SIGN SAVVY RECOGNITION USING MACHINE LEARNING

Minor project-II report submitted
in partial fulfillment of the requirement for award of the degree of

Bachelor of Technology
in
Computer Science & Engineering

By

PAARA VARSHITH	(21UECS0437)	**(VTU20508)**
PARUPALLI SAMPATH	(21UECS0714)	**(VTU20359)**
HEMA SUNDHAR VEGI	(21UECS0718)	**(VTU20416)**

Under the guidance of
Mrs.S.KIRUTHIGA,
ASSISTANT PROFESSOR

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
SCHOOL OF COMPUTING

VEL TECH RANGARAJAN DR. SAGUNTHALA R&D INSTITUTE OF SCIENCE & TECHNOLOGY

(Deemed to be University Estd u/s 3 of UGC Act, 1956)
Accredited by NAAC with A++ Grade
CHENNAI 600 062, TAMILNADU, INDIA

May, 2024

SIGN SAVVY RECOGNITION USING MACHINE LEARNING

Minor project-II report submitted
in partial fulfillment of the requirement for award of the degree of

Bachelor of Technology
in
Computer Science & Engineering

By

PAARA VARSHITH	(21UECS0437)	**(VTU20508)**
PARUPALLI SAMPATH	(21UECS0714)	**(VTU20359)**
HEMA SUNDHAR VEGI	(21UECS0718)	**(VTU20416)**

Under the guidance of
Mrs.S.KIRUTHIGA,
ASSISTANT PROFESSOR

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
SCHOOL OF COMPUTING

VEL TECH RANGARAJAN DR. SAGUNTHALA R&D INSTITUTE OF SCIENCE & TECHNOLOGY

(Deemed to be University Estd u/s 3 of UGC Act, 1956)
Accredited by NAAC with A++ Grade
CHENNAI 600 062, TAMILNADU, INDIA

May, 2024

CERTIFICATE

It is certified that the work contained in the project report titled "SIGN SAVVY RECOGNITION USING MACHINE LEARNING" by "PAARA VARSHITH (21UECS0437), PARUPALLI SAMPATH (21UECS0714), HEMA SUNDHAR VEGI (21UECS0718) " has been carried out under my supervision and that this work has not been submitted elsewhere for a degree.

Signature of Supervisor	**Signature of Professor In-charge**
Mrs.S.Kiruthiga	**Dr.Peter Soosai Anandaraj A**
Computer Science & Engineering	**Computer Science & Engineering**
School of Computing	**School of Computing**
Vel Tech Rangarajan Dr. Sagunthala R&D	**Vel Tech Rangarajan Dr. Sagunthala R&D**
Institute of Science & Technology	**Institute of Science & Technology**
May, 2024	**May, 2024**

DECLARATION

We declare that this written submission representation of ideas in our own words and where others' ideas or words have been included, we have adequately cited and referenced the original sources. We also declare that we have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in our submission. We understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

PAARA VARSHITH

Date: / /

PARUPALLI SAMPATH

Date: / /

HEMA SUNDHAR VEGI

Date: / /

APPROVAL SHEET

This project report entitled SIGN SAVVY RECOGNITION USING MACHINE LEARNING by PAARA VARSHITH (21UECS0437), PARUPALLI SAMPATH (21UECS0714),HEMA SUNDHAR VEGI (21UECS0718) is approved for the degree of B.Tech in Computer Science & Engineering.

Examiners **Supervisor**

Mrs.S.Kiruthiga., Assistant Professsor.,

Date: / /

Place:

ACKNOWLEDGEMENT

We express our deepest gratitude to our respected **Founder Chancellor and President Col. Prof. Dr. R. RANGARAJAN B.E. (EEE), B.E. (MECH), M.S (AUTO),D.Sc., Foundress President Dr. R. SAGUNTHALA RANGARAJAN M.B.B.S.** Chairperson Managing Trustee and Vice President.

We are very much grateful to our beloved **Vice Chancellor Prof. S. SALIVAHANAN,** for providing us with an environment to complete our project successfully.

We record indebtedness to our **Professor & Dean, Department of Computer Science & Engineering, School of Computing, Dr. V. SRINIVASA RAO, M.Tech., Ph.D.,** for immense care and encouragement towards us throughout the course of this project.

We are thankful to our **Head, Department of Computer Science & Engineering, Dr.M.S. MURALI DHAR, M.E., Ph.D.,** for providing immense support in all our endeavors.

We also take this opportunity to express a deep sense of gratitude to our **Internal Supervisor Mrs.S.KIRUTHIGA ASSISTANT PROFESSOR,** for her cordial support,valuable information and guidance, she helped us in completing this project through various stages.

A special thanks to our **Project Coordinators Mr.V. ASHOK KUMAR, M.Tech., Ms. U.HEMAVATHI, M.E., Ms. C. SHYAMALA KUMARI, M.E.,** for their valuable guidance and support throughout the course of the project.

We thank our department faculty, supporting staff and friends for their help and guidance to complete this project.

PAARA VARSHITH	**(21UECS0437)**
PARUPALLI SAMPATH	**(21UECS0714)**
HEMA SUNDHAR VEGI	**(21UECS0718)**

ABSTRACT

The project title ”Sign Savvy Recognition Using Machine Learning” is quite self-explanatory, as it describes the essence of the project.The ”Sign Language Recognition System Using Machine Learning” project holds immense societal, technological, and educational significance, addressing the needs of a marginalized community while pushing the boundaries of AI and fostering inclusivity and accessibility.An Sign Language is one of the way to communicate with deaf people. In this work sets, included features and variation in the language with locality have been the major barriers which has led to little research being done in ISL. One should learn sign language to interact with them. Learning usually takes place in peer groups. There are very few study materials available for sign learning. Because of this, the process of learning sign language learning is a very difficult task. The initial stage of isign learning is Finger spelled signi learning and moreover, are used when no corresponding sign exists or signer is not aware of it. Most of the existing tools for sign language learning use external sensors which are costly. Our project aims at extending a step forward in this field by collecting a dataset and then use various feature extraction techniques to extract useful information which is then input into various supervised learning techniques. Currently, we have reported four fold cross validated results for the different approaches, and the difference from the previous work done can be attributed to the fact that in our four fold i cross validation, the validation set Correspond to images of a iperson different from the persons in the training set.

Keywords:

Classification, Computer vision, Convolutional neural networks (CNN), Data augmentation, Feature extraction, Gesture recognition, Hand tracking, Machine learning, Real-time detection. Sign Savvy recognition,

LIST OF FIGURES

LIST OF ACRONYMS AND ABBREVIATIONS

CV	Computer Vision
CNN	Convolutional Neural Network
DA	Data Augmentation
GR	Gesture Recognition
LSF	Language of Sign Features
ML	Machine Learning
RTD	Real-Time Detection
SSR	Sign Savvy Recognition

TABLE OF CONTENTS

Chapter 1

INTRODUCTION

1.1 Introduction

Sign Savvy Recognition Systems aim to automatically interpret and translate sign language gestures into spoken language or text, facilitating communication between individuals who use sign language and those who do not. By leveraging advances in computer vision, pattern recognition, and deep learning, SLRS have the potential to bridge the communication gap and promote inclusivity for the deaf and hard of hearing population.

Sign language is a vital mode of communication for individuals who are deaf or hard of hearing, enabling them to convey thoughts, ideas, and emotions through hand gestures, facial expressions, and body movements. However, the inability of many people to understand sign language poses significant barriers to communication and inclusion for the deaf community. To address this challenge, researchers and technologists have been exploring the development of Sign Language Recognition Systems (SLRS) using Machine Learning (ML) techniques.Indian Sign Language (ISL) is used in a community of deaf people throughout India. But ISL is not used in deaf schools to teach deaf children. Teacher training programs do not guide teachers in teaching methods that use ISL. There are no tutorials that include sign language. Parents of deaf children are unaware of sign language and its ability to remove communication barriers. ISL interpreters are an urgent need in institutions and places where communication between the deaf and the normal occurs but less than 300 translators are in India. Therefore, an institution that meets all of these requirements was a necessity. After a long struggle for the deaf community, the Department approved the establishment of the ISLRTC in New Delhi on 28 September 2015. Unlike spoken languages, where grammar is expressed using punctuation-based symbols, feature, attitude and syntax sign languages use gesture, punctuation, and body and facial expressions to form grammar.
Many sign languages have developed independently around the world, and no sign

language can be identified. Both signed systems and handwritten characters have been found worldwide. Until the 19th century, much of what we know about historical sign languages is limited to the characters of the alphabet that were developed to facilitate the transfer of words from spoken language into sign language, rather than from the language itself. Talking to people with a hearing impairment is a major challenge. Deaf and mute people use sign language to communicate, the ordinary people face the problem of recognizing their sign language. There is therefore a need for programs that recognize different signals and transmit information to ordinary people.

1.2 Aim Of The Project

The aim of the project ”Sign Savvy Recognition System Using Machine Learning” is to develop a robust and accurate system capable of interpreting sign language gestures from videos or images.Sign language recognition systems bridge the communication gap between the deaf community and the hearing population, enabling seamless interaction and understanding.Sign language recognition empowers the deaf community by providing them with tools to express themselves and engage more effectively in a predominantly spoken language society.

1.3 Project Domain

This project focuses on developing a system that can recognize and interpret sign language gestures using machine learning techniques. The system will employ computer vision algorithms to capture and analyze hand movements, enabling real-time translation of sign language into text or spoken language. The goal is to bridge communication barriers between individuals who are deaf or hard of hearing and those who do not understand sign language.

1.4 Scope Of The Project

The objective and scope of the project ”Sign Savvy Recognition Using Machine Learning” are crucial for defining its goals and boundaries.Sign language involves intricate hand movements, facial expressions, and body language, making it a challenging task for machines to accurately interpret and recognize.

Chapter 2

LITERATURE REVIEW

[1] S. Singh et al.,"Sign Savvy Recognition using Machine Learning"(2020) Their review paper provides an overview of machine learning techniques used for sign language recognition. It covers approaches such as Hidden Markov Models (HMMs), Support Vector Machines (SVMs), and Decision Trees for gesture recognition. The paper discusses challenges, datasets, and evaluation methods in sign language recognition research.

[2] Li et al.,"Deep Learning-Based Hand Gesture Recognition for Sign Language Applications" (2019) Their study proposes a deep learning-based approach for hand gesture recognition in sign language applications.The authors use a combination of convolutional neural networks (CNNs) and recurrent neural networks (RNNs) to capture spatial and temporal information from hand gesture sequences.Experimental results demonstrate the effectiveness of the proposed approach in accurately recognizing sign language gestures.

[3] Nguyen et al., "Real-Time Sign Language Detection Using Convolutional Neural Networks" (2020),Their paper presents a real-time sign language detection system based on convolutional neural networks (CNNs). The authors develop a dataset of sign language gestures captured from multiple viewpoints and lighting conditions. The CNN model is trained to recognize hand gestures in real-time, achieving high accuracy and low latency.

[4] Zhang et al., "Sign Language Recognition Using 3D Convolutional Neural Networks" (2021) . Their research explores the use of 3D convolutional neural networks (3D CNNs) for sign language recognition. The authors propose a novel architecture that processes spatiotemporal information from sign language videos to accurately recognize gestures. Experimental results demonstrate superior performance compared to traditional 2D CNNs, especially in capturing temporal dynamics.

[5] Wang et al.,"Towards Real-Time Sign Savvy Recognition Using Recurrent Neural Networks" (2019). Their study investigates the use of recurrent neural networks (RNNs) for real-time sign language recognition. The authors propose a hybrid RNN

architecture that combines long short-term memory (LSTM) cells with gated recurrent units (GRUs) to capture both short-term and long-term dependencies in sign language sequences. Experimental results demonstrate the feasibility of real-time sign language recognition using the proposed approach.

[6] Al-Hamadi et al.,"Real-Time Sign Language Recognition using Convolutional Neural Networks" (2020). Their paper presents a real-time sign language recognition system based on Convolutional Neural Networks (CNNs). It describes the architecture of the CNN model and the integration of hand tracking for gesture localization. The system achieved high accuracy and real-time performance, making it suitable for practical applications.

[7] S. A. Khan et al.,"A Survey on Deep Learning Techniques for Hand Gesture Recognition in Sign Language"(2016) Their survey paper provides a comprehensive overview of deep learning techniques for hand gesture recognition in sign language. It covers various CNN architectures, RNNs, and attention mechanisms used for gesture recognition. The paper discusses challenges, datasets, and future directions in deep learning-based sign language recognition research.

[8] Park et al.,"Enhanced Hand Gesture Recognition for American Sign Language Using Deep Learning" (2022) Their study explores enhanced deep learning techniques for hand gesture recognition in American Sign Language (ASL). The authors propose a novel deep learning architecture that combines CNNs and attention mechanisms to capture spatial and temporal dependencies in sign language gestures. Experimental results show that the proposed approach outperforms existing methods in terms of accuracy and robustness in ASL recognition tasks.

[9] Pandey et al.,"Sign Language Recognition Using Convolutional Neural Networks" (2020) Their research investigates the use of CNNs for sign language recognition, focusing on the recognition of isolated signs from video sequences. The authors propose a CNN architecture tailored for sign language recognition and evaluate its performance on public datasets. Experimental results demonstrate the effectiveness of the CNN-based approach in achieving high accuracy and robustness in sign language recognition tasks.

Chapter 3

PROJECT DESCRIPTION

3.1 Existing System

The Sign Language Transformer is a sign language recognition system developed by researchers at Google AI and the University of Oxford. It employs a transformer-based architecture, similar to those used in natural language processing tasks.The system operates on video sequences of sign language gestures and encodes them into a sequence of feature vectors using a transformer encoder. Then, a transformer decoder generates the corresponding text representation of the sign language.

