Sign Language Interpreter

A Project Report

Submitted in Partial fulfilment of the

Requirements for the for the award of the Degree of

BACHELORS OF SCIENCE (COMPUTER SCIENCE)

By Ayaan Amin Shaikh

Seat No – 3005605

Under the esteemed guidance of

Prof. Javed Pathan

Assistant Professor

DEPARTMENT OF COMPUTER SCIENCE

RIZVI COLLEGE OF ARTS, SCIENCE AND COMMERCE

(Affiliated to University of Mumbai)

MUMBAI-400050

MAHARASHTRA

2024-2025

RIZVI COLLEGE OF ARTS, SCIENCE AND COMMERCE

(Affiliated to University of Mumbai)

MUMBAI-MAHARASHTRA-400050

DEPARTMENT OF COMPUTER SCIENCE

CERTIFICATE

This is to certify that the project entitled, "**Sign Language Interpreter**", is benefited work of **Your Name** bearing **Seat No: 25** submitted in **Partial fulfilment of the requirements for the award of degree of BACHELOR OF SCIENCE in COMPUTER SCIENCE from University of Mumbai.**

Project Guide **HOD**

External Examiner

Date: ………….. **College Seal**

ACKNOWLEDGEMENT

I owe special thanks to the department of Computer Science of **Rizvi College of Arts Science and Commerce** for giving me a chance to prepare this project. I thank the Principal, **Dr. Ashfaq Khan** for his leadership and management. I thank the Coordinator and the Head of the Department **Professor Arif Patel** for providing us the required facilities and guidance throughout the course which culminated into the thesis. last but not the least to the project guide for this odd semester – **Professor Javed Pathan** and also to my **Parents and Friends** for supporting me throughout the semester and helping me in finalizing the project. Also deep gratitude to the **Staff and Faculty of Rizvi College of Arts Science and Commerce** for their help and support

SIGN LANGUAGE INTERPRETER USING PYTHON

RIZVI COLLEGE OF ARTS, SCIENCE AND COMMERCE
(Affiliated to University of Mumbai) MUMBAI-MAHARASHTRA – 400050

DEPARTMENT OF COMPUTER SCIENCE

DECLARATION

I, **Ayaan Amin Shaikh**, Roll No. **3005605,** hereby declare that the project synopsis entitle **“Sign Language Interpreter using Python”**, submitted for approval, **for Bachelors of Science** in Computer Science Sem VI project. For academic year **2024-25**

Signature of the Guide **Signature of the Student**

Place:

INTRODUCTION:

Sign language is manual communication commonly used by people who are deaf. Sign language is not universal; people who are deaf from different countries speak different sign languages. The gestures or symbols in sign language are organized in a linguistic way. Each individual gesture is called a sign. Each sign has three distinct parts: the handshape, the position of the hands, and the movement of the hands. Iindian Sign Language (ISL) is the most commonly used sign language in the India .

Wherever communities of people with hearing challenges or people who experience deafness exist, sign languages have developed as useful means of communication and form the core of local deaf cultures. Although signing is used primarily by the deaf and hard of hearing, it is also used by hearing individuals, such as those unable to physically speak, those who have trouble with oral language due to a disability or condition (augmentative and alternative communication), and those with deaf family members including children of deaf adults.

There is no “universal” sign language. Different countries typically have their own version of sign language, which is unique to their region and culture. For example, American Sign Language (ASL) is different from Australia’s Auslan sign language, which is different from the British Sign Language (BSL) used in the United Kingdom and Indian Sign Language (ISL) is commonly used in India . A person fluent in ASL may travel to Sydney, Australia, and have trouble understanding someone using a local version of sign language—instead of different dialects or accents apparent in oral language, the signs and gestures are different.

Today, there are more than 300 different sign languages in the world, spoken by more than 72 million deaf or hard-of-hearing people worldwide.

OBJECTIVES:

The objective of sign language is to facilitate communication for individuals who are deaf or hard of hearing. It serves as a visual language that uses hand gestures, facial expressions, and body movements to convey meaning.

