Sign Language Detection

MINOR PROJECT
SOFTWARE REQUIREMENT SPECIFICATION
(VI Semester)

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE AND ENGINEERING

SUBMITTED BY

HIMANSHI CHAINANI (20BCZN015) – Section G
ALOK SHARMA (20BCZN031) – Section G

Project Guide: Mr. Tushar Vyas (Assistant Professor, CSE)

JECRC UNIVERSITY, JAIPUR

FEBRUARY 2023 to JUNE 2023

1. Introduction

1.1 Document Purpose

The goal of this project's software requirements specification is to clearly and precisely define the functional and non-functional requirements for sign language detecting software.

It will also give a clear picture of the project's objectives, scope, restrictions, and deliverables.

Throughout the development process, the document will aid in communication and collaboration between the developers and their mentor.

This software requirements specification (SRS) document's scope includes the creation of a sign language detection system employing computer vision techniques. The project's goal is to create an application that can recognise real-time sign language motions and convert them into text.

The project will be written in Python and will make use of libraries such as OpenCV, TensorFlow, and Keras to construct computer vision techniques.

The project will be divided into two major modules: training and detection. The deep learning model will be trained on a dataset of sign language gestures by the training module. The detection module will be in charge of detecting sign language motions in real time using the trained model.

To facilitate user engagement, the software will include a graphical user interface (GUI).
The SRS will explain the software system's functional and non-functional requirements, including the exact features that the software system must deliver as well as the performance, security, and safety criteria that it must meet.

1.2 Product Scope

The software being specified is a sign language detection system that recognises hand motions and converts them into text using convolutional neural networks (CNN). The software's goal is to help people with hearing or speech impairments communicate more effectively by providing a reliable and precise method of understanding sign language. The technology attempts to reliably and in real-time recognise a wide range of sign language motions, enabling for easy interaction between persons with and without hearing or speech problems.

This software has various advantages. To begin with, it can greatly improve the quality of life for people who have hearing or speech difficulties by allowing them to communicate more efficiently and easily. Furthermore, the software can be used in a variety of settings, such as schools, workplaces, and public places, where communication barriers can be a major issue for people with hearing or speech impairments.

The software's goals are to recognise a wide variety of sign language movements accurately, and to be user-friendly and accessible to users with varied degrees of technical expertise. The software's goals include fostering inclusion and accessibility in public settings, enhancing communication between people with hearing or speech impairments and their peers, and advancing the field of computer vision and machine learning.

The CNN design of the sign language detection system enables for accurate gesture identification. The system receives input from a camera, which captures the user's hand motions, and then processes the input using CNN to recognise the gestures and produce text.

To summarise, sign language detection software is a novel and significant system with the potential to significantly enhance the quality of life for those with hearing or speech difficulties. The programme fosters inclusion and accessibility by precisely and in real-time recognising a wide range of sign language movements, bridging the communication gap between persons with and without hearing or speech difficulties.

1.3 Intended Audience and Document Overview

The document contains thorough information on the software architecture, data flow, and technical specifications to help developers understand the project's needs as well as the technologies and tools that will be utilised to develop and deploy it.
The document will specify the features, requirements, and directions that must be included in these publications.

The project will be user-friendly, with a simple interface that is simple to navigate and understand. The document will detail the software's features and functionalities, as well as the system requirements and installation instructions, to assist users in efficiently interacting with the product.

The document is aimed at a wide range of audiences, from developers to end users, and contains information customised to each group's specific needs and requirements.

Intended Audience
This document is intended for the following audiences:

1. Instructors/Professors
Instructors and professors are an important audience for this project. They will use this document to understand the requirements and specifications for the Sign Language Detection software. Additionally, they will be interested in knowing the design and implementation details of the software, as well as the testing and validation procedures.

This will help them to provide guidance and feedback to the development team, and ensure that the project meets the desired academic standards.

2. Development Team
The development team, including the project leader and team members, are the primary audience for this document. They will use this document to understand the requirements and specifications for the Sign Language Detection software, as well as the design and implementation details.

