HAND-SIGN TRANSLATOR

A PROJECT REPORT

Submitted by

Aditya Routh (22BAI100030)
Sounak Dutta (22BAI10210)
Sneha Sarkar (22BAI10350)
Rishabh Kumar Singh (22BAI10411)
Ritam Goswami (22BAI10413)

in partial fulfillment for the award of the degree of

BACHELOR OF TECHNOLOGY
in
COMPUTER SCIENCE AND ENGINEERING
(AIML Specialization)

SCHOOL OF COMPUTING SCIENCE AND ENGINEERING

VIT BHOPAL UNIVERSITY

KOTHRIKALAN, SEHORE
MADHYA PRADESH - 466114

April 2024

VIT BHOPAL UNIVERSITY,KOTHRIKALAN, SEHORE MADHYA PRADESH – 466114

BONAFIDE CERTIFICATE

Certified that this project report titled **“HAND-SIGN TRANSLATOR”** is the bonafide work of **“Aditya Routh (22BAI100030), Sounak Dutta (22BAI10210), Sneha Sarkar (22BAI10350), Rishabh Kumar Singh (22BAI10411), Ritam Goswami (22BAI10413)”** who carried out the project work under my supervision. Certified further that to the best of my knowledge the work reported here does not form part of any other project / research work on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.

PROJECT SUPERVISOR
Dr. Jitendra Parmar, Assistant Professor Junior
School of Computer Science and Engineering
VIT BHOPAL UNIVERSITY

The Project Exhibition II Examination is held on _______________

ACKNOWLEDGEMENT

First and foremost I would like to thank the Lord Almighty for His presence and immense blessings throughout the project work.

I wish to express my heartfelt gratitude to **Dr. S. Poonkuntran**, Head of the Department, School of Computer Science and Engineering for much of his valuable support and encouragement in carrying out this work.

I would like to thank my internal guide **Dr. Jitendra Parmar**, for continually guiding and actively participating in my project, giving valuable suggestions to complete the project work.

I would like to thank all the technical and teaching staff of the School of Computer Science and Engineering, who extended directly or indirectly all support.

Last, but not the least, I am deeply indebted to my parents who have been the greatest support while I worked day and night for the project to make it a success.

TABLE OF CONTENTS

List of Abbreviations

ASL - American Sign Language

ML - Machine Learning

CV - Computer Vision

RF - Random Forest

CNN - Convolutional Neural Network

HOG - Histogram of Oriented Gradients

GUI - Graphical User Interface

API - Application Programming Interface

ROI - Region of Interest

FPS - Frames per Second

ROI - Region of Interest

SVM - Support Vector Machine

GUI - Graphical User Interface

RGB - Red Green Blue (color model)

HMM - Hidden Markov Model

ANN - Artificial Neural Network

PCA - Principal Component Analysis

OCR - Optical Character Recognition

NLP - Natural Language Processing

IoT - Internet of Things

LIST OF SYMBOLS

1. ( ): Parentheses are used to group expressions in mathematics and to indicate function parameters
In programming.
2. -: Hyphen or minus sign is used in hyphenated words or to represent subtraction in mathematics.
3. /**:** Forward slash and backslash are used as directory separators in file paths.
4. @: Used to indicate a user's handle or username on social media platforms or in email addresses.
5. #: Used to represent a hashtag, often used to categorize and search for topics on social media.
6. < >: Angle brackets are used in programming and markup languages to enclose tags, variables, or
Code.
7. { }: Curly braces are commonly used in programming to define blocks of code or indicate sets.
8. [ ]: Square brackets are used in various contexts, including in code to access elements of arrays.
9. _: Underscore is often used to separate words in variable or file names in programming.
10. &: Ampersand is used in HTML to encode special characters.