1) To develop manpower for using Indian Sign Language (ISL) and teaching and conducting research in ISL, including bilingualism.
2) To promote the use of Indian Sign Language as educational mode for deaf students at primary, secondary and higher education levels.
3) To carry out research through collaboration with universities and other educational institution in India and abroad and create linguistic records/ analyses of the Indian Sign Language, including creation of Indian Sign Language corpus (Vocabulary).
4) To orient and train various groups, i.e. Govt. officials, teachers, professionals, community leaders and the public at large for understanding and using Indian Sign Language.
5) To collaborate with organizations of the deaf and other institutions in the field of disability to promote and propagate Indian Sign Language.
6) To collect information relating to Sign Language used in other parts of the world so that this input can be used to upgrade the Indian Sign Language.
7) To help bridge communication gaps between the deaf community and non-sign language users.

METHODOLOGY:

1. The project will be completely developed and modeled in the “ Spiral based model ”, as adequate amount of analyzing, developing and testing is required.

2. The “ Spiral Model ” is much more useful as we are able to test the current modules at every instance before finalizing out for the end product.

3. Firstly, the requirement analysis is conducted to get know the actual requirements and objectives of the website, such as needs of the community, determining the desired features and functionalities.

4. Further the content as well as website development work is carried out once the design and content is approved by the representative as well the developer. Leading in utilizing suitable technologies and frameworks and required libraries to create a user-friendly web Apk.
5. Even after carrying out multiple testing sessions during the development, one final testing is done ensuring the functionality, usability, and security, identifying and rectifying any bugs or issues.

6. A user-friendly content management system is also implemented to facilitate easy updating and management of the web Apk.

7. Once all components of the website are developed, tested and approved, the website will be deployed to a suitable hosting environment.

By following this methodology, the Sign Language Interpreter project can be executed systematically, resulting in the development and launch of a user-friendly, engaging web Apk that successfully translates real time gestures into text

SCOPE:

Future advancements in sign language recognition (SLR) will refine algorithms and machine learning methods for improved accuracy. Customization for regional sign languages will address communication challenges in noisy settings. Real-time systems converting sign language to text will be developed, integrating deep learning techniques. Enhanced datasets and lightweight mobile models will improve accessibility and communication for the hearing impaired

TOOLS AND TECHNOLOGIES:

1. The project consists of various different “Tools and Technologies” to get the most desirable outcome the users
2. The tools and technologies used in this project are Python, Opencv, Tensorflow and CNN Model .
3. Python and Opencv library plays a major role in the development of the frontend as well as the backend of the web Apk. It helps in the designing as well as .
4. Robflow is used for the creation of datasets on which bases our model is going to be trained

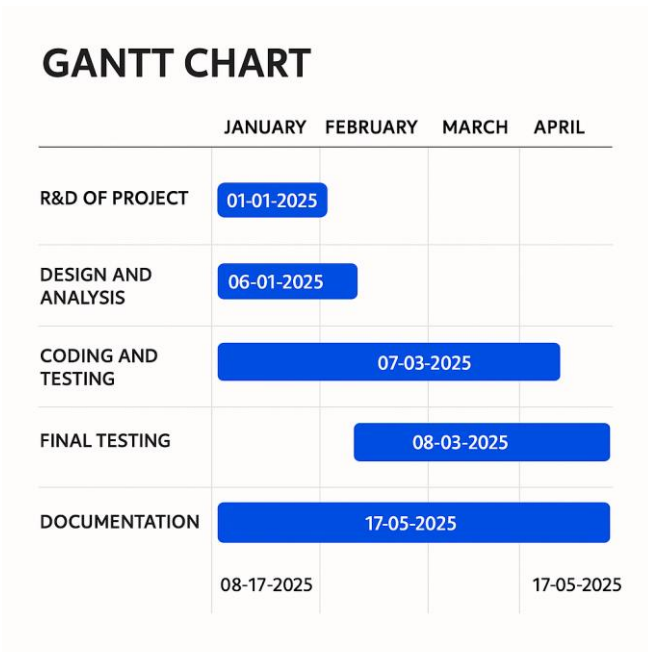
The above mentioned are the tools and technologies which are required and used by the developer. Along with it there are a few basic requirements mentioned below for both the User as well as for the developer.

1. The requirements needed for the Developers are Laptop or a Desktop machine containing some high-end specifications for ensuring the smooth running of the web Apk
2. And the requirements needed for the users are a Mobile device or desktop machine with a good network connection and browsers installed in it.