3.2 Proposed System

The purpose of the system is to improve the existing system in this area in terms of response time and accuracy with the use of efficient algorithms, high quality data sets and better sensors.Sign Language detection system shows what the position of hands in viewfinder of camera module means with good accuracy. It can then be used to help people who are just beginning to learn Sign Language or those who don't know sign language but have a close one who is deaf.

3.3 Feasibility Study

3.3.1 Economic Feasibility

The economic feasibility of a sign language recognition system using machine learning involves considering various factors related to costs, benefits, and potential returns on investment. This assessment helps you understand the costs, benefits, risks, and potential returns associated with the project, enabling you to make strategic choices and allocate resources effectively.Take into account any regulatory requirements, legal considerations, or compliance standards that may affect the economic feasibility of the project. Ensure that the system complies with relevant accessibility guidelines, data privacy regulations, and industry standards.

3.3.2 Technical Feasibility

The technical feasibility of a sign savvy recognition using machine learning involves evaluating whether the project is technically viable given the available resources, expertise, and technological constraints.sign language recognition system using machine learning is technically viable and identify any potential barriers or challenges that need to be addressed during project planning and execution.

Data Availability and Quality:Determine if there is a sufficient amount of high-quality data available for training the machine learning models. This includes datasets of sign language gestures with accurate annotations.

Machine Learning Expertise:Assess the availability of expertise in machine learning, deep learning, and computer vision within the project team or organization. Consider whether team members have the necessary skills and experience to develop and deploy machine learning models for sign language recognition.

Model Selection and Development:Evaluate different machine learning models and algorithms for sign language recognition, considering factors such as performance, scalability, and suitability for real-time inference. Determine the appropriate model architecture (e.g., CNNs, RNNs, transformers) based on the characteristics of the input data and the complexity of the recognition task.

Software and Tools:Identify and assess the availability of software libraries, frameworks, and development tools for building sign language recognition systems using machine learning.

Risk Assessment:Identify potential technical risks and challenges that could affect the feasibility of the project, such as data scarcity, algorithmic complexity, hardware limitations, or software dependencies.

3.3.3 Social Feasibility

The social feasibility of a sign language recognition system using machine learning involves evaluating its potential impact on individuals, communities, and society as a whole.the social feasibility of a sign language recognition system and ensure that it aligns with principles of inclusivity, diversity, and social justice. This approach helps foster a more inclusive and equitable society where everyone has the opportunity to participate and communicate effectively, regardless of their abilities or communication preferences.

3.4 System Specification

3.4.1 Hardware Specification

The hardware specifications for a sign language recognition system using machine learning depend on various factors such as the complexity of the models, the size of the dataset, real-time requirements, and deployment environment.the hardware specifications to the specific requirements and constraints of your sign language recognition system, considering factors such as budget, performance goals, and scalability requirements. Additionally, as machine learning technology continues to evolve, staying updated with the latest hardware advancements can help improve the efficiency and performance of your system.

3.4.2 Software Specification

The software specifications for a sign language recognition system using machine learning encompass the tools, libraries, and frameworks necessary for development, training, and deployment.
Programming Languages:
Python: Python is widely used in the machine learning community due to its extensive libraries and frameworks for data manipulation, visualization, and model training. Most machine learning frameworks are compatible with Python.
C/C++:Some performance-critical components or libraries may be implemented in C/C++ for efficiency, especially for real-time inference on embedded systems or specialized hardware.
Machine Learning Frameworks:
TensorFlow: TensorFlow is an open-source deep learning framework developed by Google. It provides a comprehensive ecosystem for building and training machine learning models, including support for neural networks, convolutional networks, and recurrent networks.
PyTorch: PyTorch is another popular deep learning framework known for its dynamic computation graph and intuitive API. It is widely used for research and

development in machine learning and artificial intelligence.
Keras: Keras is a high-level neural networks API that runs on top of TensorFlow, Theano, or Microsoft Cognitive Toolkit (CNTK). It offers a user-friendly interface for building and training deep learning models.
scikit-learn: scikit-learn is a Python library for machine learning built on top of NumPy, SciPy, and matplotlib. It provides simple and efficient tools for data preprocessing, model selection, and evaluation.
These tools and libraries offer a comprehensive ecosystem for building robust and scalable machine learning applications while providing flexibility and ease of development.

3.4.3 Standards and Policies

WCAG (Web Content Accessibility Guidelines): These guidelines provide recommendations for making web content more accessible to people with disabilities, including those who use sign language. Adhering to WCAG ensures that your sign language recognition system is accessible to a diverse range of users.
ISO 9241 (Ergonomics of Human-System Interaction): ISO 9241 provides standards for designing interactive systems that are user-friendly, including considerations for accessibility and usability.
ACM Code of Ethics and Professional Conduct: The ACM (Association for Computing Machinery) provides a code of ethics and professional conduct that outlines principles and guidelines for ethical behavior in computing professions, including considerations for fairness, accountability, and transparency in the development and use of technology.

Standard Used: ISO/IEC 27001

Chapter 4

METHODOLOGY

4.1 General Architecture Of The Sign Savvy Language

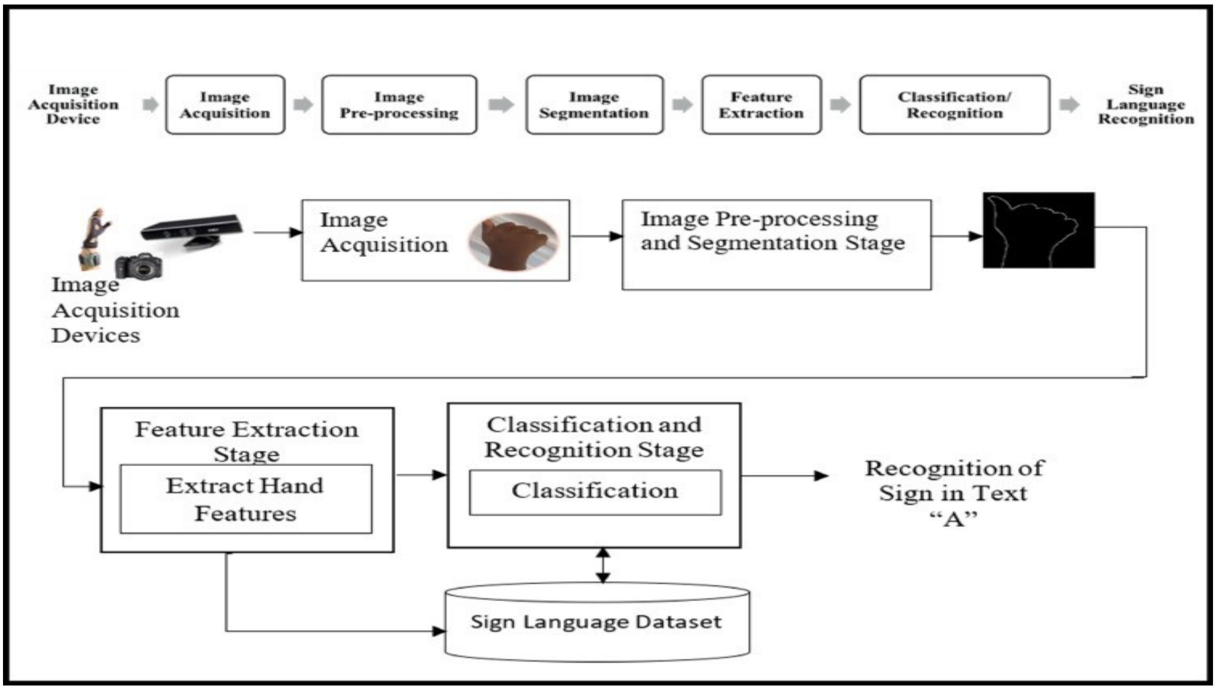

Figure 4.1: **General Architecture Of The Sign Savvy Language**

In Figure 4.1 represents the General Architecture is represented by provides a high-level overview of the components and their interactions in a Sign Language Recognition System using Machine Learning. Depending on specific requirements and constraints, the actual implementation may vary.

4.2 Design Phase

4.2.1 Data Flow Diagram For Classificaton And Informartion

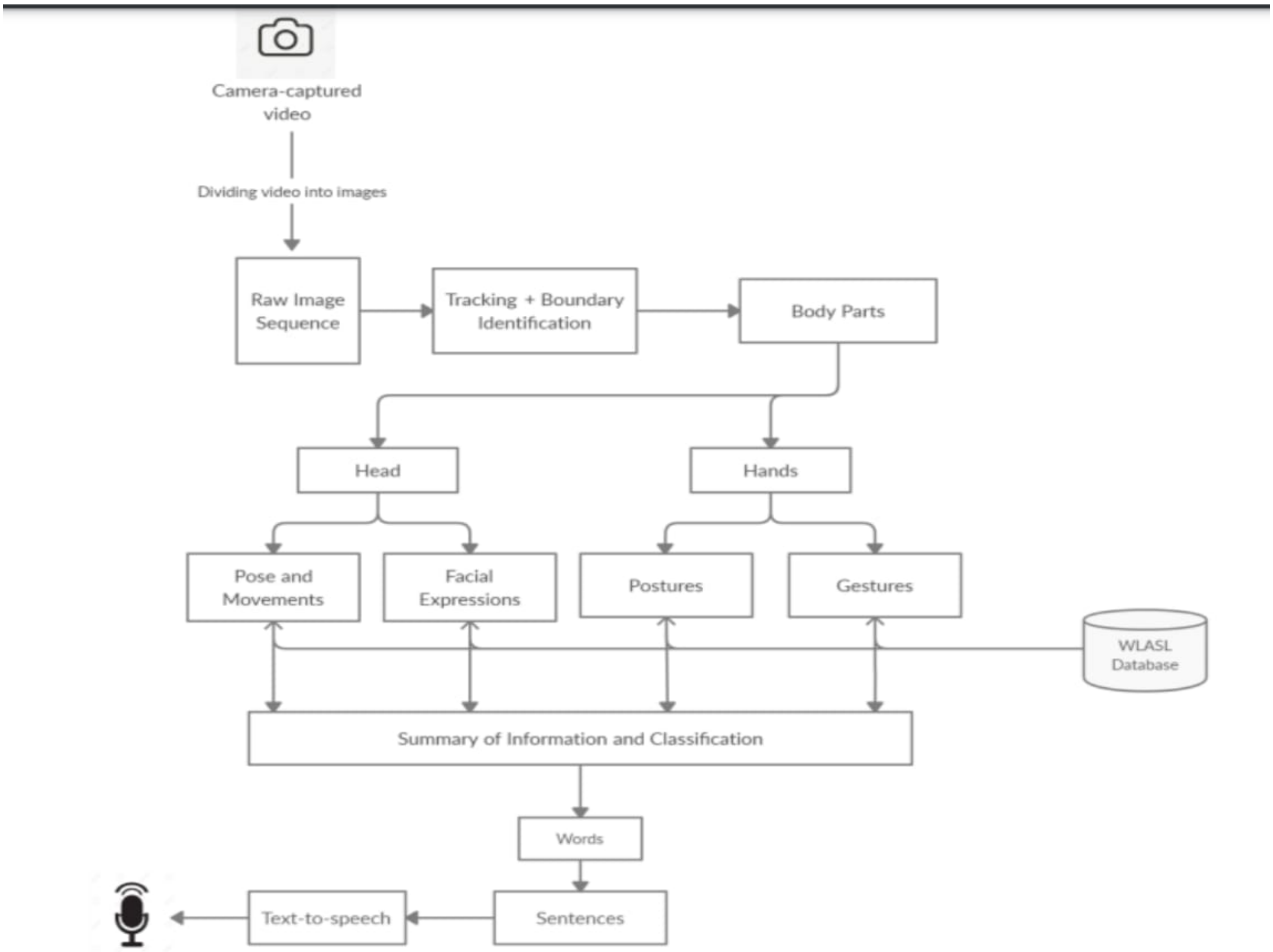

Figure 4.2: **Data Flow Diagram For Classificaton And Informartion**

In Figure 4.2 represents the Data Flow Diagram (DFD) is represented by Sign Savvy Recognition using Machine Learning would illustrate the flow of data within the system, showing how input is processed, recognized, and displayed.