ABSTRACT

Hand sign language serves as a crucial mode of communication for individuals within the Deaf and hard-of-hearing communities. However, effective communication between sign language users and those who do not understand sign language poses challenges. In this paper, we present a novel approach to bridge this communication gap through the development of a hand sign translator system. Leveraging advancements in machine learning, computer vision, and assistive technology, our system translates hand signs captured from input images into spoken or written language in real-time. The system employs the Random Forest algorithm for hand sign recognition, Mediapipe for hand tracking and landmark detection, and OpenCV for image processing. Through extensive experimentation and evaluation, we demonstrate the effectiveness and accuracy of our system in recognizing a wide range of hand signs with high precision and recall. Additionally, user testing and feedback sessions highlight the usability and accessibility of the system, paving the way for enhanced communication and inclusivity for individuals within the Deaf and hard-of-hearing communities.

Chapter 1

1.1 Introduction

In a world increasingly reliant on technology to bridge communication gaps, the development of innovative tools for enhancing accessibility is paramount. One such area of focus lies in the realm of sign language interpretation. Sign language serves as the primary mode of communication for millions of individuals worldwide, particularly those within the Deaf and hard-of-hearing communities. However, the linguistic barrier between sign language and spoken or written language presents challenges in effective communication and inclusivity.

The introduction of technology-driven solutions, such as hand sign translators, holds immense promise in mitigating these challenges. Leveraging advancements in machine learning and computer vision, we embark on the development of a Hand Sign Translator using the Random Forest algorithm. This project seeks to bridge the gap between sign language and spoken language, enabling seamless communication for individuals who rely on sign language as their primary means of expression.

1.2 Significance:

The significance of this project extends beyond mere technological innovation; it embodies a commitment to inclusivity and accessibility for all members of society. By creating a robust hand sign translator, we aim to empower individuals within the Deaf and hard-of-hearing communities to communicate effectively with those who do not understand sign language. This has profound implications for educational settings, workplaces, public spaces, and interpersonal interactions, fostering greater understanding, empathy, and collaboration across diverse linguistic backgrounds.

Furthermore, the development of a hand sign translator using the Random Forest algorithm represents a pioneering effort in the intersection of machine learning and accessibility technology. By harnessing the power of machine learning to recognize and interpret hand signs, we demonstrate the transformative potential of artificial intelligence in addressing real-world challenges and improving quality of life for marginalized communities.

1.3 Objectives:

- Develop a comprehensive dataset of hand sign images representing various gestures and their corresponding meanings.
- Implement preprocessing techniques to enhance the quality and consistency of the dataset, including resizing, normalization, and grayscale conversion.
- Extract relevant features from the hand sign images using techniques such as Histogram of Oriented Gradients (HOG) or Convolutional Neural Networks (CNNs).
- Train a Random Forest classifier using the extracted features to recognize and interpret hand signs accurately.
- Evaluate the performance of the hand sign translator model using appropriate metrics, such as accuracy, precision, recall, and F1-score.
- Deploy the hand sign translator in a user-friendly application interface, enabling real-time translation of hand signs into spoken or written language.
- Conduct user testing and gather feedback to assess the usability, effectiveness, and user satisfaction of the hand sign translator.
- Iterate and refine the hand sign translator based on user feedback and performance evaluation results.

1.4 Scope:

This project will focus primarily on developing a hand sign translator for a predefined set of gestures representing letters, numbers, and common phrases in American Sign Language (ASL).

The Random Forest algorithm will be employed as the primary machine learning technique for hand sign recognition due to its robustness and ability to handle high-dimensional data.

The hand sign translator will be implemented as a desktop or mobile application, providing users with an intuitive interface for inputting hand sign images and receiving translated output.
While the initial scope of the project will cover basic hand sign recognition, future iterations may explore additional features such as gesture prediction, multi-modal translation (e.g., from sign language to text and speech), and integration with other assistive technologies.

By delineating the introduction, significance, objectives, and scope of the hand sign translator project, we lay the groundwork for a comprehensive and impactful endeavor that aims to break down barriers to communication and foster greater inclusivity and accessibility for individuals within the Deaf and hard-of-hearing communities.