   With the mentioned tools and technologies the users as well as the developer can access the website with ease .

TIMELINE:

The timeline of the overall project can be represented with the Gantt Chart below. [2]



Tasks	Duration	Start Date	End Date
R&D of project	1 Week	01-01-2025	07-01-2025
Design and Analysis	5 Days	06-01-2025	11-01-2025
Coding and Testing	8 Weeks	12-01-2025	07-03-2025
Final Testing	1 Week	08-03-2025	14-03-2025
Documentation	10 Weeks	08-01-2025	17-03-2025

RESOURCES:

1. Human Resources:

- The developer is solely responsible for performing all necessary roles required to fulfill the human resource needs.

2. Hardware and Software:

- Computers or laptops are required for the development and testing of the website.
- Web development tools such as text editors, integrated development environments (IDEs), and content management system (CMS) software.
- Server and hosting infrastructure to support the website, including domain registration and hosting services.

3. Content and Information:

- Proper, training of the models to get 100% accurate outputs

4. Collaborative Inputs:

- Financial resources allocated for web development, design, content creation, hosting, domain registration, and ongoing maintenance.

5. Time and Timelines:

- Sufficient time should be allocated for requirement gathering, design, development, testing, content creation, and the launch of the website.
- Timelines and milestones need to be established to ensure the timely completion of the project.

By ensuring access to these necessary resources, the Sign Language Recognition project can be effectively be executed, resulting in the development of a functional, useful, and user-friendly interface through which the user can interact with specially disabled people .

EXPECTED OUTCOMES:

- **Improved Communication Accessibility:**

  Develop systems that can accurately and quickly translate sign language into text or speech (and vice-versa).

- **Real-time Translation:**

  Aim for near-instantaneous translation to minimize delays and enhance natural communication flow.

- **Enhanced Accuracy:**

  Refine algorithms and machine learning models to improve the recognition and interpretation of sign language gestures, facial expressions, and subtle movements.

- **Broader Applications:**

  Explore uses beyond basic translation, such as creating assistive tools for education, emergency services, and social interactions.

- **Personalized Solutions:**

  Develop systems that can adapt to different sign languages and regional variations.

- **Increased Independence and Participation:**

  Enable sign language users to participate more fully in social, economic, and political life.

- **Reduced Communication Barriers:**

  Minimize the challenges faced by sign language users in various settings.

- **Develop a Universal Sign Language:**
  The study also focuses on identifying common signs used worldwide, with the goal of establishing a universal sign language.

- **Bridging Communication Gaps:**

  Ensure effective communication between sign language users and non-signers in various settings.

- **Providing Access to Information and Services:**

  Facilitate access to education, healthcare, legal proceedings, and other essential services for sign language users.

- **Promoting Inclusion and Understanding:**

  Foster a more inclusive society by promoting awareness and understanding of sign language and Deaf culture.

- **Supporting Diverse Communities:**

  Address the needs of diverse sign language communities by providing access to qualified interpreters.

- **Improving Quality of Life:**

  Enhance the quality of life for sign language users by enabling them to participate fully in society.

- **Training and Professional Development:**

  Develop training programs and resources for sign language interpreters to ensure they are equipped with the skills and knowledge needed to effectively perform their duties.

- **Advocacy and Awareness:**
  Advocate for the rights and needs of sign language users and promote awareness of sign language and Deaf culture.

ADVANTAGES & LIMITATIONS:

Some Advantages of sign language interpreter are as follows.

Advantages of Signed Languages

1. **Accessibility for Deaf and Hard of Hearing Individuals:**
   - Signed languages provide a natural and effective means of communication for those who are deaf or hard of hearing, allowing them to express themselves fully.
2. **Visual Communication:**
   - Signed languages utilize visual-spatial modalities, which can enhance understanding through gestures, facial expressions, and body language.
3. **Cultural Identity:**
   - Signed languages are often deeply tied to the cultural identity of the Deaf community, fostering a sense of belonging and shared experience.
4. **Immediate Communication:**
   - In environments where sound is not practical (e.g., loud settings or quiet spaces), signed languages allow for effective communication without the need for vocalization.
5. **Expressiveness:**
   - The use of facial expressions and body movements can add layers of meaning, making communication rich and nuanced.