4.2.2 Use Case Diagram For Sign Language

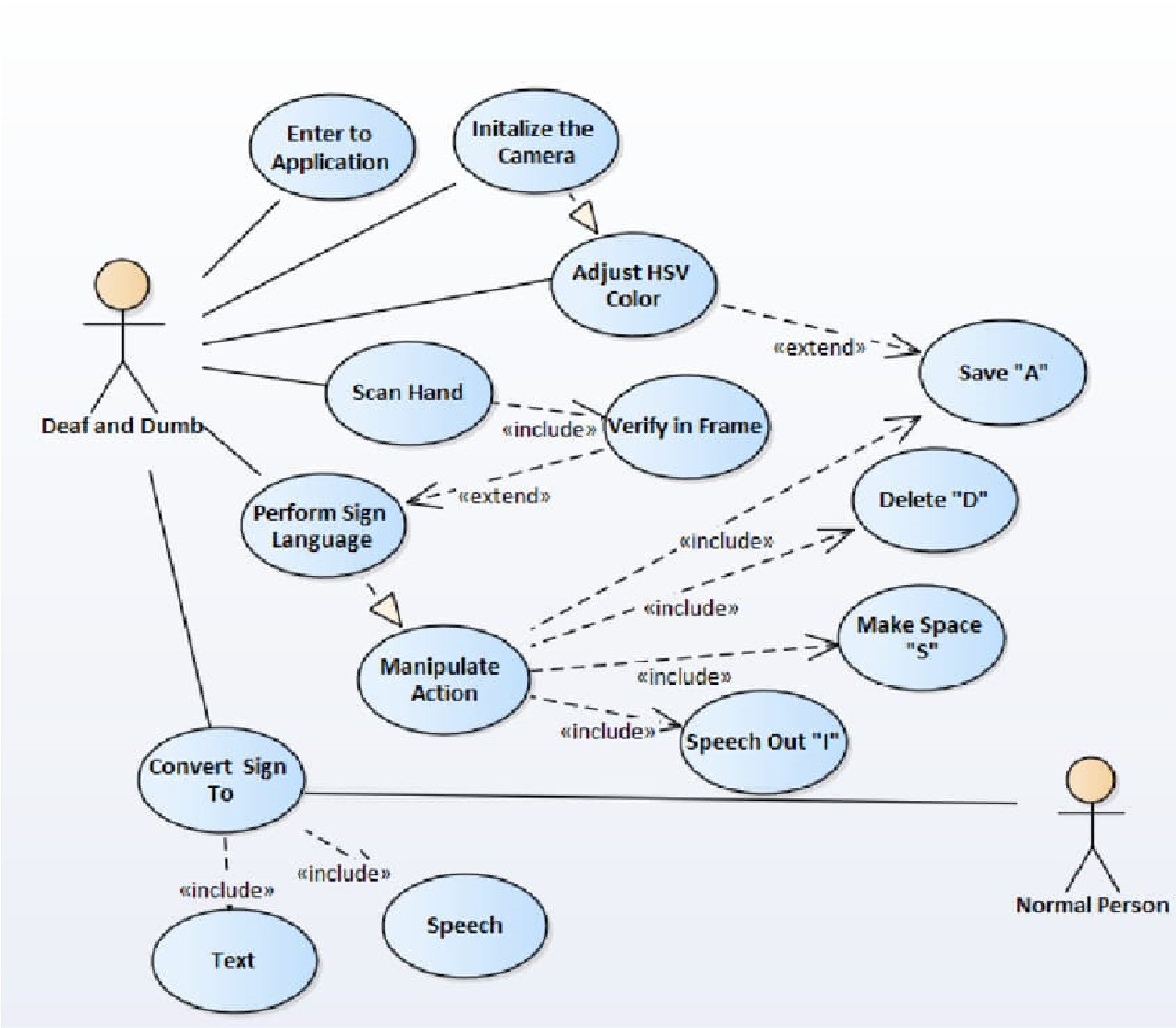

Figure 4.3: **Use Case Diagram For Sign Language**

In Figure 4.3 represents the Use Case Diagram is represented by Sign Savvy Recognition Using Machine Learning could include the following actors, use cases, and their relationships.

4.2.3 Class Diagram For The Sign Savvy Recognition using Machine Learning

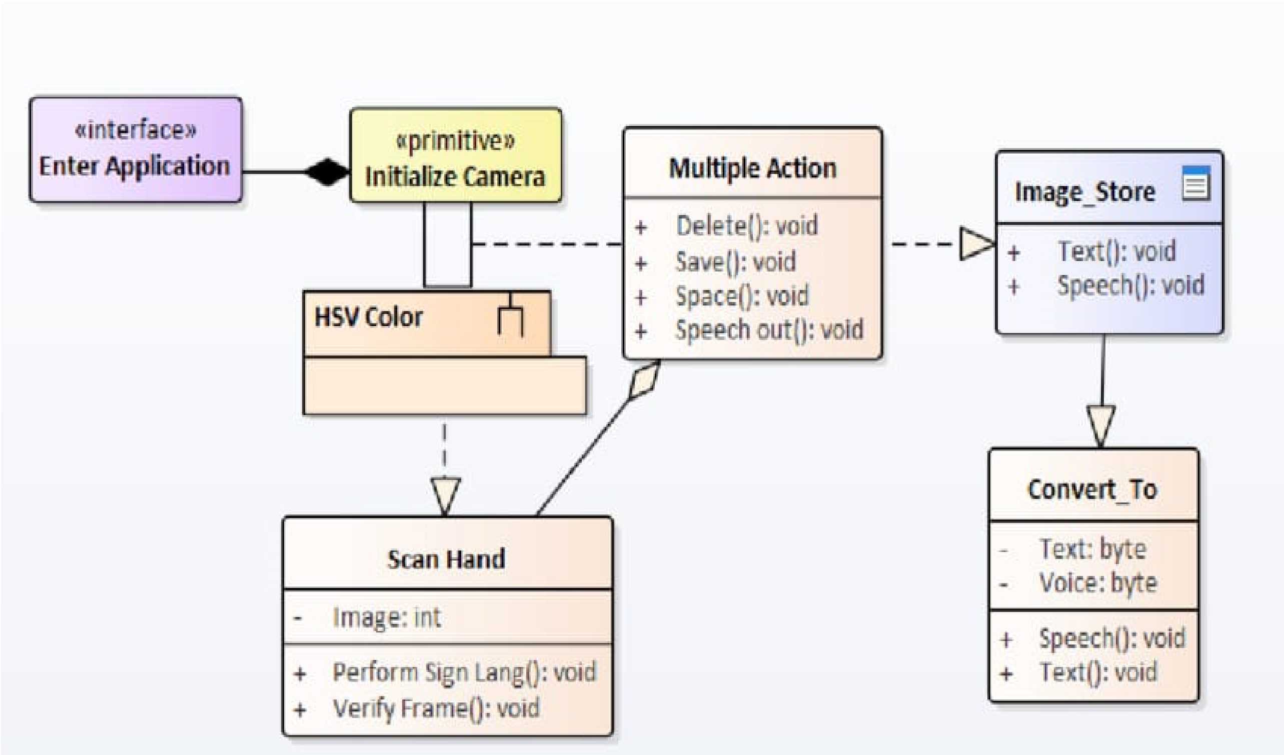

Figure 4.4: **Class Diagram Sign Savvy Recognition using Machine Learning**

In Figure 4.4 represents the Class Diagram is represented by provides a more detailed representation of the classes involved in a Sign Savvy Recognition using Machine Learning, including their attributes, methods, and relationships.

4.2.4 Sequence Diagram Sign Savvy Language Recognition System

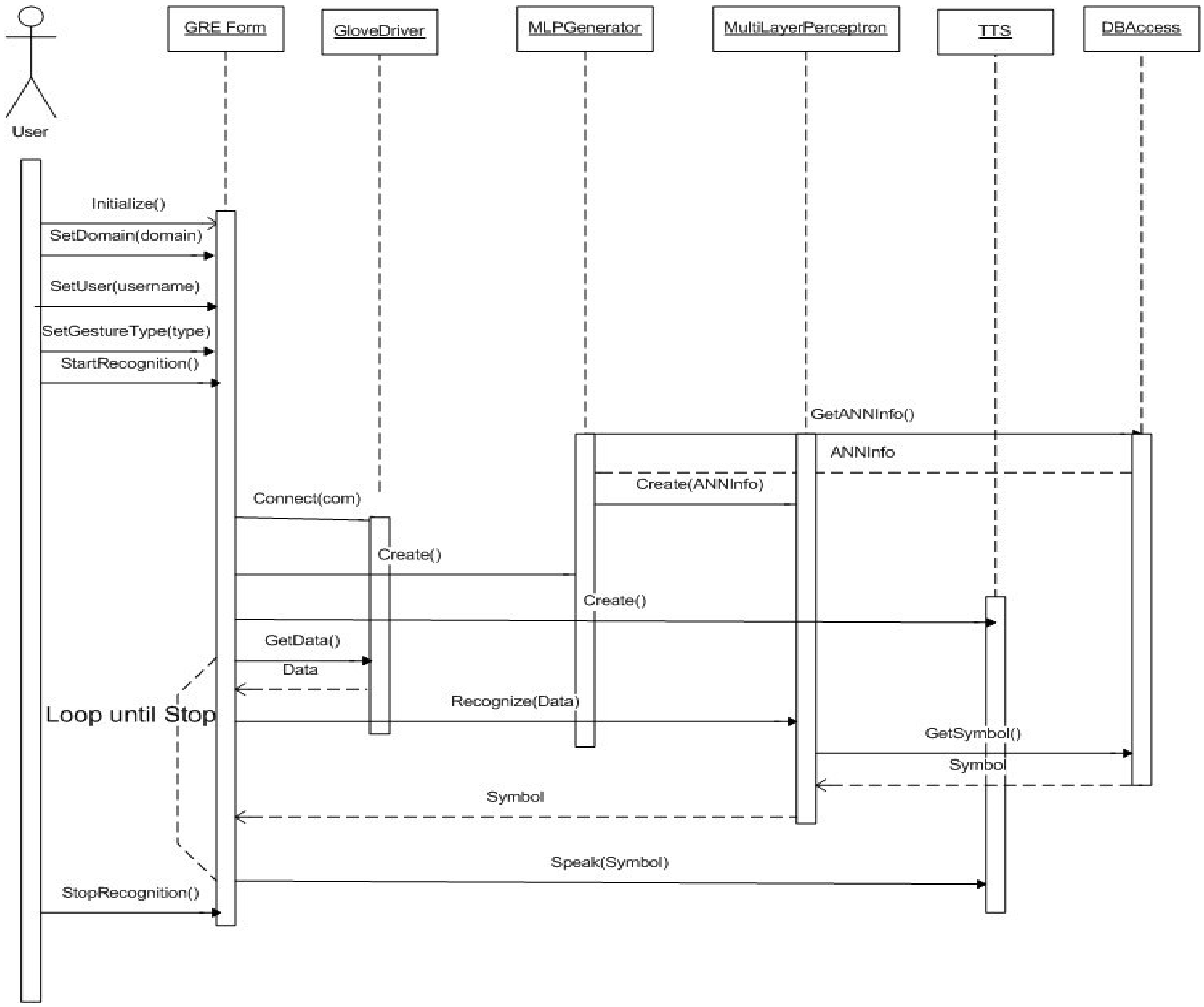

Figure 4.5: **Sequence Diagram of Sign Savvy Language Recognition System**

In Figure 4.5 Sequence Diagram is represented by Sign Savvy Recognition using Machine Learning would illustrate the interactions between different components of the system over time.

4.2.5 Collaboration Diagram Data Processing

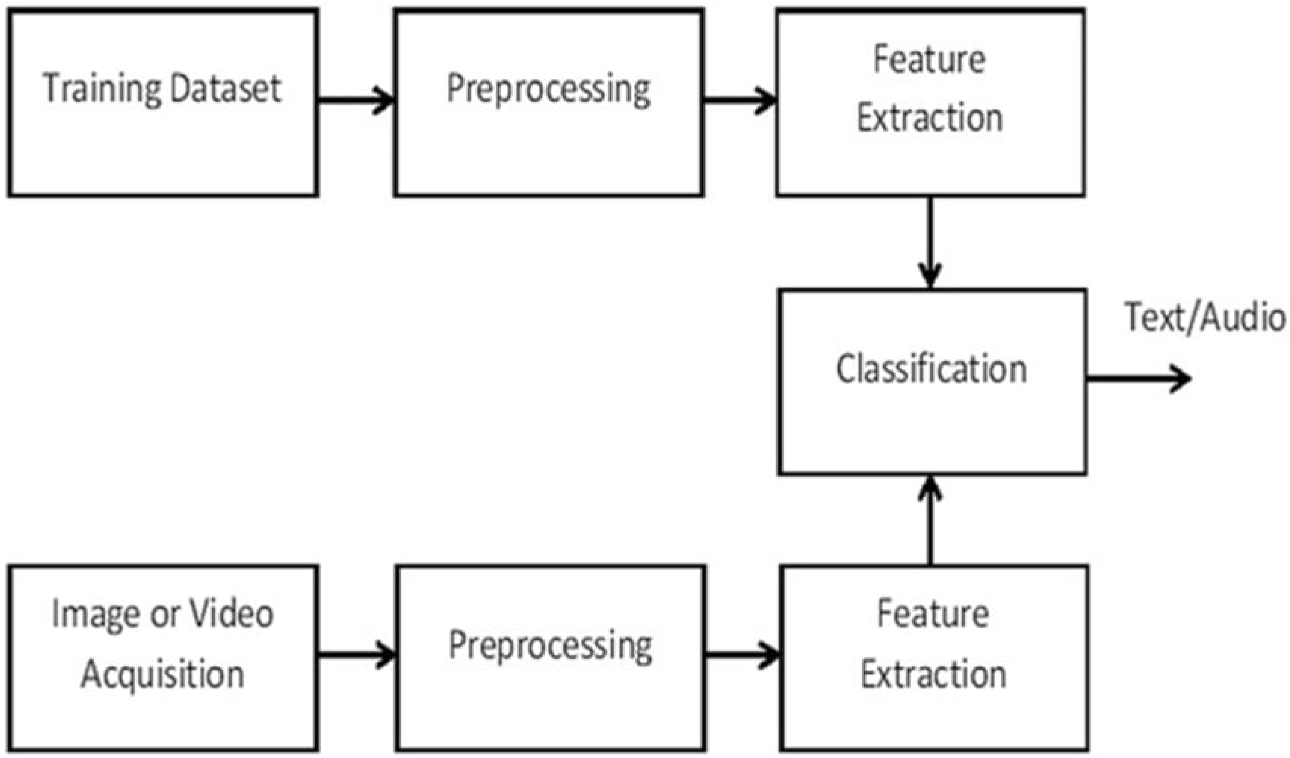

Figure 4.6: **Collaboration diagram Data Processing**

In Figure 4.6 Collaboration Diagram is represented by illustrates the communication and interactions between the User, System, and Administrator in a Sign Savvy Recognition using Machine Learning. Each object collaborates with others to perform specific tasks within the system.