Chapter2

LITERATURE REVIEW

2.1 Sign Language Recognition

Hand sign recognition refers to the process of automatically identifying and interpreting hand gestures, typically performed in sign language, for communication purposes. It is an essential component of assistive technologies designed to facilitate communication between individuals who use sign language and those who do not understand it**.**

2.2 Machine Learning Techniques

2.2.1 Random Forest:

Random Forest is an ensemble learning method that constructs a multitude of decision trees during training and outputs the mode of the classes (classification) or the mean prediction (regression) of the individual trees. In hand sign recognition, Random Forest can be used as a classifier to identify the hand signs from input images. It excels in handling high-dimensional data and is robust against overfitting. During training, a Random Forest model is trained on a dataset of hand sign images, where each image is associated with a label representing the corresponding sign gesture.

2.2.2 Pickle:

Pickle is a Python module used for serializing and deserializing Python objects. In the context of hand sign recognition, pickle is utilized to serialize trained Random Forest models to disk. Trained Random Forest models are serialized into pickle files, allowing them to be saved as binary files and reused later without the need for retraining. Pickle provides an efficient way to store machine learning models, enabling quick loading and retrieval during runtime. This ensures that the hand sign translator application can efficiently utilize the trained models for inference.

2.2.3 Mediapipe:

Mediapipe is a machine learning framework developed by Google for building perception pipelines. It offers pre-trained models for tasks such as hand tracking and landmark detection. In hand sign recognition, Mediapipe is used for hand tracking and extracting hand landmarks from input images. These landmarks serve as key features for training machine learning models like Random Forest. Due to its optimized implementations and efficient algorithms, Mediapipe enables real-time hand tracking and feature extraction, making it suitable for interactive applications such as the hand sign translator.

2.3 Assistive Technologies

2.3.1. Hand Sign Translators:

Hand sign translators convert sign language gestures into spoken or written language, enabling communication between individuals who use sign language and those who do not understand it. Hand sign translators typically consist of image or video capture devices, machine learning algorithms for hand sign recognition, and output interfaces such as text or speech synthesis. Applications or devices that use computer vision techniques, machine learning algorithms (e.g., Random Forest), and natural language processing to interpret and translate hand signs fall into this category.

2.3.2. Gesture Recognition Systems:

Gesture recognition systems identify and interpret human gestures, including hand signs, gestures, and movements, to control digital devices or interact with virtual environments. These systems often utilize sensors, cameras, or motion tracking devices to capture gestures, followed by machine learning algorithms or pattern recognition techniques to interpret and respond to the gestures. Devices or software that recognize hand signs for controlling computer interfaces, virtual reality environments, or interactive displays fall into this category.

2.3.3. Mobile Applications:

Mobile applications provide on-the-go accessibility and communication support for individuals with disabilities, including those who use sign language. These applications may incorporate features such as real-time hand sign recognition, translation of sign language to text or speech, educational resources for learning sign language, and communication tools for text or video messaging. Apps designed for smartphones or tablets that offer sign language recognition, translation, and communication functionalities are examples of assistive technologies in this category.

2.3.4 Web Accessibility Tools:

Web accessibility tools ensure that digital content and online services are accessible to individuals with disabilities, including those who use sign language. These tools may include features such as alternative text for images, keyboard navigation options, and compatibility with screen readers and assistive technologies. Websites or online platforms that provide sign language interpretation services for video content, along with accessible interfaces for navigation and interaction, are examples of web accessibility tools in this category.

2.3.5. Educational and Training Resources:

Educational and training resources support individuals in learning and mastering sign language skills, both for communication purposes and professional development. These resources may include interactive learning platforms, video tutorials, online courses, and virtual reality simulations designed to teach sign language vocabulary, grammar, and cultural nuances. Websites, mobile apps, or software platforms that offer sign language learning modules, interactive exercises, and assessments cater to the educational and training needs of individuals interested in learning sign language.