Limitations of Sign Language

1. **Limited Reach:**
   - Signed languages may not be understood by hearing individuals who do not know the language, potentially leading to communication barriers.
2. **Variability:**
   - Different regions may have different signed languages or dialects, which can create challenges for communication across different communities.
3. **Learning Curve:**
   - For those who are not familiar with signed languages, there can be a steep learning curve, requiring time and practice to become proficient.
4. **Limited Resources:**
   - In some areas, there may be fewer educational resources, interpreters, or materials available for learning signed languages compared to spoken languages.
5. **Misconceptions:**
   - There are often misconceptions about signed languages being simply gestures or miming, which can undermine their complexity and richness as fully developed languages.

REFERENCES:

1. https://youtu.be/-UXYAAqm9fE?si=aY6fXbnCDY6-EYib - R&D

2. https://youtu.be/zC22yPmc6Kw?si=or5Ibzw5H9_dBAbd - Gantt chart

3. https://chatgpt.com/share/67d6a709-aab4-8006-8091-d93aa206272b - Methodology

4. Project Introductions: What They Are and How To Write Them | Indeed.com - Introduction

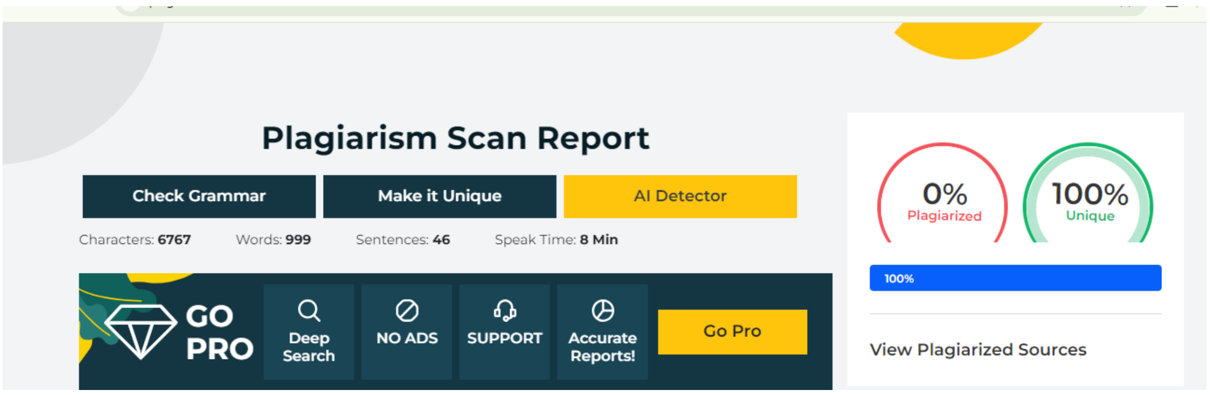
5. How to Write An effective Project Objective, With Examples [2024] • Asana - Objectives

6. https://www.quora.com/What-are-the-advantages-and-disadvantages-of-a-signed-language - Advantages & limitations

PLAGIARISM REPORT [22]

A plagiarism report is a document or a summary that provides information about the presence of plagiarism in a piece of written or academic work. Plagiarism refers to the act of using someone else's words, ideas, or work without proper attribution or permission, presenting them as your own. Plagiarism is considered unethical and can have serious consequences, particularly in academic and professional settings.

A plagiarism report is typically generated by plagiarism detection software or services. It scans a given document or text for similarities to existing sources, such as published articles, books, websites, and other written material. When the software identifies matching or highly similar content, it highlights or marks the specific passages that may be considered plagiarized.



Plagiarism Scan Report

Characters: 6767 Words: 999 Sentences: 46 Speak Time: 8 Min

0% Plagiarized

100% Unique

Plagiarism check for Sign Language Interpreter

DECLARATION

I hereby declare that the project entitled, **“Sign Language Interpreter”** done at **Rizvi College of Arts, Science and Commerce**, has not been in any case duplicated to submit to any other university for the award of any degree. To the best of my knowledge other than me, no one has submitted to any other university.

The project is done in partial fulfilment of the requirements for the award of degree of **BACHELOR OF SCIENCE (COMPUTER SCIENCE)** to be submitted as final semester project as part of our curriculum.