4.2.6 Activity Diagram For Sign Language Recognition System

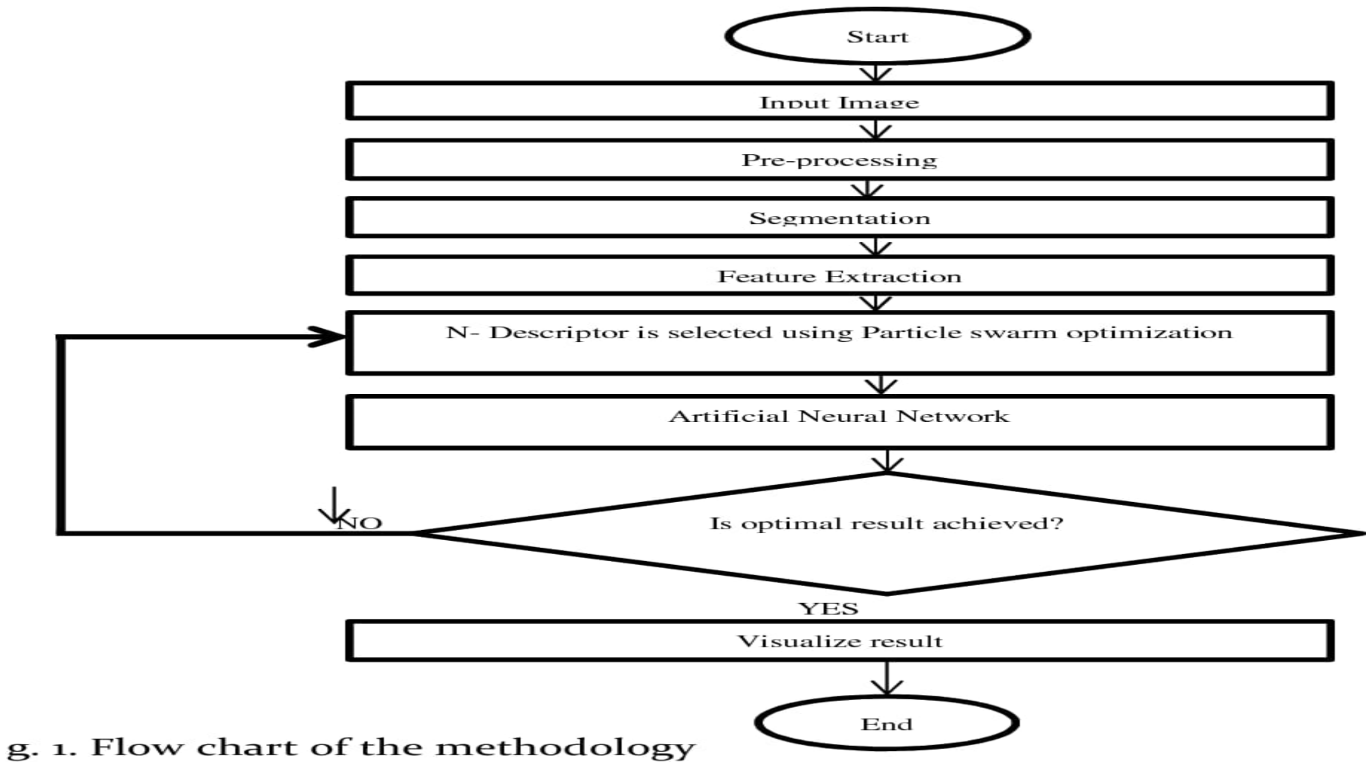

Figure 4.7: **Activity Diagram of Sign Language Recognition System**

In Figure 4.7 Activity Diagram is represented by outlines the sequence of steps involved in recognizing sign language gestures using machine learning within the system. Each step leads to the next until the process is completed.

4.3 Algorithm & Pseudo Code

4.3.1 Algorithm For sign Savvy language

Step 1: Data Collection: Gather a dataset of sign language gestures. This dataset should contain labeled examples of various signs in the target sign language. Data Preprocessing: Clean and preprocess the data. This may involve tasks such as resizing images, normalizing pixel values, and removing noise.
Step 2: Feature Extraction:Extract relevant features from the preprocessed data. For image-based approaches, this might involve techniques like hand segmentation, keypoint detection, or extracting spatial features from the hand region.
Step 3: Model Training:Split the dataset into training and validation sets. Train the selected model on the training data. This involves feeding the input features into the

model and adjusting its parameters to minimize a chosen loss function. Validate the model's performance on the validation set and iteratively adjust hyperparameters if needed.

Step 3: Deployment:Once satisfied with the model's performance, deploy it in a real-world setting. This could involve integrating it into a mobile application, web service, or other platforms where it can be used for real-time sign language recognition.

Step 4: Monitoring and Maintenance:Continuously monitor the system's performance in production. Collect feedback from users and identify areas for improvement. Maintain the system by periodically retraining the model with new data and updating it to adapt to changes in user behavior or input patterns.

Step 5: Accessibility Considerations:Ensure that the system is accessible to all users, including those with disabilities. This might involve providing alternative input methods or interfaces for users who are unable to perform sign language gestures.

Step 6: Ethical Considerations:Consider factors such as data privacy, model fairness, and potential biases that might affect the system's performance and usability. Involve members of the sign language community in the design and development process to ensure that the system meets their needs effectively.

4.3.2 Pseudo Code

```
import {
  KNNImageClassifier
} from 'deeplearn-knn-image-classifier';
import * as dl from 'deeplearn';

// Webcam Image size. Must be 227.
const IMAGE_SIZE = 227;
// K value for KNN. 10 means that we will take votes from 10 data points to classify each tensor.
const TOPK = 10;
// Percent confidence above which prediction needs to be to return a prediction.
const confidenceThreshold = 0.98

// Initial Gestures that need to be trained.
// The start gesture is for signalling when to start prediction
// The stop gesture is for signalling when to stop prediction
var words = ["start", "stop"];

/*
```

```
The Main class is responsible for the training and prediction of words.
It controls the webcam, user interface , as well as initiates the output of predicted words.
*/
class Main {
  constructor() {

  //This function sets up the webcam
  startWebcam() {
    navigator.mediaDevices.getUserMedia({
        video: {
          facingMode: 'user'
        },
        audio: false

    // Variables for training information for the user
    var exampleCountDisplay = document.getElementById('counter_' + btnType);
    var checkMark = document.getElementById('checkmark_' + btnType);

    // Create Gesture Card
    var gestureCard = document.createElement("div");
    gestureCard.className = "trained-gestures";

    var gestName = "";
    if (i == 0) {
      gestName = "Start";
    } else {
      gestName = "Stop";
    }
    var gestureName = document.createElement("h5");
    gestureName.innerText = gestName;
    gestureCard.appendChild(gestureName);
    this.trainedCardsHolder.appendChild(gestureCard);
    } else {
      alert('You haven\'t added any examples yet.\n\nAdd a Gesture, then perform the sign in front
          of the webcam.');
    }
  }

    var clearBtn = document.createElement('button');
    clearBtn.className = "clearButton";
    clearBtn.innerText = "Clear";
    this.trainingCommands.appendChild(clearBtn);

    // Change training class from none to specified class if training button is pressed
    trainBtn.addEventListener('mousedown', () => {
      this.train(i);
    });
```

4.4 Module Description

4.4.1 Vectorization (TF IDF vectorizer))

Vectorization Vectorization is the process of converting textual data into numerical vectors and is a process that is usually applied once the text is cleaned. It can help improve the execution speed and reduce the training time of your code. In this article, we will discuss some of the best techniques to perform vectorization. The TF-IDF (Term Frequency-Inverse Document Frequency) vectorizer is a common tool used in machine learning (ML) to convert textual data into a matrix of TF-IDF features.

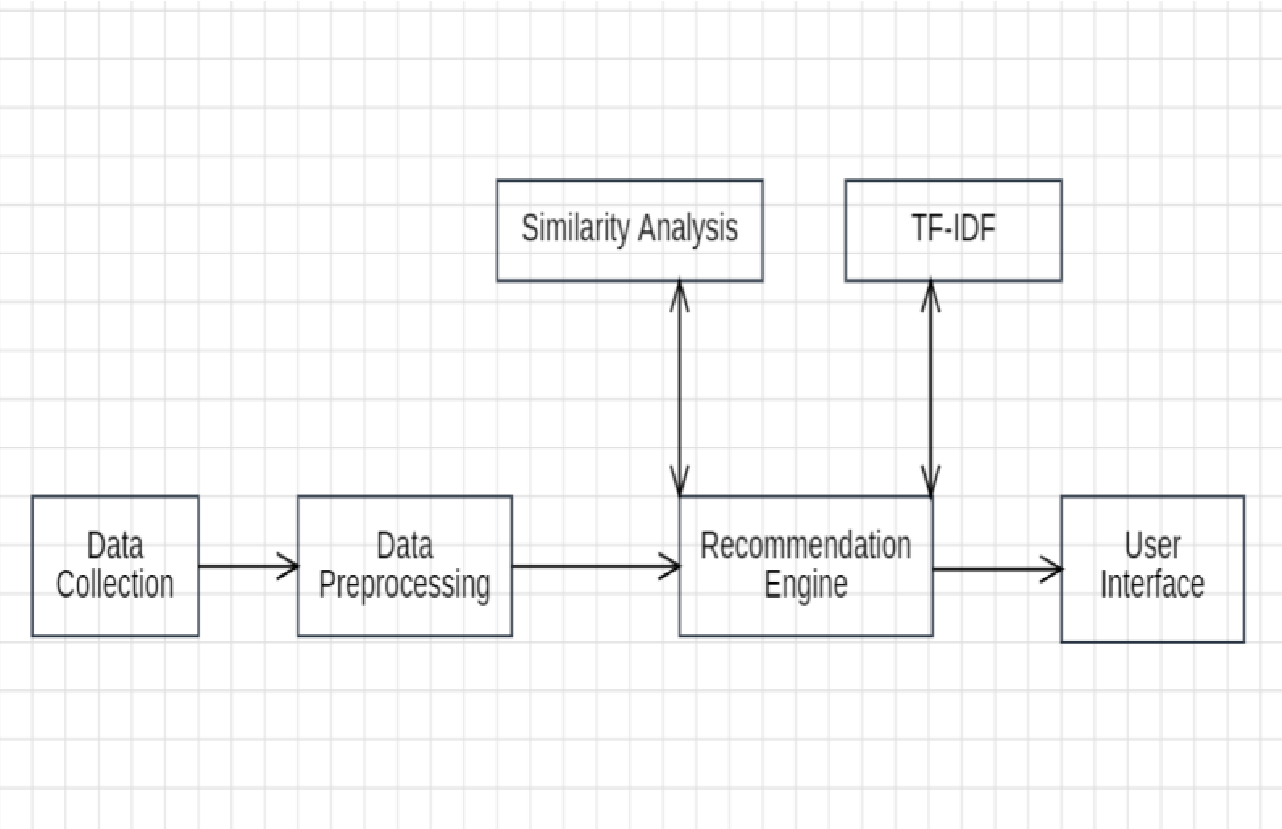

Figure 4.8: **Architecture Diagram For vectorization**

4.4.2 Data Preprocessing

Data Preprocessing: Clean and preprocess the data to extract relevant features. This might involve resizing images, normalization, denoising, etc. For video data, you might need to extract frames and possibly use techniques like optical flow to capture motion information.

Feature Extraction: Extract features from the preprocessed data. Depending on the approach, this could involve techniques like hand tracking, keypoint detection, or

deep learning-based feature extraction methods.
Model Selection: Choose an appropriate machine learning model for sign language recognition. This could range from traditional machine learning algorithms like Support Vector Machines (SVMs) or Random Forests to deep learning models such as Convolutional Neural Networks (CNNs) or Recurrent Neural Networks (RNNs).
Model Training: Train the selected model on the preprocessed data. This involves splitting the data into training and validation sets, feeding it into the model, and optimizing the model's parameters to minimize a chosen loss function.
Evaluation: Evaluate the trained model's performance on a separate test set. Common evaluation metrics for classification tasks include accuracy, precision, recall, and F1 score.

