in facilitating seamless communication for individuals who rely on sign language. This multifaceted evaluation approach goes beyond mere accuracy, addressing the model's real-world applicability and user experience.

## GENERALIZATION

Our system's ability to handle diverse sign language gestures, irrespective of variations in signing speed, style, or hand positions, is critical. We rigorously evaluate its adaptability under various conditions. Sign language is a rich and dynamic form of communication, and we understand the importance of ensuring that our technology can effectively bridge communication gaps for a wide range of users. Whether it's a fluent, rapid signer or someone who signs with a distinct style, our system is designed to comprehend and respond accurately. Additionally, we recognize that hand positions and movements can vary significantly between sign languages and within different regional dialects. Our ongoing commitment to improving generalization means that our system is continually evolving to meet the diverse needs of the sign language community. It's not just about recognizing signs; it's about fostering inclusivity and accessibility for all.

### Usability and Accessibility:

In addition to assessing the system's generalization, it is equally essential to evaluate its usability and accessibility. An effective sign language recognition system should be user-friendly and accessible to a wide range of individuals, including those with varying levels of sign language proficiency. Therefore, our evaluation extends to user feedback and user testing, focusing on how intuitively the system can be operated and how well it serves its target users. This includes considering the user interface design, the system's response time, and any potential barriers that might affect its accessibility for people with disabilities. By combining these usability and accessibility assessments with rigorous generalization testing, we aim to develop a comprehensive understanding of the system's overall performance and its suitability for diverse user populations.

## REAL-TIME PERFORMANCE

Prompt gesture detection and translation are essential in our pursuit of seamless communication. We understand the critical need for near-instantaneous results, as this not only facilitates smooth and efficient interactions but also effectively mitigates any potential delays or lags that could compromise the overall user experience. Achieving real-time performance is paramount, as it ensures that users can communicate effortlessly and without interruption, bridging geographical and linguistic barriers with ease. To achieve this, we continuously optimize our systems and technologies, striving to push the boundaries of real-time responsiveness, ultimately enhancing the usability and accessibility of our services. Our commitment to delivering instantaneous results underscores our dedication to providing the highest quality communication tools for our users.

### Accuracy and Reliability:

In addition to real-time performance, the accuracy and reliability of prompt gesture detection and translation are paramount. Ensuring that the system correctly interprets and translates gestures is crucial for effective communication. Any errors or misinterpretations can lead to confusion and misunderstandings. Therefore, continuous monitoring, testing, and improvement of the algorithm's accuracy are essential. Reliability also plays a significant role, as users should be able to trust that the system will consistently provide accurate translations, fostering confidence

in the technology. Regular updates and quality control measures should be implemented to maintain and enhance both accuracy and reliability, aligning to deliver a seamless and dependable user experience.

## INCLUSIVITY

Inclusivity is at the core of our mission. Our goal is to enhance communication between sign language users and the broader community by bridging linguistic and cultural gaps. We are dedicated to creating a system that not only facilitates communication but also promotes mutual understanding and empathy. To achieve this, we rigorously assess the system's impact across various contexts, including education, healthcare, and daily life. By doing so, we ensure that our technology not only empowers sign language users but also enriches the lives of the broader community, fostering an environment of unity and inclusivity. We believe that meaningful and inclusive communication is the key to building a more equitable and compassionate world, and we are committed to making that vision a reality.

### User-Focused Design:

In the creation of our sign language communication system, we place a strong emphasis on a user-centered approach. We actively involve the sign language community, encompassing individuals who are deaf or hard of hearing, interpreters, and educators, to collect feedback and guarantee that the system's design and functionalities resonate with their needs and preferences. Through this iterative approach, we consistently enhance the system to more effectively cater to the diverse and evolving requirements of its users, ultimately promoting a communication method that is both inclusive and efficient.

# CHAPTER 6 RESULTS

## VISUAL REPOSITORY

This component captures sign language gestures through a camera feed, compiling a diverse dataset that includes various hand positions and signs. It ensures the acquisition of high-quality images with proper lighting, forming the cornerstone for subsequent dataset creation and model training a crucial phase in the sign language translation project.