Additionally, they will use this document to plan and coordinate the development activities, as well as to ensure that the software meets the desired quality standards.

3. Evaluators

Evaluators are responsible for evaluating the software and providing feedback on its usability, accessibility, and effectiveness. They may be members of the deaf and hard-of-hearing communities, or experts in sign language or assistive technology. They will use this document to understand the scope and objectives of the project, as well as the design and implementation details of the software. Additionally, they will use this document to provide feedback on the software's features, usability, and accessibility.

By including these audiences in the Intended Audience section, we aim to provide the necessary information and guidance for the successful development and testing of the Sign Language Detection software. We recognize the importance of each audience in contributing to the success of the project, and we are committed to ensuring that their needs and expectations are met.

1.4 Definitions, Acronyms and Abbreviations

Definitions:

Sign Language: A visual language that uses a combination of hand gestures, facial expressions, and body language to communicate ideas and concepts.

Sign Language Detection: The process of using computer vision and machine learning algorithms to identify and interpret sign language gestures and convert them into text or speech.

Computer Vision: The field of study focused on enabling computers to interpret and understand digital images or videos.

Machine Learning: A subfield of artificial intelligence that enables computers to learn from data without being explicitly programmed.

Convolutional Neural Network (CNN): A type of deep neural network commonly used in computer vision tasks, such as image recognition, that uses a hierarchical approach to learn increasingly complex features from data.

OpenCV: An open-source computer vision library that provides a set of programming functions for real-time computer vision applications.

Python: A popular programming language used for various purposes, including machine learning and computer vision.

Webcam: A camera that captures video footage in real-time, which can be used to input video data into a computer vision system for processing.

Google Colab: A cloud-based platform for running Python notebooks that allows users to access powerful computing resources, including GPUs and TPUs.

Max Pooling: A pooling operation commonly used in convolutional neural networks (CNNs) to reduce the dimensionality of feature maps and capture the most important features.

Flattening Layer: A layer in a neural network that flattens the input data into a one-dimensional vector, which can be used as input to fully connected layers.

Google Drive: A cloud-based storage platform that allows users to store and share files and documents online.

Acronyms:

ASL: American Sign Language

CNN: Convolutional Neural Network

FPS: Frames Per Second

OpenCV: Open Source Computer Vision

Abbreviations:

API: Application Programming Interface

FPS: Frames Per Second (a measure of video or image processing speed)

GUI: Graphical User Interface

SRS: Software Requirement Specification

1.5 References and Acknowledgments

We would like to express our sincere gratitude to our mentors and educators who have provided us with invaluable guidance and support throughout the development of our Sign Language Detection Minor Project.

We are grateful to Tushar Sir for his tireless efforts in helping us understand the intricacies of computer vision and deep learning. His expertise and insights have been instrumental in shaping our project and improving our skills in these fields. We would also like to thank Swikruti Ma'am for her support and encouragement, as well as her valuable feedback on our project. Her insights and guidance have helped us to better understand the nuances of sign language and refine our approach to detecting it.

We would like to extend our thanks to CampusX for their informative and engaging Deep Learning Playlist, which provided us with a solid foundation in the theory and practical applications of deep learning. We would also like to acknowledge the numerous resources available on OpenCV, including their comprehensive tutorials, GitHub repository, and the helpful tutorials and walkthroughs provided by Murtaza's Workshop, ProgrammingKnowledge, and Sentdex.

Finally, we would like to express our gratitude to our classmates, who provided us with feedback and support throughout the development of this project. Their insights and perspectives have been invaluable in helping us to identify areas for improvement and refine our approach to sign language detection.

Overall, this project would not have been possible without the support and guidance of these individuals and resources, and we are deeply grateful for their contributions to our learning and growth as aspiring computer vision and machine learning profession

2. Overall Description

2.1 Product Perspective

Context and Origin

The Sign Language Detection software is being developed to bridge the communication gap between people who are deaf or hard-of-hearing and those who are not. The software will use computer vision and deep learning techniques to recognize and translate sign language gestures into text. This will allow people who are deaf or hard-of-hearing to communicate more effectively with others, and enable greater accessibility in public spaces and events.