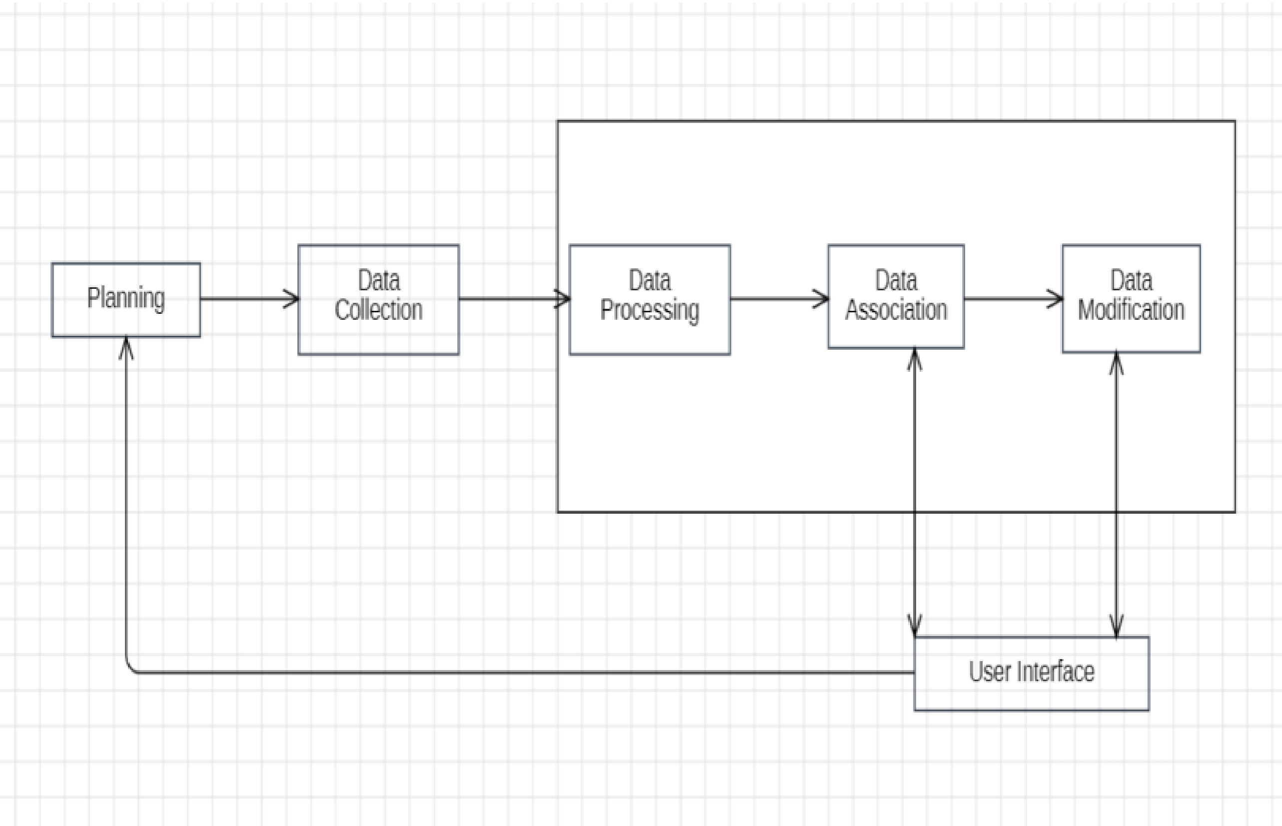

Figure 4.9: **Architecture Diagram For Data Collection and Data Processing**

4.4.3 Web Scraping Algorithm:

Web Scraping Algorithm: Web scraping is a technique used to extract data from websites. It allows auto- mated retrieval of information from web pages, enabling tasks such as data collec- tion, analysis, and monitoring. Web scraping involves extracting specific data ele- ments or information from the HTML structure of web pages. This can include text, images, links, tables, and more, depending on the requirements of the scraping task. Web scraping algorithms incorporate error handling mechanisms to deal with poten- tial issues such as connection errors, timeouts, or changes in the website's structure. This ensures robustness and reliability in data retrieval. Scraping programs adhere to legal and ethical guidelines, including respect for website terms of service, robots.txt directives, and copyright laws. They also avoid overloading servers with excessive requests. Figure 4.10: Architecture Diagram For Web Scraping Algorithm 4.5 Steps to implement the project 4.5.1 Data Collection and Processing The data was collected through various datasets that are available online which included farmer name, age, educated or not, acres of land, crop cultivated, address 20

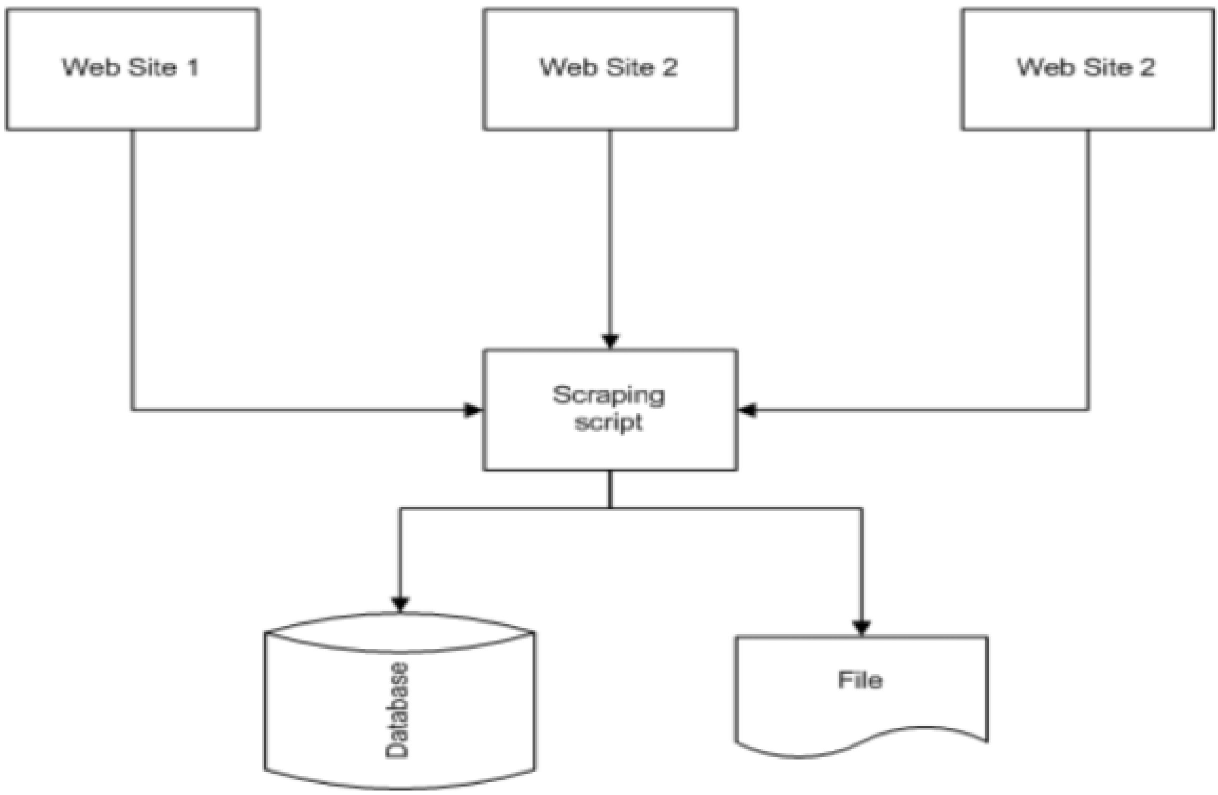

Figure 4.10: **Architecture Diagram For Web Scraping Algorithm**

Chapter 5

IMPLEMENTATION AND TESTING

5.1 Input and Output

5.1.1 Input Design For Sign Savvy Recognition

Figure 5.1: **Input Design For The Sign Savvy Recognition**

The above figure 5.1 shows the input design for the sign savvy language.

5.1.2 Output Design For The Sign Savvy Recognition

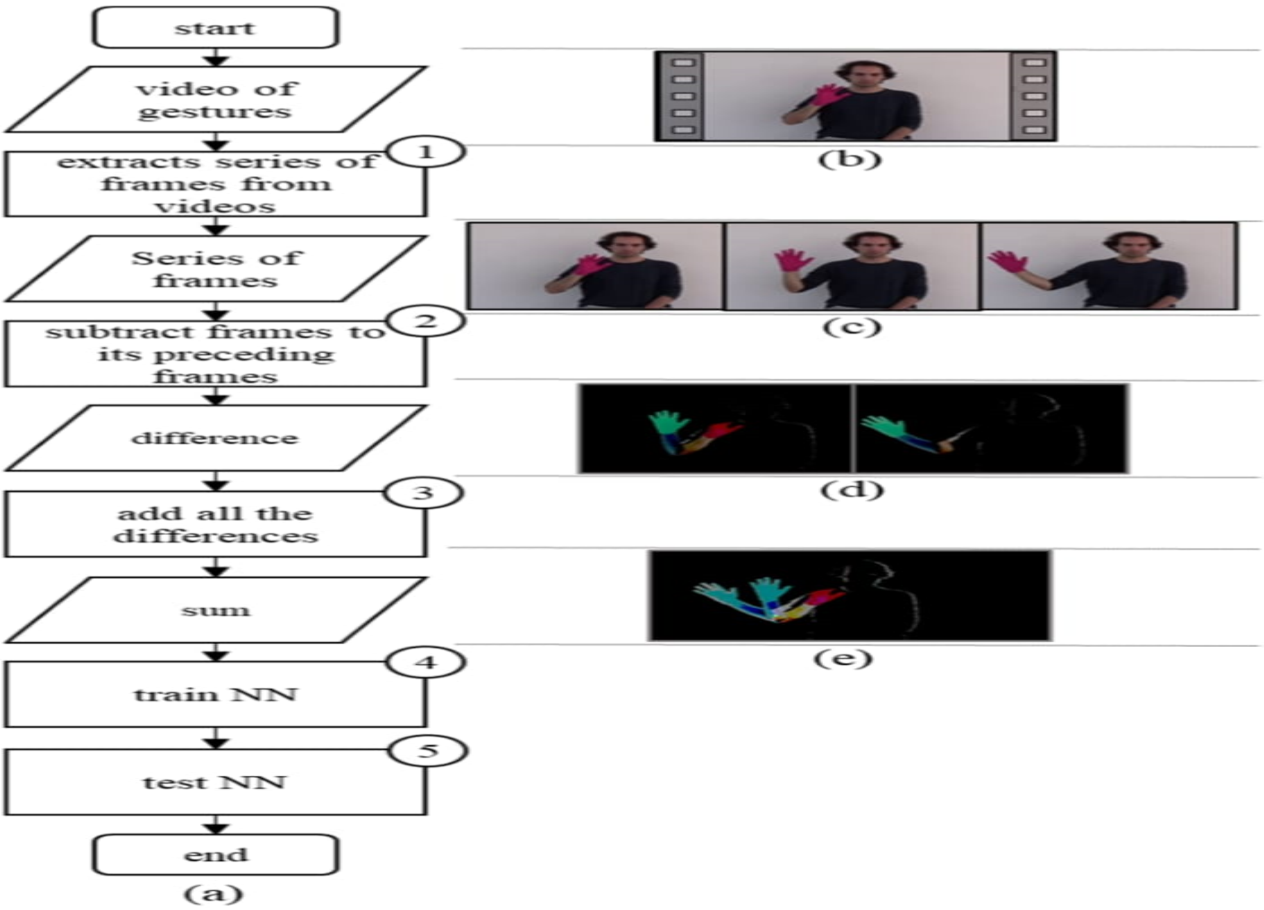

Figure 5.2: **Output Design The Sign Savvy Recognition**

The above figure 5.2 shows the output design for the sign savvy language.which gives the conversion of the sign language into the group of words.

5.2 Testing

Testing a sign language recognition system using machine learning is crucial to ensure its accuracy, reliability, and robustness in real-world scenarios.Test individual components of the system, such as data preprocessing, feature extraction, and model training, using unit tests. Verify that each component functions correctly and produces expected outputs.Test the integration of different components to ensure they work together seamlessly.Determine whether errors are due to limitations in the training data, model architecture, or implementation. Use techniques like confusion matrix analysis, error visualization, and qualitative analysis of misclassified examples to gain insights into model performance.Test the accessibility features of the system to ensure it is usable by individuals with disabilities, including those with visual or motor impairments.Evaluate the system's compatibility with assistive technologies and adherence to accessibility standards such as WCAG.

5.3 Types of Testing

5.3.1 Unit testing

Unit testing is the process where you test the smallest functional unit of code. Software testing helps ensure code quality, and it's an integral part of software development. It's a software development best practice to write software as small, functional units then write a unit test for each code unit.

Input

```
@charset "UTF-8";
/*!
 * animate.css -http://daneden.me/animate
 * Version - 3.7.0
 * Licensed under the MIT license - http://opensource.org/licenses/MIT
 *
 * Copyright (c) 2018 Daniel Eden
 */

@-webkit-keyframes bounce {
    from , 20, 53-webkit-animation-timing-function: cubic-bezier(0.215, 0.61, 0.355, 1 ;
        animation-timing-function: cubic-bezier(0.215, 0.61, 0.355, 1);
    }
    40, 43-webkit-animation-timing-function: cubic-bezier(0.755, 0.05, 0.855, 0.06 ;
        animation-timing-function: cubic-bezier(0.755, 0.05, 0.855, 0.06);
        -webkit-transform: translate3d(0, -30px, 0);
        transform: translate3d(0, -30px, 0);
    }
    70 -webkit-animation-timing-function: cubic-bezier(0.755, 0.05, 0.855, 0.06 ;
        animation-timing-function: cubic-bezier(0.755, 0.05, 0.855, 0.06);
        -webkit-transform: translate3d(0, -15px, 0);
        transform: translate3d(0, -15px, 0);
    }
    90 -webkit-transform: translate3d(0, -4px, 0 ;
        transform: translate3d(0, -4px, 0);
    }
}
```

```
@keyframes bounce {
    from ,  20, 53-webkit-animation-timing-function: cubic-bezier(0.215, 0.61, 0.355, 1 ;
        animation-timing-function: cubic-bezier(0.215, 0.61, 0.355, 1);
        -webkit-transform: translate3d(0, 0, 0);
        transform: translate3d(0, 0, 0);
    }
    40, 43-webkit-animation-timing-function: cubic-bezier(0.755, 0.05, 0.855, 0.06 ;
        animation-timing-function: cubic-bezier(0.755, 0.05, 0.855, 0.06);
        -webkit-transform: translate3d(0, -30px, 0);
        transform: translate3d(0, -30px, 0);
    }
    70 -webkit-animation-timing-function: cubic-bezier(0.755, 0.05, 0.855, 0.06 ;
        animation-timing-function: cubic-bezier(0.755, 0.05, 0.855, 0.06);
        -webkit-transform: translate3d(0, -15px, 0);
        transform: translate3d(0, -15px, 0);
    }
    90 -webkit-transform: translate3d(0, -4px, 0 ;
        transform: translate3d(0, -4px, 0);
    }
}

.bounce {
    -webkit-animation-name: bounce;
    animation-name: bounce;
    -webkit-transform-origin: center bottom;
    transform-origin: center bottom;
}

}

.shake {
    -webkit-animation-name: shake;
    animation-name: shake;
}
@keyframes swing {
    20 -webkit-transform: rotate3d(0, 0, 1, 15deg ;
        transform: rotate3d(0, 0, 1, 15deg);
    }
    40 -webkit-transform: rotate3d(0, 0, 1, -10deg ;
        transform: rotate3d(0, 0, 1, -10deg);
    }
    60 -webkit-transform: rotate3d(0, 0, 1, 5deg ;
        transform: rotate3d(0, 0, 1, 5deg);
    }
    80 -webkit-transform: rotate3d(0, 0, 1, -5deg ;
        transform: rotate3d(0, 0, 1, -5deg);
    }
}
```

5.3.2 Integration testing

Integration testing (sometimes called integration and testing, abbreviated IT) is the phase in software testing in which the whole software module is tested or if it consists of multiple software modules they are combined and then tested as a group.