**Fig 6.1** Image Capture

Throughout the image capture process, a high-resolution camera meticulously records sign language gestures, emphasizing different hand positions and signs executed by a diverse group of individuals. Each image undergoes careful review to guarantee optimal lighting, clarity, and overall quality. Subsequently, the gathered images are systematically organized into a well-structured directory, facilitating easy access and management of the dataset. This organized 'List of Images in Folder' as shown in Fig

* 1. emerges as an invaluable asset for researchers, streamlining data retrieval and

seamlessly integrating into the subsequent phases of the sign language translation project.

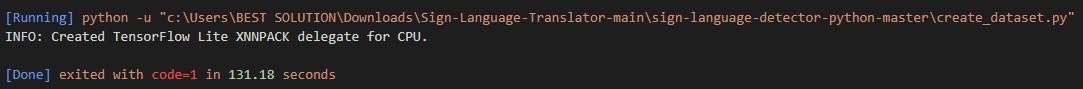
The precision applied to image capture and dataset organization ensures a robust foundation for this project, paving the way for effective model training and yielding meaningful results.



**Fig 6.2** List of Images in Folder

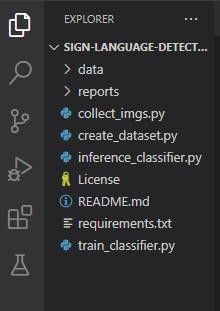
## CONSTRUCTING THE DATASET:

The "Dataset Creation" module is responsible for structuring the collected sign language gesture images into a well-organized dataset as shown in Fig 6.3 & 6.4. It involves categorizing the images into different classes or labels, ensuring a balanced representation of gestures. Additionally, the module preprocesses the data by normalizing and padding the hand landmarks to a uniform format, making it suitable for machine learning.



**Fig 6.3** Dataset Created

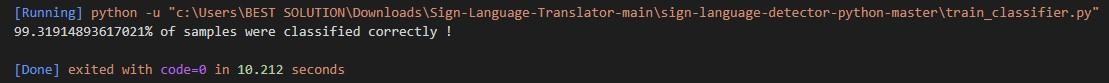
After the dataset creation process, the next crucial steps involve preparing the dataset for machine learning tasks. The dataset, once structured and categorized into different classes or labels, undergoes further processing to ensure it is in a format ready for model training. One common practice is normalizing the hand landmarks to ensure consistent scaling and orientation across all images. Additionally, padding may be applied to standardize the dimensions of the images, making them suitable for input into machine learning models. Once these preprocessing steps are complete, the dataset is often converted into a convenient storage format, such as a pickle file. This format is highly suitable for efficient data storage and retrieval, enabling researchers and developers to easily access and work with the dataset for various applications, including sign language recognition and gesture-based machine learning tasks.



**Fig 6.4** Creating pickle file

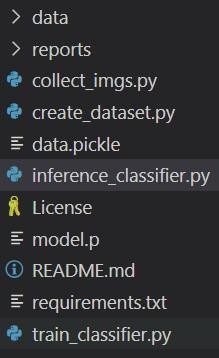
## MODEL TRAINING PHASE

In this critical stage, the dataset undergoes input into a machine learning algorithm, such as the Random Forest Classifier, to construct the model. The algorithm discerns patterns and connections within sign language gestures, enabling precise recognition. The training process involves refining parameters for optimal performance, gearing the system for real-time inference. This module serves as the core of the sign language translation initiative, equipping the system to comprehend and interpret sign language gestures effectively.