Ayaan Amin Shaikh

TABLE OF CONTENTS

CHAPTER 1. INTRODUCTION

1.1 Introduction to the Web-App [4]

Sign languages (also known as signed languages) are languages that use the visual-manual modality to convey meaning, instead of spoken words. Sign languages are expressed through manual articulation in combination with non-manual markers. Sign languages are full-fledged natural languages with their own grammar and lexicon. Sign languages are not universal and are usually not mutually intelligible, although there are similarities among different sign languages.

Linguists consider both spoken and signed communication to be types of natural language, meaning that both emerged through an abstract, protracted aging process and evolved over time without meticulous planning. This is supported by the fact that there is substantial overlap between the neural substrates of sign and spoken language processing, despite the obvious differences in modality.

Sign language should not be confused with body language, a type of nonverbal communication. Linguists also distinguish natural sign languages from other systems that are precursors to them or obtained from them, such as constructed manual codes for spoken languages, home sign, "baby sign", and signs learned by non-human primates.

Wherever communities of people with hearing challenges or people who experience deafness exist, sign languages have developed as useful means of communication and form the core of local deaf cultures. Although signing is used primarily by the deaf and hard of hearing, it is also used by hearing individuals, such as those unable to physically speak, those who have trouble with oral language due to a disability or condition (augmentative and alternative communication), and those with deaf family members including children of deaf adults.

1.2 Aim

The aim of this **Indian Sign Language (ISL) recognition project** is to develop an accurate and efficient system capable of detecting and classifying hand gestures, distinguishing between numbers and alphabets. This project seeks to improve accessibility by assisting individuals with speech or hearing impairments in communicating effectively through sign language. To achieve high accuracy, the system will utilize machine learning techniques, specifically training a model on grayscale images to enhance palm gesture recognition.

1.3 Objectives [5]

The objective of this Indian Sign Language (ISL) recognition project is to develop an accurate and efficient system that can detect and classify palm gestures with high precision. The project aims to enhance communication for individuals with hearing and speech impairments by providing a tool that facilitates seamless interaction between sign language users and those unfamiliar with it. To improve accuracy, the system will utilize grayscale image processing techniques for effective palm gesture detection. Additionally, real-time detection will be implemented to ensure smooth and responsive gesture recognition. A user-friendly interface will be designed, allowing users to easily upload images and perform live detection..

1.4 Goal

The goal of the Toy Store web application is to become the leading online destination for purchasing toys by offering a diverse range of high-quality products, a seamless shopping experience, and exceptional customer service. Make it easy for customers to find, compare, and purchase toys that suit their needs and preferences. Provide a user-friendly interface, secure payment options, and fast delivery to ensure a hassle-free shopping experience. Offer toys that foster creativity, education, and development in children. Grow the customer base by reaching new audiences through effective marketing and a strong online presence. Ultimately, the goal is to bring joy to children and convenience to parents by being the go-to platform for all toyrelated purchases.

1.5 Problem Definition

Indian Sign Language (ISL) serves as a crucial mode of communication for individuals with hearing or speech impairments. However, the lack of widespread understanding of ISL creates communication barriers, making it difficult for non-signers to interact effectively with sign language users. This project aims to develop an ISL recognition system that accurately detects palm gestures and classifies them as either numbers or alphabets. The system will utilize grayscaling techniques for preprocessing, ensuring enhanced accuracy in gesture detection. A machine learning or deep learning model will be trained to recognize and classify gestures despite challenges such as variations in lighting, background noise, and hand positioning. Additionally, the project will feature a user-friendly interface that allows users to either upload images or use live detection for real-time gesture recognition.

CHAPTER 2. REQUIRED SPECIFICATION

2.1 Introduction

The Sign Language Interpreter , developed using python and some of its libraries like Opencv, Tensorflow and web technologies, is a sophisticated application designed . This platform serves as a powerful tool that empowers users to compose, modify, and generate music through an interactive interface, providing a unique blend of creativity and technology.

2.2 System Environment

The system environment for the Toy Store Web App is a critical aspect of its design and operation. It encompasses the hardware, software, network infrastructure, and other components that collectively enable the app to function effectively

2.3 Software Requirements

- Web Browser **x**
- Xampp
- Apache server

2.4 Hardware Requirements

- Processor (CPU): Intel Core , i5 , i7 seventh generation or above with integrated graphics card Operating System: Windows , Linux , Macos on any operating system.
- Memory : Minimum 4 GB RAM
- Storage: Minimum 128 GB internal storage drive
- Monitor/Display: 14” LCD monitor , resolution of 1600 x 900 for better view.