Input

```
<!--
Index.html is the User Interface for the Sign Language Translator.
It contains the Welcome Screen, Title Bar, Status Bar, Translator Window, and Video Call frame.

To Learn More how to run this code, visit README.md
Author: Sufiyaan Nadeem
Fix Alerts
-->

<!DOCTYPE html>
<html lang="en">

<head>
    <!--Sets up stylesheet and script file linking. Adds favicon and site title as well-->
    <meta charset="UTF-8">
    <link rel="shortcut icon" type="image/png" href="Images/asl_logo.ico" />
    <title>Sign Language Recognition| Sufiyaan</title>
    <link href="https://fonts.googleapis.com/css?family=Google+Sans:100,300,400,500" rel="stylesheet
        ">
    <link rel="stylesheet" href="CSS/style.css">
    <link rel="stylesheet" href="CSS/animate.css">
    <script src="dist/build.js"></script>
</head>

<body>
    <!--Welcome Screen-->
    <div id="welcomeContainer" class="animated">
        <div id="welcomeScreen">
            <h1>Welcome to Sign Language Recognition</h1>
            <h3 class="animated fadeInUp delay-2s">...</h3>
        </div>
        <button id="proceedButton">Proceed</button>
    </div>

    <!--Title bar explains the stage of the program
    (eg: instructions for Training, Prediction, and Video Chat)-->
    <div id="titleBar">
        <h1 id="stage">Train Gestures</h1>
        <h3 id="steps">Train about 30 samples of your Start Gesture and 30 for your idle, Stop
            Gesture.</h3>
```

```
        <button id="nextButton" class="animated flash delay-3s">Next</button>
        <button id="predictButton" class="animated flash slideInRight faster">Translate</button>
        <button id="backButton" class="animated slideInLeft faster">Back to Training</button>
        <button id="videoCallBtn" class="videoCallBtn animated slideInRight faster">Video Call</button>
    </div>

    <!--Status bar shows the status of translation-->
    <div id="status">
        <p id="status-text">Status: Not Ready</p>
    </div>
        <div id="videoHolder" class="videoContainerTrain">
            <video id="video" class="videoTrain" src='' muted autoplay playsinline></video>
            <iframe src="https://tokbox.com/embed/embed/ot-embed.js?embedId=f37957b6-0f91-4fc5-90ce-f818cc85b5bf&room=DEFAULT_ROOM&iframe=true"
                width=650 height=370 allow="microphone; camera" id="videoCall"></iframe>
        </div>

        <!--Training Holder displays the screen where users add and train new gestures-->
        <div id="trainingHolder">
            <h5 id="add-gesture">Add Gesture</h5>
            <img src="Images/plus_sign.svg" alt="Plus Sign" id="plus_sign" class="plus_sign animated">
            <form id="add-word" autocomplete="off">
                <input type="text" id="new-word" placeholder="New Gesture Title">
                <input type="submit" id="submit-word" value="Add Word &#9658;">
            </form>
            <button id="doneRetrain" class="doneRetrain">Done Retraining</button>
    </div>

    <!--Trained Card Holder contains all the trained gestures' cards.
    It is outside the Training and Translation Windows because it has to be displayed on both screens-->
    <div id="trainedCardsHolder">
    </div>
</body>

</html>
```

5.3.3 System testing

System testing examines every component of an application to make sure that they work as a complete and unified whole. A QA team typically conducts system testing after it checks individual modules with functional or user story testing and then each component through integration testing.

Input

```
import {
  KNNImageClassifier
} from 'deeplearn-knn-image-classifier';
import * as dl from 'deeplearn';

// Webcam Image size. Must be 227.
const IMAGE_SIZE = 227;
// K value for KNN. 10 means that we will take votes from 10 data points to classify each tensor.
const TOPK = 10;
// Percent confidence above which prediction needs to be to return a prediction.
const confidenceThreshold = 0.98

// Initial Gestures that need to be trained.
// The start gesture is for signalling when to start prediction
// The stop gesture is for signalling when to stop prediction
var words = ["start", "stop"];
/*
The Main class is responsible for the training and prediction of words.
It controls the webcam, user interface, as well as initiates the output of predicted words.
*/
class Main {
  constructor() {

  //This function sets up the webcam
  startWebcam() {
    navigator.mediaDevices.getUserMedia({
        video: {
          facingMode: 'user'
        },
        audio: false
    var gestName = "";
    if (i == 0) {
      gestName = "Start";
    } else {
      gestName = "Stop";
    }
    var gestureName = document.createElement("h5");
    gestureName.innerText = gestName;
    gestureCard.appendChild(gestureName);
    this.trainedCardsHolder.appendChild(gestureCard);
    } else {
      alert('You haven\'t added any examples yet.\n\nAdd a Gesture, then perform the sign in front
          of the webcam.');
    }
  }

  /*This creates the training and clear buttons for the new gesture. It also creates the
    Gesture Card.*/
```

```
  createTrainingBtns(i) { //i is the index of the new word
    // Create Train and Clear Buttons
    var trainBtn = document.createElement('button');
    trainBtn.className = "trainBtn";
    trainBtn.innerText = "Train";
    this.trainingCommands.appendChild(trainBtn);

    var clearBtn = document.createElement('button');
    clearBtn.className = "clearButton";
    clearBtn.innerText = "Clear";
    this.trainingCommands.appendChild(clearBtn);

    // Change training class from none to specified class if training button is pressed
    trainBtn.addEventListener('mousedown', () => {
      this.train(i);
    });
```

5.3.4 Test Result

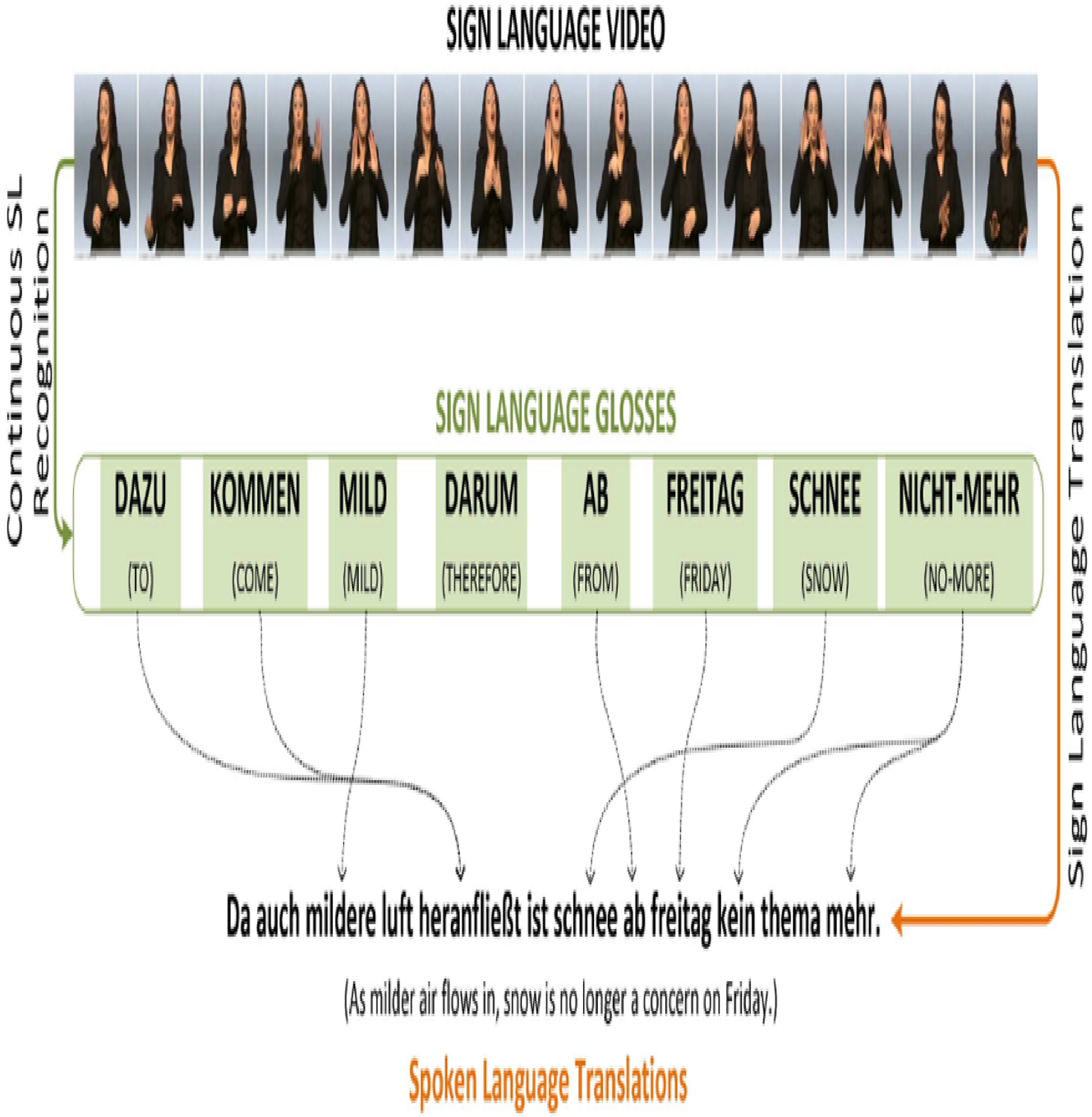

Figure 5.3: **Test Image**

The above figure 5.3 Represents about the test image translation of the sign language converting into the group of words by using machine learning.

Chapter 6

RESULTS AND DISCUSSIONS

6.1 Efficiency of the Proposed System

The efficiency of a Sign Savvy Recognition using Machine Learning can be evaluated based on several factors:
Recognition Accuracy: The primary measure of efficiency is the system's ability to accurately recognize sign language gestures. Higher accuracy means the system can correctly interpret and translate a wide range of gestures, leading to more effective communication.
Processing Speed: The speed at which the system can process input, recognize gestures, and provide output is crucial, especially in real-time applications. Lower latency ensures a more responsive system, which is essential for interactive communication.
Training Time and Maintenance: The time required to train the machine learning model and update it with new data affects the system's agility and adaptability.
User Experience: The overall user experience, including ease of use, responsiveness, and reliability, plays a crucial role in determining the system's efficiency.
Energy Efficiency: For embedded or mobile applications, energy efficiency is a key consideration.
Scalability: The system's ability to scale to accommodate growing data volumes and user demands is essential for its long-term viability. Scalable architectures and efficient algorithms allow the system to handle increased workload without sacrificing performance.

6.2 Comparison of Existing and Proposed System

To compare an existing Sign Savvy Recognition using Machine Learning with a proposed system,By comparing these aspects between the existing and proposed systems, stakeholders can gain insights into the potential benefits and improvements offered by the proposed Sign Language Recognition System using Machine Learning. Additionally, empirical testing and validation can provide quantitative data to support the comparison and inform decision-making.

Existing system:

Existing Sign Language Recognition System using Machine Learning could be based on various architectures and technologies. Here, I'll provide a general overview of how such a system might be structured:

Data Collection and Preprocessing: Utilizes cameras or sensors to capture sign language gestures. Preprocesses the captured data to enhance quality and prepare it for input into the machine learning model. Preprocessing may involve techniques such as image enhancement, noise reduction, and feature extraction.

Machine Learning Model: Employs machine learning algorithms to recognize sign language gestures. Commonly uses deep learning architectures such as Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), or Transformer-based models. Trained on labeled datasets of sign language gestures to learn patterns and features associated with different signs.

Output Visualization: Displays the recognized sign language gesture to the user. May present the output as text, images, animations, or other suitable formats. Provides feedback to the user to ensure the correct interpretation of the recognized gesture.

Proposed system:

This proposed Sign Language Recognition System emphasizes cutting-edge machine learning techniques, personalized user experiences, and inclusivity to empower individuals with diverse communication needs. By leveraging advanced technologies and user-centric design principles, the system aims to bridge communication barriers and foster greater accessibility and inclusion for the sign language community.

Gesture Recognition: Employs the trained machine learning model for gesture recognition. Utilizes optimized inference algorithms to ensure fast and efficient recognition of gestures in real-time. Implements techniques for handling variations in hand movements, lighting conditions, and background clutter.

Output Visualization and Feedback: Provides intuitive and user-friendly interfaces for displaying recognized sign language gestures. Incorporates visual aids, such as text captions or animations, to enhance understanding and communication.

Indian Sign Language (ISL): - Indian Sign Language (ISL) is a complete language with its own grammar, syntax, vocabulary. And certain languages. It is used by over 5 million deaf people in India. Currently, there is no publicly available data set in ISL for sign language recognition (SLR) testing methods. In this connection, the dictionary presents the Ketik language dataset - Include - 0.27 million frames ISL data set in 4,287 videos 26-word symbols in 153-word range. Reported Experienced signature to provide similarities related to natural conditions. A subset of 50-word symbols is selected for all word categories to describe INCLUDE-50 for rapid testing of SLR methods by hyperparameter tuning. As a group SLR study in ISL, we are looking Many deep neural networks consisting of various techniques, e.g., extraction, Coding and coding.