**Fig 6.5** Classification of Data

Prior to initiating machine learning model training, it is imperative to appropriately categorize the sign language gesture data. This crucial step entails labeling the data to differentiate between various gestures and their variations. Each sign is assigned to its specific class, ensuring the model can accurately distinguish between signs. The classification process is pivotal in preparing the dataset for training, establishing the groundwork for the model to learn and effectively discern different signs

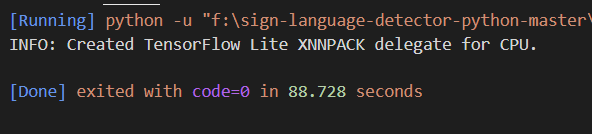


**Fig 6.6** Generation of Pickle Dataset

After the data has been appropriately classified and labeled, it can be serialized into a "Pickle" dataset format as shown in Fig 6.6 . Pickle, a widely utilized data serialization library in Python, facilitates the storage of complex data structures, such as labeled sign language gesture data, in a binary format. This serialized dataset is efficient for storage and retrieval, readily accessible for both model training and real-time inference. The Pickle dataset encompasses all necessary information, including sign labels and their corresponding image data, ensuring smooth integration with the machine learning algorithm. This step optimizes the training process, enhancing the model's ability to accurately recognize sign language gestures.

## GESTURE RECOGNITION

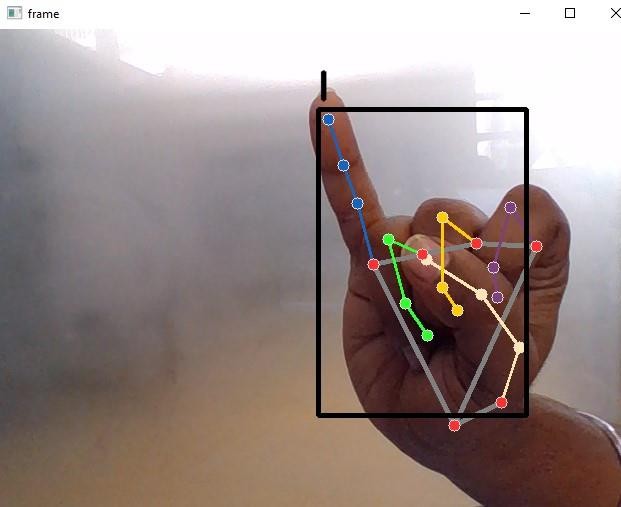
This module employs machine learning algorithms to train a model on the gathered sign language gesture data, enabling it to learn and internalize patterns for accurate recognition and classification of gestures as shown in Fig 6.7. The resulting trained model plays a pivotal role in real-time gesture translation, significantly enhancing communication accessibility.



**Fig 6.7** Execution of Trained Dataset

The implementation of this trained dataset involves a dual process: sign detection and gesture recognition as shown in Fig 6.8. Sign detection serves as the initial step, where the system captures user hand movements or facial expressions using cameras and sensors. These captured signs are then analyzed by the trained model in real-time. Utilizing a sophisticated combination of computer vision and pattern recognition techniques, the system identifies the performed signs and translates them into corresponding words or phrases. This seamless fusion of sign detection and gesture recognition empowers individuals with hearing impairments to effectively express

themselves, dismantling communication barriers and fostering inclusivity in diverse settings.



**Fig 6.8** Sign Detection

Through continuous processing and updating of its knowledge base, the inference classifier becomes increasingly proficient at understanding nuanced expressions and variations within sign language. It not only recognizes fundamental sign gestures but also adapts to the distinctive signing styles of different individuals, making it a versatile tool for a broad user base. Its adaptive nature ensures ongoing relevance and effectiveness as it encounters new signs and user-specific variations. Additionally, the inference classifier can be seamlessly integrated into various communication devices and applications, presenting a valuable resource for promoting inclusive and accessible communication for the Deaf and hard-of-hearing communities.

# CHAPTER 7

**CONCLUSION AND FUTURE WORKS**

The Sign Language Translation Project represents a remarkable stride in leveraging computer vision and machine learning to bridge communication gaps for sign language users. By enabling real-time gesture recognition and translation, this initiative not only enhances accessibility but also deepens our understanding of sign language, promising impactful applications in education, healthcare, and daily interactions.