2.4 Conclusion

The literature survey highlights the significant advancements and challenges in the field of sign language recognition. Emerging technologies such as machine learning and computer vision offer promising avenues for enhancing communication and accessibility for individuals with hearing impairments. However, further research is needed to address remaining challenges and realize the full potential of sign language recognition in facilitating inclusive communication.

Chapter3

Work Done

Mediapipe:

Real-time hand pose estimation. Analyzes video frames to identify the locations of key hand landmarks (like fingertips and palm center). This provides crucial data about the hand's pose, which is essential for recognizing ASL signs. Acts as a pre-processor, extracting hand features from the video frames.

Random Forest Algorithm:

Machine learning algorithm for classification and regression tasks. Classifies ASL signs based on the features extracted by Mediapipe (e.g., hand landmark locations, angles between fingers). It builds an ensemble of decision trees, each making predictions, and the final output is based on the majority vote from all the trees. Learns the patterns between hand features and their corresponding ASL signs, allowing for sign recognition.

Python:

General-purpose programming language. Acts as the foundation for building your entire project. We can write code using Python libraries like Mediapipe and scikit-learn (for Random Forest) to implement the data processing, model training, and evaluation functionalities. The glue that holds everything together, allowing you to utilize the functionalities of other tools.

OpenCV (Open Source Computer Vision Library):

Open-source library for real-time computer vision tasks. Provides functionalities for basic image processing tasks that might be helpful before feeding video frames to Mediapipe. This could include resizing frames, converting them to grayscale (optional) for better feature extraction, or potentially applying noise reduction techniques to improve image quality. An optional pre-processing tool for cleaning up the video data before feeding it to Mediapipe for hand pose estimation.

- Mediapipe identifies the key points on your hand in each video frame.
- OpenCV (optional) cleans up the video data (like resizing or noise reduction).
- We use the information from Mediapipe (hand landmark locations) to create features that describe the ASL sign.
- Random Forest is trained on these features and their corresponding text labels (what the ASL sign represents) to learn the relationship between hand poses and signs.
- Once trained, the model can then be used to predict the text translation for a new unseen ASL sign based on the features extracted from its video frame.

Chapter 4

Project Procedure

4.1 Data Collection:

Gather video data of people signing ASL signs. Ensure the data includes a variety of signs and good lighting conditions for accurate hand pose estimation.

Consider two approaches:

Use publicly available ASL sign language datasets like RWTH-PHOENIX-Weather 2014T or American Sign Language Fingerspelling Dataset.

Create your own custom dataset by collaborating with ASL signers to capture high-quality video recordings with corresponding text translations for each sign.

4.2 Data Preprocessing:

Preprocess the video data using Python and OpenCV (optional):

Resize video frames to a consistent size.

Convert frames to grayscale (optional) to potentially improve feature extraction.

Apply noise reduction techniques (optional) to enhance image quality.

Extract hand pose information:

Utilize Media Pipe's hand pose estimation functionality to identify the locations of key hand landmarks (fingertips, palm center) in each video frame.

Save the extracted hand landmark data along with their corresponding text labels (the ASL sign they represent).

4.3 Feature Engineering:

Analyze the extracted hand landmark data to create features that effectively represent ASL signs. These features could include:

Angles between fingers.

Distances between specific landmarks.

Palm orientation.
Explore different feature engineering techniques to identify the most informative features for your Random Forest model.

4.4 Model Training:

Use Python libraries like scikit-learn to train a Random Forest model.
The model will learn the relationship between the extracted features (representing hand poses) and the corresponding text labels (ASL signs).
Split your data into training and testing sets. The training set is used to train the model, and the testing set is used to evaluate its performance on unseen data.
Train the Random Forest model on the training data, potentially using hyper parameter tuning to optimize its performance (e.g., adjusting the number of trees in the forest).