The software will be developed using Python programming language, along with TensorFlow and Keras libraries for machine learning and computer vision. The core technology that the software will be based on is Convolutional Neural Networks (CNNs), which have proven to be highly effective in recognizing and classifying image data.

System Architecture

The Sign Language Detection software will consist of several major components, each with its own set of subcomponents and interconnections:

1. Input Module: This module will be responsible for capturing video input from a camera or a video file. It will use OpenCV library to preprocess the input video and extract frames for further analysis.
2. Gesture Detection Module: This module will use CNNs to detect and classify sign language gestures in the input frames. It will use a pre-trained model for gesture recognition, which will be fine-tuned on a custom dataset of sign language gestures.
3. Translation Module: This module will be responsible for translating the recognized gestures into text or speech. It will use natural language processing (NLP) techniques to generate the output text or speech.
4. Output Module: This module will be responsible for displaying the output text or speech on a screen or a speaker. It will also provide an option to save the output as a text file or an audio file.

2.2 Product Functionality

The Sign Language Detection system must perform the following major functions:

- Capture live video input from a webcam or pre-recorded video input from a file.
- Detect human hands and track their movements within the video frame.
- Identify the gestures performed by the hands using a pre-trained convolutional neural network (CNN) model.
- Translate the identified gestures into corresponding text or speech output.
- Allow users to choose their preferred mode of output (text or speech).
- Allow users to pause and resume video input.
- Allow users to adjust the sensitivity of gesture detection.

The functions are organized as follows:

Input

The system must be able to capture live video input from a webcam.

Gesture Detection

The system must be able to detect human hands and track their movements within the video frame. It must also be able to identify the gestures performed by the hands using a convolutional neural network (CNN) model.

Output

The system must be able to translate the identified gestures into corresponding text output.

2.3 Users and Characteristics

There are various types of users that we anticipate will use the Sign Language Detection product. Here are some of the most relevant characteristics of each user type:

Individuals with hearing or speech impairments: These users will be the primary beneficiaries of the software. They may have varying levels of sign language proficiency, and may use the software to communicate with others in different settings, including classrooms, workplaces, and social situations.

Educators and carers: These users may use the software to facilitate communication with those who have hearing or speech difficulties. They may have varying levels of technical expertise and may utilise the software often in their work.

Sign language interpreters: These users may have extensive sign language knowledge and may utilise the product to supplement their interpretation services. A subset of the product functionalities may be used to examine or check sign language translations.

Researchers: These users may collect data on sign language usage, gesture recognition, or machine learning algorithms using the device.

Business or government organizations: Organisations in business or government: These users may utilise the product to help their communication or accessibility objectives. They may have specific requirements for security, data privacy, or adherence to accessibility regulations.

Language learners: Language learners can use the product to study sign language as a second language. They may have various levels of technological skill and educational backgrounds, which may necessitate the use of additional language learning elements or resources.

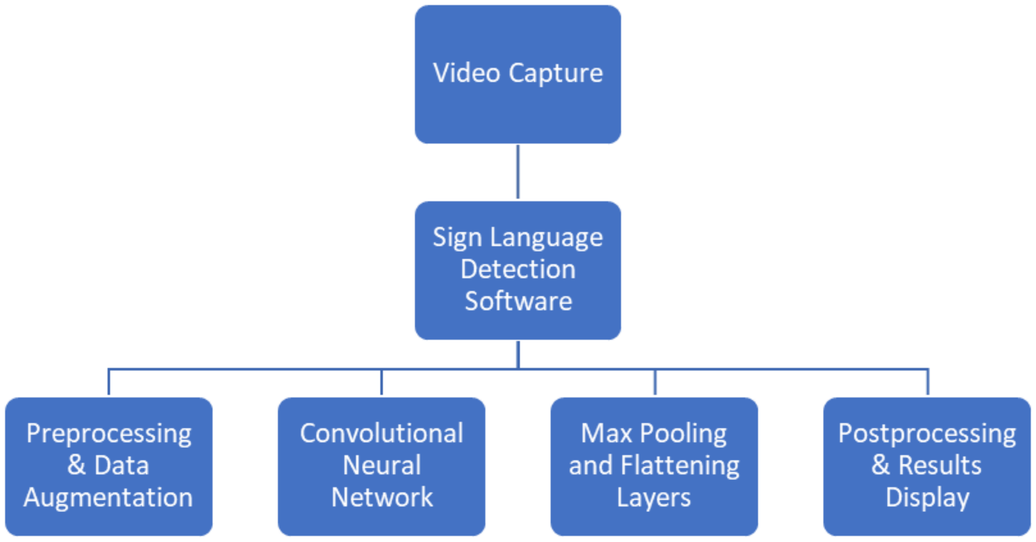
General users: These users may not have a direct need for sign language communication but may use the product out of curiosity or to learn more about sign language. They may have varied levels of technical knowledge and educational backgrounds, and their use of the product may be infrequent.