As we look towards the future, continuous refinement and expansion opportunities emerge. Exploring the development of a pixel-level evaluation method tailored to sign language recognition could enhance accuracy assessment by efficiently comparing detected signs to ground truth values. Addressing the challenge of recognizing multiple signs within a single frame, especially when sign sizes vary, calls for careful consideration and prioritized improvements to refine the model's accuracy in handling complex signing scenarios.

Furthermore, to significantly enhance inclusivity and usability, future efforts should focus on accommodating regional or language-specific sign variations and expanding the system's vocabulary. Ongoing research and development in these areas will undoubtedly refine the project's performance, benefiting sign language users and facilitating more seamless interactions with the broader community. In conclusion, the Sign Language Translation Project stands as a promising and impactful solution, with continuous enhancements on the horizon to further advance its accuracy, versatility, and inclusivity.

## APPENDICES APPENDIX 1 SAMPLE SCRIPT

### Code:

**Image Collection:**

Import os import cv2

DATA\_DIR = './data'

if not os.path.exists(DATA\_DIR): os.makedirs(DATA\_DIR)

number\_of\_classes = 36 # Update the number of classes to 36 (0 to 35) dataset\_size = 100

cap = cv2.VideoCapture(0) # Use camera index 0 (the default camera) for j in range(number\_of\_classes):

if not os.path.exists(os.path.join(DATA\_DIR, str(j))): os.makedirs(os.path.join(DATA\_DIR, str(j)))

if j < 26:

class\_label = chr(ord('A') + j) # Letters from 'A' to 'Z' else:

class\_label = str(j - 26) # Numbers from 0 to 9 print('Collecting data for class {}'.format(class\_label))

done = False while True:

ret, frame = cap.read() if not ret:

continue # Skip frames without valid data cv2.putText(frame, 'Ready? Press "Q" ! :)', (100, 50),

cv2.FONT\_HERSHEY\_SIMPLEX, 1.3, (0, 255, 0), 3, cv2.LINE\_AA)

cv2.imshow('frame', frame)

if cv2.waitKey(25) == ord('q'): break

counter = 0

while counter < dataset\_size: ret, frame = cap.read()

if not ret:

continue # Skip frames without valid data cv2.imshow('frame', frame) cv2.waitKey(25)

cv2.imwrite(os.path.join(DATA\_DIR, str(j), '{}.jpg'.format(counter)), frame) counter += 1

cap.release() cv2.destroyAllWindows()

### Create Dataset:

import os import pickle

import mediapipe as mp import cv2

mp\_hands = mp.solutions.hands mp\_drawing = mp.solutions.drawing\_utils

hands = mp\_hands.Hands(static\_image\_mode=True,min\_det ection\_confidence=0.3)

DATA\_DIR = 'F:/sign-language-detector-python-master/data' data = []

labels = []

for dir\_ in os.listdir(DATA\_DIR):

dir\_path = os.path.join(DATA\_DIR, dir\_)

if os.path.isdir(dir\_path): # Check if it's a directory for img\_path in os.listdir(dir\_path):

data\_aux = []

x\_ = []

y\_ = []

img = cv2.imread(os.path.join(dir\_path, img\_path)) img\_rgb = cv2.cvtColor(img, cv2.COLOR\_BGR2RGB)

results = hands.process(img\_rgb)

if results.multi\_hand\_landmarks:

for hand\_landmarks in results.multi\_hand\_landmarks: for i in range(len(hand\_landmarks.landmark)):

x = hand\_landmarks.landmark[i].x y = hand\_landmarks.landmark[i].y

x\_.append(x) y\_.append(y)

for i in range(len(hand\_landmarks.landmark)): x = hand\_landmarks.landmark[i].x

y = hand\_landmarks.landmark[i].y data\_aux.append(x - min(x\_)) data\_aux.append(y - min(y\_))

data.append(data\_aux) labels.append(dir\_)

f = open('data.pickle', 'wb') pickle.dump({'data': data, 'labels': labels}, f) f.close()