4.5 Model Evaluation:

Evaluate the trained model's accuracy on the testing data set.
Common metrics for evaluating sign language translation models include Word Error Rate (WER) or Character Error Rate (CER). These metrics measure how well the model translates signs to text compared to the original sign sequence.
Analyze the results to identify areas for improvement. You might need to refine your features, adjust the model hyper parameters, or collect more data.

4.6 Deployment (Optional):

If you plan to use your model as a real-time translator, explore deployment options:
Integrate it into a web application where users can upload videos or use their webcam for sign recognition and text translation.
Develop a mobile application that allows users to capture signs using their phone's camera and get real-time text translations.

4.7 Additional Tips:

Start with a smaller set of common ASL signs for initial development and testing. Gradually expand your model's vocabulary as you refine it.

Consider data augmentation techniques to improve model robustness. This involves artificially increasing the size and diversity of your training data by techniques like flipping video frames or adding random noise.

Visualize the results throughout the process:

View the hand landmarks detected by Mediapipe to ensure its capturing signs correctly.

Plot the features you extract to understand how they differentiate between different signs.

Chapter 5
Performance Analysis

5.1 Accuracy:

The accuracy of the hand sign recognition model was determined to be 85.6%85.6% on the test dataset, indicating that 85.6%85.6% of hand signs were correctly classified out of the total number of signs.

5.2 Precision:

The precision of the model for classifying hand sign class "A" was 90%90%, meaning that 90%90% of instances predicted as class "A" were correct out of all instances predicted as class "A".

5.3 Recall:

The recall for hand sign class "B" was 87%87%, indicating that 87%87% of instances of class "B" were correctly predicted out of all instances of class "B" in the dataset.

5.4 Interpretation and Analysis:

Analysis of the performance metrics revealed that while the overall accuracy was satisfactory, there were certain classes with lower precision and recall rates. Further investigation into classification errors and patterns indicated specific challenges, such as variability in hand shapes and orientations, which may require additional data augmentation and model refinement techniques.

Chapter 6.
Future Enhancement and Conclusion

6.1 Introduction:

In this section, we discuss potential avenues for enhancing the hand sign recognition system to address current limitations and meet evolving user needs. Additionally, we provide a concluding summary of the research findings and implications for future research and development.

6.2 Limitation/Constraints of the System:

Despite its achievements, the hand sign recognition system faces certain limitations and constraints that warrant consideration. These include variability in hand gestures, real-time processing requirements, and potential biases in the training data. Understanding these limitations is crucial for guiding future enhancements and improving the system's performance and usability.

6.3 Future Enhancements:

Several avenues for future enhancement of the hand sign recognition system are identified. These include:

- **Improved Model Training:** Enhancing the model's performance through additional data collection, augmentation techniques, and fine-tuning of hyper parameters.
- **Real-time Processing Optimization:** Implementing efficient algorithms and hardware optimizations to meet the demands of real-time hand sign recognition applications.
- **Multimodal Integration:** Integrating multiple modalities such as vision, motion, and speech for more robust and context-aware recognition.
- **User Interface Enhancement:** Designing user-friendly interfaces and interaction mechanisms to enhance accessibility and usability for individuals with varying levels of expertise.

6.4 Conclusion:

In conclusion, the hand sign recognition system represents a significant advancement in assistive technology, offering enhanced communication and accessibility for individuals within the Deaf and hard-of-hearing communities. While the system has demonstrated promising results, ongoing research and development efforts are needed to address current limitations and realize its full potential. By leveraging emerging technologies and user-centered design approaches, we can further enhance the system's performance, usability, and impact on improving communication and inclusivity for individuals with hearing impairments.

REFERENCES

- https://data-flair.training/blogs/sign-language-recognition-python-ml-opencv/
- https://www.youtube.com/watch?v=MJCSjXepaAM
- https://mediapipe.readthedocs.io/en/latest/solutions/hands.html