2.4 Operating Environment

The software for Sign Language Detection will run on a hardware platform that includes a computer with a modern CPU and GPU, as well as enough RAM and storage. A camera will be required by the software to gather video input of the user executing sign language movements.

The app will run on Windows 10 or later and will require Python 3.8 or higher. It will also necessitate the installation and configuration of a variety of software components and libraries, including OpenCV, TensorFlow, and Keras.

Here's a high-level diagram of the overall system's primary components and subsystem interconnections:


Video Capture
Sign Language Detection Software
Preprocessing & Data Augmentation
Convolutional Neural Network
Max Pooling and Flattening Layers
Postprocessing & Results Display

The video capture component captures video input from the user while he or she is executing sign language gestures. This input is then preprocessed and augmented to improve the quality of the data.

Following preprocessing, the data is fed into a convolutional neural network (CNN), which extracts relevant features from the data. The CNN output is then routed through max pooling and flattening layers to minimise its dimensionality.

The max pooling and flattening layers' output is then postprocessed to determine the sign language motion being done. A user interface component then displays the results to the user.

Other applications running on the same system must coexist peacefully with the Sign Language Detection software. It will have no external interfaces because it is designed to be a stand-alone application.

2.5 Design and Implementation Constraints

There are various items and difficulties that will limit the options available to sign language detection developers. Among these limitations are:

- Hardware limitations: The hardware may lack the necessary processor power or memory capacity to run the machine learning models properly. To train the models and conduct other computationally expensive operations, Google Colab, a cloud-based platform, will be employed.

- Language requirements: Python will be used to construct the software, as it is the preferred language for machine learning and deep learning applications.

- Specific technologies and databases: The software will be developed using specific machine learning and deep learning libraries such as TensorFlow and Keras. As the database used to store the sign language data is images stored on Google Drive, it will need to be accessed using Google Drive API.

- The sign language detection project's security considerations include ensuring that only authorised individuals have access to the code and that the code is secured from unauthorised modifications. The developers will use suitable version control methods, such as Git, to ensure that the code is secure and that any modifications can be tracked. Before release, the project will be thoroughly tested to discover and correct any security flaws. To guarantee that any security vulnerabilities are addressed in a timely and effective manner, the developers will also implement correct bug reporting and repair methods.

- Design conventions and programming standards: To ensure that the software is straightforward to maintain, test, and modify in the future, it must adhere to design conventions and programming standards. This includes following coding standards, employing version control, and correctly documenting the code.

2.6 User Documentation

In order to use this project, one must have access to the source code as well as the necessary software tools, such as Python and the relevant libraries. They would also require access to a computer or gadget equipped with a camera in order to collect the input video.

Once the required software and hardware are in place, the user can run the programme by typing the appropriate commands into their terminal or using an IDE such as Jupyter Notebook. The programme will then collect the input footage using the camera and process it to detect Sign Language gestures.