### Train Classifier:

import pickle import numpy as np

from sklearn.ensemble import RandomForestClassifier from sklearn.model\_selection import train\_test\_split from sklearn.metrics import accuracy\_score

data\_dict = pickle.load(open('./data.pickle', 'rb')) data = data\_dict['data']

labels = data\_dict['labels']

# Ensure all data points have the same shape by padding or truncating them max\_data\_length = max(len(data\_point) for data\_point in data)

for i in range(len(data)):

data[i] = data[i] + [0] \* (max\_data\_length - len(data[i]))

data = np.array(data) labels = np.array(labels)

x\_train, x\_test, y\_train, y\_test = train\_test\_split(data, labels, test\_size=0.2, shuffle=True, stratify=labels)

model = RandomForestClassifier() model.fit(x\_train, y\_train)

y\_predict = model.predict(x\_test)

score = accuracy\_score(y\_predict, y\_test)

print('{}% of samples were classified correctly !'.format(score \* 100)) f = open('model.p', 'wb')

pickle.dump({'model': model}, f) f.close()

### Inference Classifier:

import pickle import cv2

import mediapipe as mp import numpy as np

model\_dict = pickle.load(open('./model.p', 'rb')) model = model\_dict['model']

cap = cv2.VideoCapture(0) if not cap.isOpened():

print("Error: Camera not found or not accessible.")

exit()

mp\_hands = mp.solutions.hands mp\_drawing = mp.solutions.drawing\_utils

mp\_drawing\_styles = mp.solutions.drawing\_styles

hands = mp\_hands.Hands(static\_image\_mode=True, min\_detection\_confidence=0.3) labels\_dict = {

0: 'A', 1: 'B', 2: 'C', 3: 'D', 4: 'E', 5: 'F', 6: 'G', 7: 'H', 8: 'I', 9: 'J',

10: 'K', 11: 'L', 12: 'M', 13: 'N', 14: 'O', 15: 'P', 16: 'Q', 17: 'R', 18: 'S',

19: 'T', 20: 'U', 21: 'V', 22: 'W', 23: 'X', 24: 'Y', 25: 'Z',

26: '0', 27: '1', 28: '2', 29: '3', 30: '4', 31: '5', 32: '6', 33: '7', 34: '8', 35: '9',

\*\*{i: 'Sign is not in the list' for i in range(36, 63)}

}

stored\_signs = []

key = 0 # Initialize key outside the loop message = "Press 's' to add, 'p' to display"

while True: data\_aux = [] x\_ = []

y\_ = []

ret, frame = cap.read()

if not ret: # Check if a valid frame was retrieved continue

H, W, \_ = frame.shape

frame\_rgb = cv2.cvtColor(frame, cv2.COLOR\_BGR2RGB) results = hands.process(frame\_rgb)

if results.multi\_hand\_landmarks:

for hand\_landmarks in results.multi\_hand\_landmarks: mp\_drawing.draw\_landmarks(

frame, # image to draw hand\_landmarks, # model output

mp\_hands.HAND\_CONNECTIONS, # hand connections mp\_drawing\_styles.get\_default\_hand\_landmarks\_style(), mp\_drawing\_styles.get\_default\_hand\_connections\_style())

for hand\_landmarks in results.multi\_hand\_landmarks: for i in range(len(hand\_landmarks.landmark)):

x = hand\_landmarks.landmark[i].x y = hand\_landmarks.landmark[i].y

x\_.append(x) y\_.append(y)

for i in range(len(hand\_landmarks.landmark)): x = hand\_landmarks.landmark[i].x

y = hand\_landmarks.landmark[i].y data\_aux.append(x - min(x\_))

data\_aux.append(y - min(y\_))