6.3 Sample Code

```
1 0 obj
<</Metadata 2 0 R/OCProperties<</D<</ON[5 0 R 26 0 R 53 0 R 80 0 R]/Order 81 0 R/RBGroups[]>>/OCGs[5 0 R 26 0 R 53 0 R 80 0 R]>>/Pages 3 0 R/Type/Catalog>>
endobj
2 0 obj
<</Length 40879/Subtype/XML/Type/Metadata>>stream
<?xpacket begin="" id="W5M0MpCehiHzreSzNTczkc9d"?>
<x:xmpmeta xmlns:x="adobe:ns:meta/" x:xmptk="Adobe XMP Core 5.6-c111 79.158366, 2015/09/25-01:12:00
        ">
   <rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#">
      <rdf:Description rdf:about=""
            xmlns:dc="http://purl.org/dc/elements/1.1/"
            xmlns:xmp="http://ns.adobe.com/xap/1.0/"
            xmlns:xmpGImg="http://ns.adobe.com/xap/1.0/g/img/"
            xmlns:xmpMM="http://ns.adobe.com/xap/1.0/mm/"
            xmlns:stRef="http://ns.adobe.com/xap/1.0/sType/ResourceRef#"
            xmlns:stEvt="http://ns.adobe.com/xap/1.0/sType/ResourceEvent#"
            xmlns:illustrator="http://ns.adobe.com/illustrator/1.0/"
            xmlns:xmpTPg="http://ns.adobe.com/xap/1.0/t/pg/"
            xmlns:stDim="http://ns.adobe.com/xap/1.0/sType/Dimensions#"
            xmlns:stFnt="http://ns.adobe.com/xap/1.0/sType/Font#"
            xmlns:xmpG="http://ns.adobe.com/xap/1.0/g/"
            xmlns:pdf="http://ns.adobe.com/pdf/1.3/">
         <dc:format>application/pdf</dc:format>
         <dc:title>
            <rdf:Alt>
               <rdf:li xml:lang="x-default">Halfchecked</rdf:li>
            </rdf:Alt>
         </dc:title>
         <xmp:CreatorTool>Adobe Illustrator CC 2015 (Windows)</xmp:CreatorTool>
         <xmp:CreateDate>2018-11-28T14:29:08-04:00</xmp:CreateDate>
         <xmp:ModifyDate>2018-11-28T16:41:44-05:00</xmp:ModifyDate>
         <xmp:MetadataDate>2018-11-28T16:41:44-05:00</xmp:MetadataDate>
```

```
            <xmp:Thumbnails>
                <rdf:Alt>
                    <rdf:li rdf:parseType="Resource">
                        <xmpGImg:width>256</xmpGImg:width>
                        <xmpGImg:height>160</xmpGImg:height>

                </rdf:Alt>
            </xmp:Thumbnails>
            <xmpMM:RenditionClass>proof:pdf</xmpMM:RenditionClass>
                <rdf:Bag>
                    <rdf:li rdf:parseType="Resource">
                        <stFnt:fontName>ProductSans-Regular</stFnt:fontName>
                        <stFnt:fontFamily>Product Sans</stFnt:fontFamily>
                        <stFnt:fontFace>Regular</stFnt:fontFace>
                        <stFnt:fontType>TrueType</stFnt:fontType>
                        <stFnt:versionString>Version 1.009;PS 1.000;
                    </rdf:li>
                </rdf:Bag>
            </xmpTPg:Fonts>
            <xmpTPg:PlateNames>
                <rdf:Seq>
                    <rdf:li>Cyan</rdf:li>
                    <rdf:li>Magenta</rdf:li>
                    <rdf:li>Yellow</rdf:li>
                    <rdf:li>Black</rdf:li>
                </rdf:Seq>
            </xmpTPg:PlateNames>
            <xmpTPg:SwatchGroups>
                <rdf:Seq>
                    <rdf:li rdf:parseType="Resource">
                        <xmpG:groupName>Default Swatch Group</xmpG:groupName>
                        <xmpG:groupType>0</xmpG:groupType>
                        <xmpG:Colorants>
                            <rdf:Seq>
                                <rdf:li rdf:parseType="Resource">
                                    <xmpG:swatchName>R=57 G=181 B=74</xmpG:swatchName>
                                    <xmpG:mode>RGB</xmpG:mode>
                                    <xmpG:type>PROCESS</xmpG:type>
                                </rdf:li>
                                <rdf:li rdf:parseType="Resource">
                                    <xmpG:swatchName>R=0 G=146 B=69</xmpG:swatchName>
                                    <xmpG:mode>RGB</xmpG:mode>
                                    <xmpG:type>PROCESS</xmpG:type>
                                </rdf:li>
```

Output:

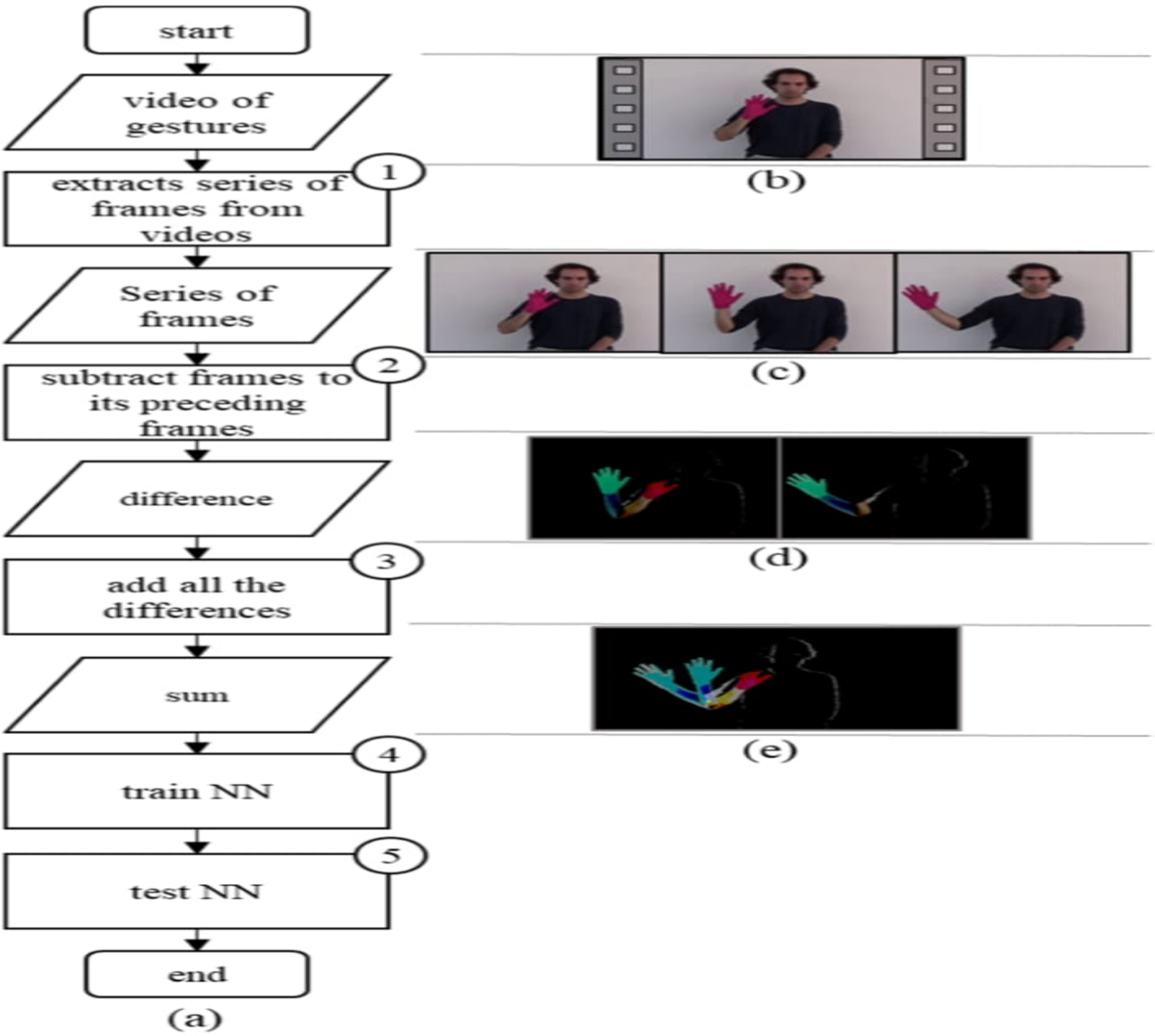

Figure 6.1: **Train Gestures**

The above figure 6.1' shows the output design for the sign savvy language.which gives the conversion of the sign language into the group of words.

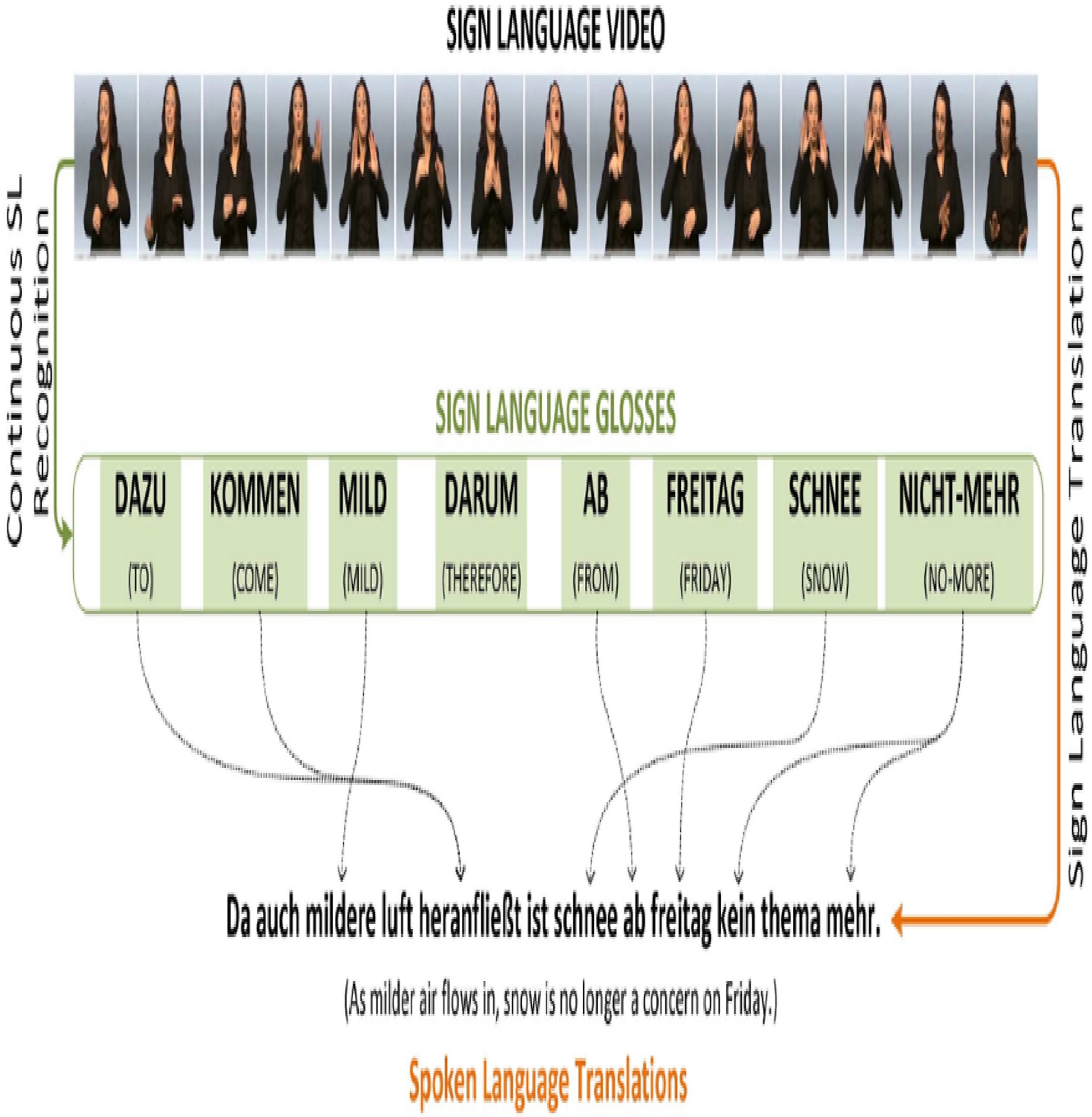

Figure 6.2: **Translation Of The Sign Video Language**

The above figure 6.2 Represents about the translation of the sign language converting into the group of words by using machine learning.

Chapter 7

CONCLUSION AND FUTURE ENHANCEMENTS

7.1 Conclusion

In conclusion, the Sign Savvy Recognition Using Machine Learning presents a promising solution to bridge communication barriers between individuals who are deaf or hard of hearing and those who do not understand sign language. Through the integration of computer vision and machine learning techniques, significant progress has been made in accurately interpreting sign language gestures in real-time.
The Sign Savvy Recognition Using Machine Learning is a promising technology that holds immense potential to bridge communication barriers for individuals who are deaf or hard of hearing. Through the integration of computer vision algorithms and machine learning techniques, significant progress has been made in accurately interpreting sign language gestures in real-time. The literature review highlights the diversity of approaches, including deep learning-based methods such as convolutional neural networks (CNNs) and the integration of hand tracking devices like the Leap Motion Controller.
These advancements signify the feasibility of developing robust and efficient sign language recognition systems.With the recent advancement in machine learning and computational intelligence methods, intelligent systems in sign language recognition continue to attract academic researchers and industrial practitioners' attention.The countries and academic institutions with a large number of published articles and solid international collaborations have been identified and presented in this paper. It is expected that this study will be an opportunity for the researcher in the countries with fewer collaborations to broaden their research collaborations.

7.2 Future Enhancements

By focusing on these future enhancements, the Sign Savvy Recognition Using Machine Learning can continue to evolve and make a meaningful impact in improving communication accessibility for individuals who rely on sign language as their primary means of communication.

Multimodal Integration: Explore the integration of multiple modalities such as video, depth, and skeletal data to improve the robustness and accuracy of sign language recognition systems.

Continual Learning: Investigate techniques for continual learning to adapt the model to new sign language gestures and variations over time, ensuring the system remains up-to-date and responsive to user needs.