To run the Sign Language detection system, the following requirements must be met:

- A computer system with a minimum of 8GB RAM and 2GHz processor speed
- A webcam for capturing video input
- A stable internet connection for accessing the system on Google Colab
- The user must be able to perform Sign Language gestures in front of the webcam

2.7 Assumptions and Dependencies

Assumptions:

1. The system's performance is determined by the quality and quantity of training data utilised to train the model.
2. The Python programming language will be used to create the CNN model.
3. The model will be trained and tested using the Google Colab platform.
4. Google Drive will house the photos used for training and testing.
5. The approach expects that the video stream from the camera will be clear and properly positioned.
6. The system assumes the user has a basic knowledge of Sign Language.
7. The hardware utilised to run the system will meet the system's minimum criteria for efficient operation.
8. The system assumes the user has access to a consistent and dependable internet connection.

Dependencies:

1. Third-party Python libraries for image processing, computer vision, and machine learning, such as OpenCV and TensorFlow, will be available and compatible with the system.

2. For training and testing the model, the system will rely on the availability and stability of the Google Colab platform.
3. The system will rely on Google Drive's availability and dependability to store training and testing data.
4. For the system to work efficiently, it will be dependent on the availability and compatibility of the operating system and hardware.

3. Specific Requirements

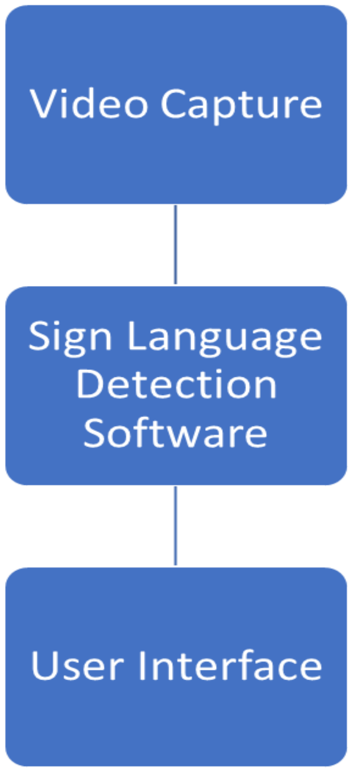
3.1 External Interface Requirements

3.1.1 User Interfaces

Capture Interface - With this interface, users would be able to collect video data from a webcam or other input device for processing. It might contain a video preview window to show the current capture, as well as a start/stop button.

Result Interface - This interface would allow users to see real-time results of sign language detection. It could include a written display of the identified indications, as well as an indicator of the system's confidence level in the detection.

The capture interface could be built to take up the majority of the screen, with a tiny area designated for the result display. The detected sign might be shown prominently in the result interface, with any extra information (such as confidence level or proposed adjustments) displayed below or to the side.



Overall, the goal of the user interface would be to provide a simple and intuitive way for users to capture sign language gestures and produce real time meaning of those gestures.

3.1.2 Hardware Interfaces

The project will use a webcam to capture video data and process it to detect sign language; the hardware interface required will be a webcam.

A USB cable is frequently used to connect webcams to computers. You can also use device camera for the same. You may need to utilise specialised libraries such as OpenCV to communicate with the webcam and capture video data.

OpenCV is a well-known open-source computer vision library that offers a wide range of functions for processing and analysing image and video data. It involves, among other things, camera calibration, object tracking, and picture filtering.

Libraries that use CNN: Several libraries that use CNN for sign language detection are available, including:

TensorFlow: TensorFlow is a Google open-source machine learning library. It comes with a number of tools for creating and training neural networks, including CNNs.

Keras is an open-source deep learning library that may be used in conjunction with TensorFlow. It offers a high-level interface for creating neural networks, such as CNNs.

These libraries can be used to train CNNs for sign language detection and process the video data captured from the webcam in real-time. They can also be used to fine-tune models for specific applications or customize the models to improve their accuracy.

3.1.3 Software Interfaces

Google Colab provides access to a cloud-based GPU for training and running CNN models for sign language identification. This eliminates the requirement to purchase dedicated hardware. Google Colab also includes a Jupyter notebook interface for writing, running, and testing code.

As for the previously listed libraries that use CNN for sign language detection (TensorFlow, Keras) within the Google Colab environment. These libraries can be installed within the Colab notebook itself using pip or conda, and then used to train and run CNN models for sign language identification.