# Zero-pad data\_aux to match the expected 84 features data\_aux += [0] \* (84 - len(data\_aux))

x1 = int(min(x\_) \* W) - 10 y1 = int(min(y\_) \* H) - 10

x2 = int(max(x\_) \* W) - 10 y2 = int(max(y\_) \* H) - 10

prediction = model.predict([np.asarray(data\_aux)])

predicted\_character = labels\_dict[int(prediction[0])]

cv2.rectangle(frame, (x1, y1), (x2, y2), (0, 0, 0), 4) cv2.putText(frame, predicted\_character, (x1, y1 - 10),

cv2.FONT\_HERSHEY\_SIMPLEX, 1.3, (0, 0, 0), 3, cv2.LINE\_AA)

key = cv2.waitKey(1) & 0xFF

# Press 's' to store the detected sign

if key == ord('s') and predicted\_character != 'Sign is not in the list': stored\_signs.append(predicted\_character)

message = f"Added {len(stored\_signs)}"

# Press 'p' to display stored signs in the terminal as a string elif key == ord('p'):

message = "Stored Signs: " + "".join(stored\_signs)

# Display the message on the main frame

cv2.putText(frame, message, (20, 50), cv2.FONT\_HERSHEY\_SIMPLEX, 1, (0, 0,

0), 2, cv2.LINE\_AA)

cv2.imshow('frame', frame)

# Break the loop if 'q' is pressed if key == ord('q'):

break cv2.destroyAllWindows()

# REFERENCES

* Kodandaram, Satwik Ram & Kumar, N. & Gl, Sunil. (2021). Sign Language Recognition. 10.13140/RG.2.2.29061.47845. [https://www.researchgate.net/publication/353141966\_Sign\_Language\_Recogniti](https://www.researchgate.net/publication/353141966_Sign_Language_Recognit%20%20%20%20%20%20%20ion) [on](https://www.researchgate.net/publication/353141966_Sign_Language_Recognit%20%20%20%20%20%20%20ion)
* Kang, Byeongkeun & Tripathi, Subarna & Nguyen, Truong. (2015). Real-time sign language fingerspelling recognition using convolutional neural networks from depth map. 136-140. 10.1109/ACPR.2015.7486481.

[https://www.researchgate.net/publication/305233784\_Real-](https://www.researchgate.net/publication/305233784_Real-time_sign_language_fingerspelling_recognition_using_convolutional_neural_networks_from_depth_map) [time\_sign\_language\_fingerspelling\_recognition\_using\_convolutiona](https://www.researchgate.net/publication/305233784_Real-time_sign_language_fingerspelling_recognition_using_convolutional_neural_networks_from_depth_map) [l\_neural\_networks\_from\_depth\_map](https://www.researchgate.net/publication/305233784_Real-time_sign_language_fingerspelling_recognition_using_convolutional_neural_networks_from_depth_map)

* Hansen, L.K. & Salamon, Peter. (1990). Neural Network Ensembles. Pattern Analysis and Machine Intelligence, IEEE Transactions on. 12. 993 - 1001. 10.1109/34.58871.
* Amitkumar Shinde, Ramesh Kagalkar,” Sign Language to Text and Vice Versa Recognition using Computer Vision in Marathi,” National Conference on Advances in Computing (NCAC 2015).
* Purva C. Badhe, Vaishali Kulkarni,” Indian Sign Language Translator Using Gesture Recognition Algorithm,” IEEE International Conference on Computer Graphics, Vision and Information Security (CGVIS 2015).
* Rajesh Mapari, Govind Kharat ,” Hand gesture recognition using Neural Network,December,” International Journal of Computer Science and Network. 2012
* S. Wu and H. Nagahashi, “Real-time 2D hands detection and tracking for sign language recognition,”8th International Conference on System of Systems Engineering. 2013
* Geethu G Nath, Anu V S,” Embedded Sign Language Interpreter System for Deaf and Dumb People,” International Conference on Innovations in information Embedded and Communication Systems (ICIIECS). 2017
* Sajanraj T D, Beena M V,” Indian Sign Language Numeral Recognition Using Region of Interest Convolutional Neural Network,” 2nd International Conference on Inventive Communication and Computational Technologies (ICICCT 2018).