Multilingual Support: Extend the system to support recognition of sign language gestures from different languages and dialects, catering to a broader range of users worldwide.

User Interface Optimization: Enhance the user interface to provide a seamless and intuitive experience for both sign language users and non-signers, incorporating feedback mechanisms to improve user interaction and satisfaction.

Real-world Deployment: Conduct extensive field testing and user studies to evaluate the system's performance in real-world scenarios and gather feedback for further refinement and improvement.

Privacy and Accessibility: Address privacy concerns by implementing privacy-preserving mechanisms for data collection and processing, while also ensuring accessibility features for users with diverse needs.

Chapter 8

PLAGIARISM REPORT

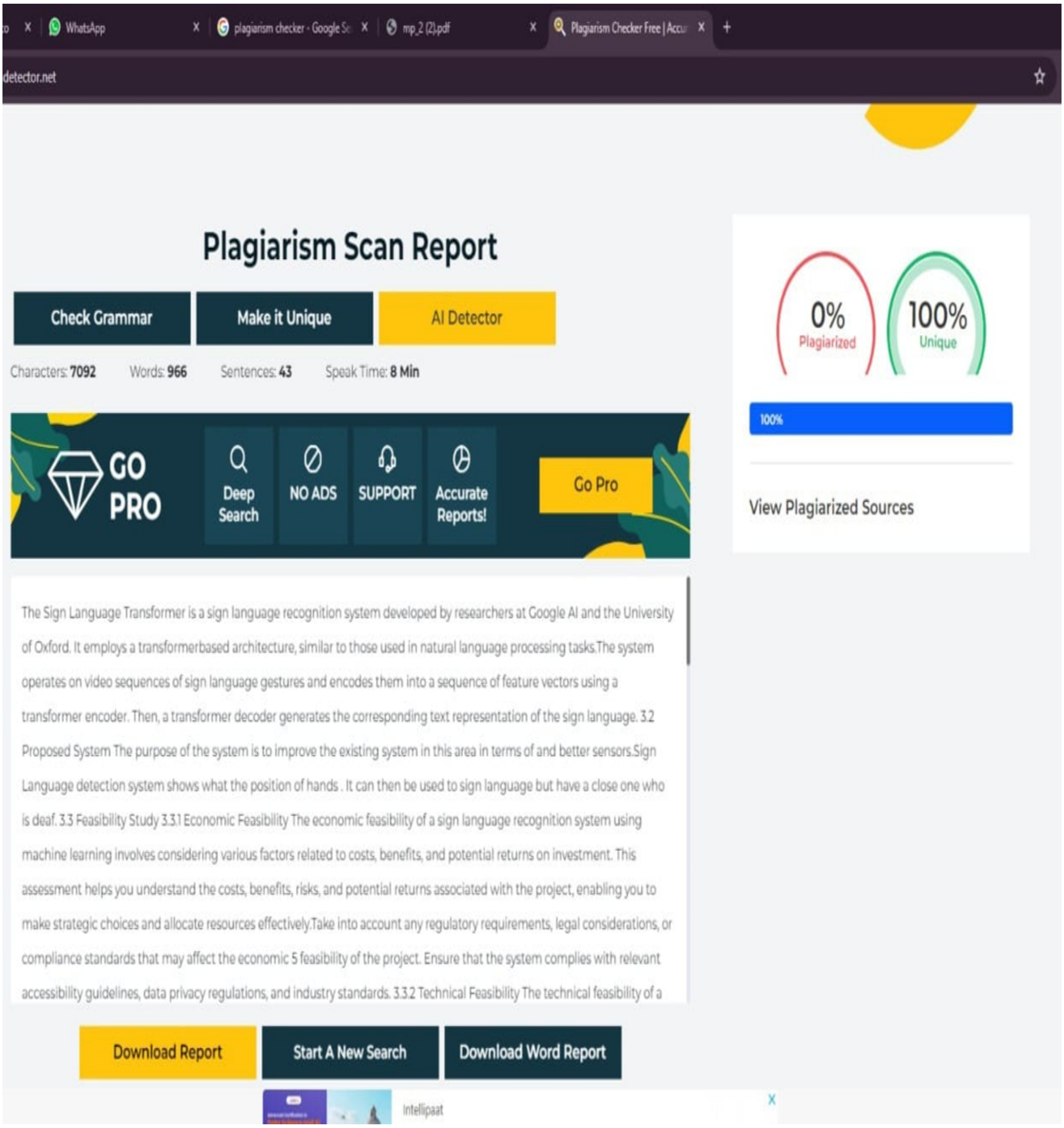

Figure 8.1: **Plagiarism Report**

Chapter 9

SOURCE CODE & POSTER PRESENTATION

9.1 Source Code

```
<!DOCTYPE html>
<html lang="en">

<head>
    <!--Sets up stylesheet and script file linking. Adds favicon and site title as well-->
    <meta charset="UTF-8">
    <link rel="shortcut icon" type="image/png" href="Images/asl_logo.ico" />
    <title>Sign Language Recognition | Sufiyaan</title>
    <link href="https://fonts.googleapis.com/css?family=Google+Sans:100,300,400,500" rel="stylesheet
        ">
    <link rel="stylesheet" href="CSS/style.css">
    <link rel="stylesheet" href="CSS/animate.css">
    <script src="dist/build.js"></script>
</head>

<body>
    <!--Welcome Screen-->
    <div id="welcomeContainer" class="animated">
        <div id="welcomeScreen">
            <h1>Welcome to Sign Language Recognition</h1>
            <h3 class="animated fadeInUp delay-2s">...</h3>
        </div>
        <button id="proceedButton">Proceed</button>
    </div>

    <!--Title bar explains the stage of the program
    (eg: instructions for Training, Prediction, and Video Chat)-->

    <!--Status bar shows the status of translation-->
    <div id="status">
        <p id="status-text">Status: Not Ready</p>
    </div>

    <!--The Translator Window displays the video of the user, various buttons, and holds
    the training and translation screens-->
```

```
    <div id="translatorWindow">

        <!--Initial Training Holder creates the screen where users train Start and Stop Gestures-->
        <div id="initialTrainingHolder">
            <img src="" alt="checkmark" id="checkmark_startButton" class="checkMark">
            <button id="startButton" class="trainButton ">Start Gesture</button>
            <button id="clear_startButton" class="clearButton">Clear</button>
            <h3 id="counter_startButton" class="counter"></h3>

            <button id="stopButton" class="trainButton ">Stop Gesture</button>
            <button id="clear_stopButton" class="clearButton">Clear</button>
            <h3 id="counter_stopButton" class="counter"></h3>
            <img src="" alt="checkmark" id="checkmark_stopButton" class="checkMark">
        </div>

        <!--Training Holder displays the screen where users add and train new gestures-->
        <div id="trainingHolder">
            <h5 id="add-gesture">Add Gesture</h5>
            <img src="Images/plus_sign.svg" alt="Plus Sign" id="plus_sign" class="plus_sign animated
                ">
            <form id="add-word" autocomplete="off">
                <input type="text" id="new-word" placeholder="New Gesture Title">
                <input type="submit" id="submit-word" value="Add Word &#9658;">
            </form>
            <button id="doneRetrain" class="doneRetrain">Done Retraining</button>

            <div id="trainingCommands"></div>
        </div>

        <!--Translation Holder is where the Gesture Card and Text that was translated is displayed
            .-->
        <div id="translationHolder">
            <div id="translatedCard">
            </div>
            <h3 id="translationText">Start Signing!</h3>
        </div>
    </div>

    <!--Trained Card Holder contains all the trained gestures' cards.
    It is outside the Training and Translation Windows because it has to be displayed on both
        screens-->
    <div id="trainedCardsHolder">
    </div>
</body>

</html>
```

9.2 Poster Presentation

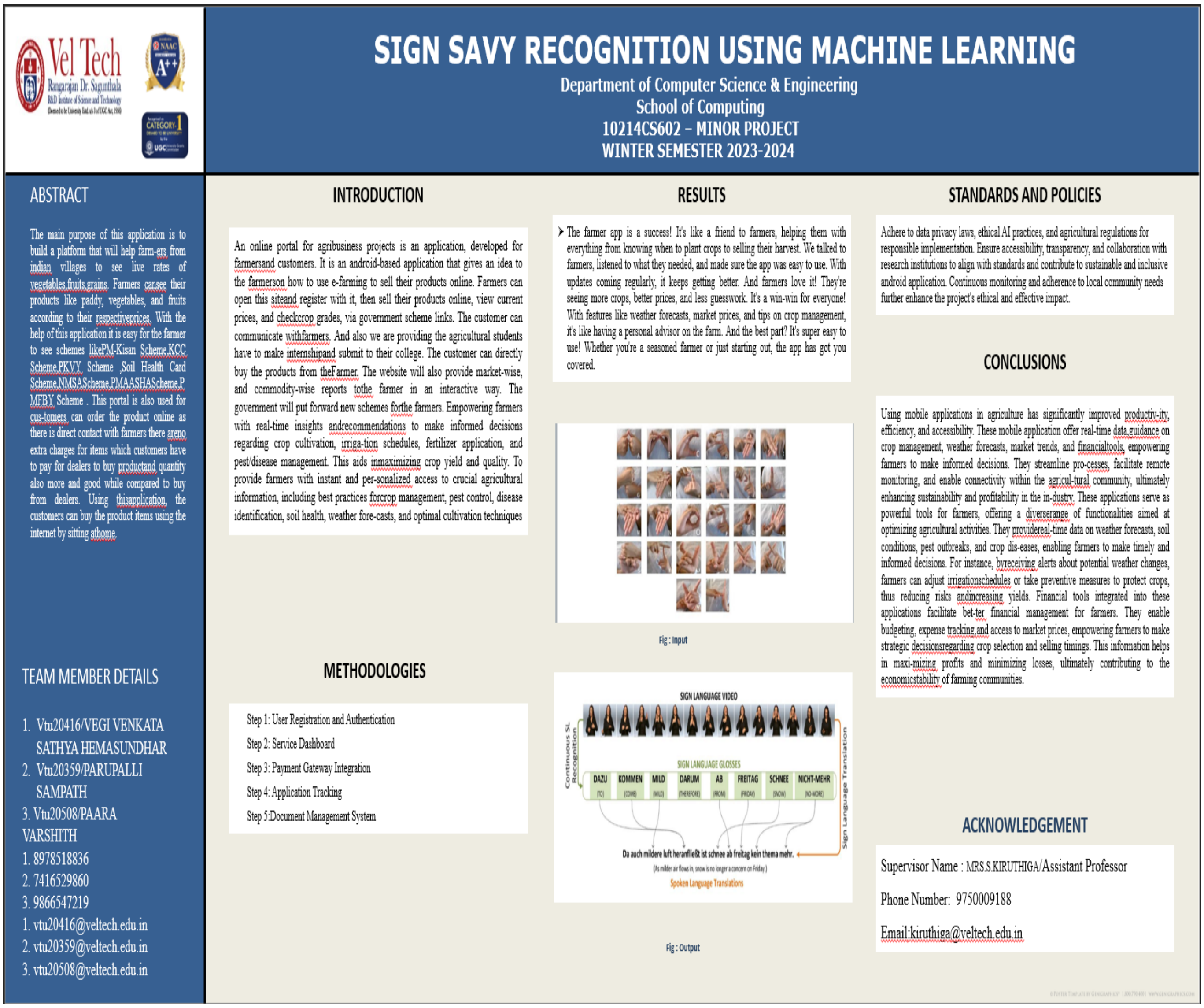

Figure 9.1: **Poster Presentation**

References

[1] Kute, Dattatray Vishnu, et al. ”Deep learning and explainable artificial intelligence techniques applied for detecting money laundering–a critical review.” IEEE access 9: 82300-82317.2021

[2] Li, Jiyang, Lin Huang, Siddharth Shah, Sean J. Jones, Yincheng Jin, Dingran Wang, Adam Russell et al. ”SignRing: Continuous American Sign Language Recognition Using IMU Rings and Virtual IMU Data.” Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies 7, no. 3: 1-29. 2023

[3] Guo, Yunqi, Jinghao Zhao, Boyan Ding, Congkai Tan, Weichong Ling, Zhaowei Tan, Jennifer Miyaki, Hongzhe Du, and Songwu Lu. ”Sign-to-911: Emergency Call Service for Sign Language Users with Assistive AR Glasses.” In Proceedings of the 29th Annual International Conference on Mobile Computing and Networking, pp. 1-15. 2023.

[4] Ahmad, Ghulab Nabi, Shafi Ullah, Abdullah Algethami, Hira Fatima, and Syed Md Humayun Akhter. ”Comparative study of optimum medical diagnosis of human heart disease using machine learning technique with and without sequential feature selection.” ieee access 10: 23808-23828. 2022

[5] Kanmani, R., S. Muthulakshmi, K. Sri Subitcha, M. Sriranjani, R. Radhapoorani, and N. Suagnya. ”Modern irrigation system using convolutional neural network.” In 2021 7th International Conference on Advanced Computing and Communication Systems (ICACCS), vol. 1, pp. 592-597. IEEE, 2021.

[6] Naveen, P., D. Saranesh, V. K. Vishnukanth, B. Sabarish, D. Vishnu, and S. R. Ashokkumar. ”Detection and recognition of species using deep convolution neural network.” In 2022 8th international conference on advanced computing and communication systems (ICACCS), vol. 1, pp. 1830-1835. 2022.

[7] Channa, Asma, Nirvana Popescu, Justyna Skibinska, and Radim Burget. ”The rise of wearable devices during the COVID-19 pandemic: A systematic review.” Sensors 21, no. 17: 5787.2021

[8] Asif, Muhammad Mugees, Sana Asif, Iqra Mubarik, and Rabia Hussain. ”Malicious applications detection in android using machine learning.” 2022.