Using Google Colab for sign language detection projects can be a quick and cost-effective approach to gain access to GPU technology and develop robust CNN models without investing in hardware.

3.2 Functional Requirements

The functional requirements of the Sign Language Detection system are divided into the following subsections:

Input Processing

1. The system shall be able to capture a video input stream from a camera.
2. The system shall be able to pre-process the input stream by extracting the frames and reducing their resolution to a standardized format.
3. The system shall be able to normalize the input stream by applying image processing techniques such as contrast normalization and histogram equalization.

Sign Detection

1. The system shall be able to detect the presence of a hand in the input stream.
2. The system shall be able to track the hand movements in real-time.
3. The system shall be able to recognize the sign language gestures made by the hand.

Output Generation

1. The system shall be able to generate text output corresponding to the recognized sign language gestures.
2. The system shall be able to display the recognized sign language gestures in real-time.

Each of these functional areas is critical to the proper functioning of the Sign Language Detection system.

4. Other Non-Functional Requirements

a. Performance Requirements

Recognition Accuracy

- The system shall be able to correctly recognize at least 95% of the signs performed by a user.
- This requirement is important to ensure that the system is reliable and can be used effectively by users who rely on accurate recognition of their signs.

Response Time

- The system shall provide a response to the user's sign within 1 second.
- This requirement is important to ensure that the system is responsive and can be used in real-time scenarios such as conversation or interpretation.

Memory Usage

- The system shall not use more than 500 MB of memory while running.
- This requirement is important to ensure that the system can run on a variety of hardware configurations without causing performance issues or crashes.
- The system shall be designed to minimize resource usage, such as CPU and memory, to ensure efficient operation and minimize costs.

Robustness

- The system shall be able to handle at least 10% noise or interference in the input video stream without a significant decrease in accuracy.
- This requirement is important to ensure that the system can handle real-world scenarios where the input video stream may not be perfect or may have some noise or interference.

These performance requirements are crucial to ensure that the Sign Language Detection project is reliable, responsive to meet the needs of its users in real-world scenarios.

b. Safety and Security Requirements

The Sign Language Detection project involves processing of personal data and may be used in various settings, such as in healthcare, education, and public environments. Therefore, it is important to ensure that the system is safe and secure for both the users and the data being processed.

The following requirements for safety have been identified:

1. All relevant safety regulations and certifications must be met by the system. This involves verifying that any hardware components are safe to use and that the software follows industry-standard safety criteria.
2. The system must not interfere with any other devices or systems in its environment.

In terms of security, the client anticipates that the product will provide a high level of security to secure user data and prevent unauthorized access. The following are the primary security requirements:

1. To ensure that only users have access to the project and data, the system must implement user authentication.
2. To resolve any security vulnerabilities and comply with industry-standard security practices, the system must be regularly updated.

c. Software Quality Attributes

In addition to the functional requirements and performance requirements, the Sign Language Detection project must also meet certain quality attributes to ensure that it satisfies the needs of its users. These quality attributes include:

Usability

- The system shall be easy to use and intuitive for users who may not have technical expertise.
- The system shall provide clear instructions for usage and error messages in understandable language.
- The system shall provide a user-friendly interface that is easy to navigate.

Maintainability

- The system shall be designed in a modular way to facilitate maintainability.
- The system shall be well-documented with clear instructions for maintenance and updates.
- The system shall have code that is easy to read and follow.

Reliability

- The system shall have a high level of reliability with minimal downtime.
- The system shall handle unexpected inputs gracefully and provide appropriate error messages.

- The system shall be tested thoroughly to ensure it performs correctly under all expected conditions.

Portability

- The system shall be able to run on a variety of platforms including Windows, and MacOS.
- The system shall be compatible with the latest versions of the major web browsers.

Performance Efficiency

- The system shall be able to process sign language gestures in real-time.
- The system shall be able to recognize a wide range of sign language gestures accurately.

Robustness

- The system must be robust and able to handle a wide range of input conditions, such as variations in lighting, background, and hand orientation.

These quality attributes will be evaluated throughout the development process to ensure that the Sign Language Detection project meets the expectations and the requirements of its users.