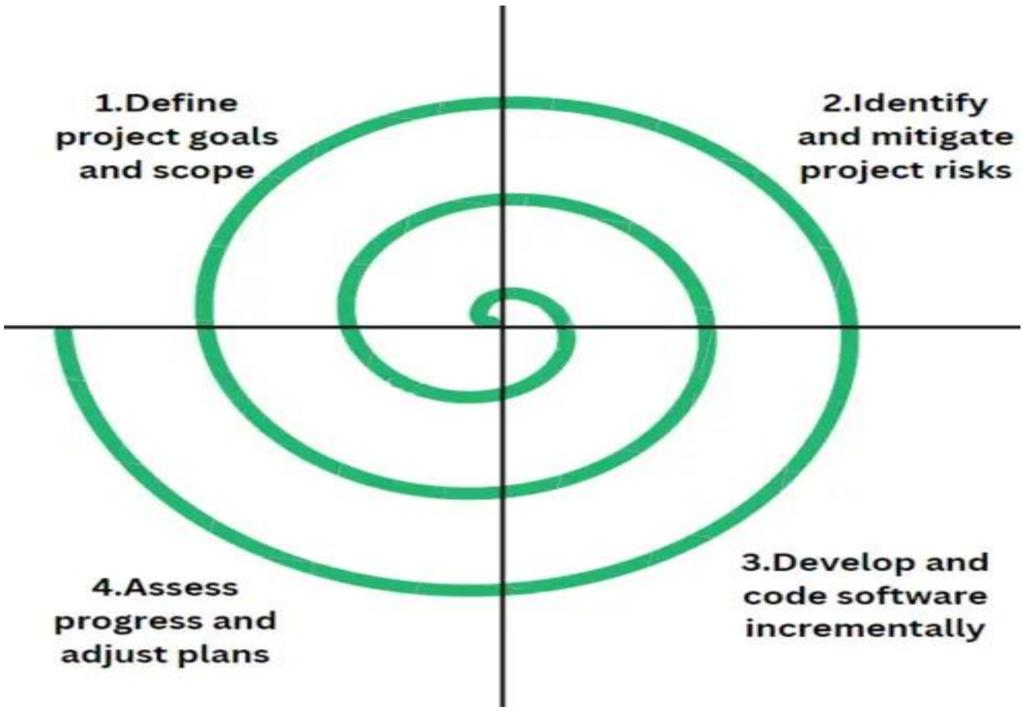
2.5 Methodology [3]

The development of Sign Language Interpreter will follow the Spiral Model, a flexible and iterative software development methodology that aligns well with the complex and evolving nature of law enforcement needs. The Spiral Model comprises multiple iterative cycles, each consisting of four main phases: Planning, Risk Analysis, Engineering, and Evaluation. In the Planning phase,we will define the project's objectives, requirements, and constraints, establishing the groundwork for subsequent iterations. The Risk Analysis phase will assess potential risks and uncertainties specific to law enforcement, such as data security and regulatory compliance, allowing us to devise risk mitigation strategies.The Engineering phase will encompass the actual development of the web app, incorporating feedback and enhancements from each iteration. Finally, the Evaluation phase will involve thorough testing, user feedback collection,. These iterative cycles will allow us to progressively refine the Toy Store Web App, ensuring that it meets the evolving needs of all users ,while maintaining a strong focus on security, functionality, and user satisfaction.

2.5 Spiral Model [20]

The Spiral Model is a software development methodology that combines elements of both iterative and waterfall models in a structured, flexible approach. It is characterized by a series of iterative cycles, or "spirals," each of which consists of four primary phases: Planning, Risk Analysis, Engineering, and Evaluation. During the Planning phase, project objectives, requirements, and constraints are defined. The Risk Analysis phase involves identifying potential risks and uncertainties, allowing for the development of mitigation strategies. The Engineering phase encompasses actual development, testing, and feedback collection, while the Evaluation phase evaluates the product's performance and progress toward meeting project goals. The Spiral Model's iterative nature allows for ongoing refinement and adaptation as the project progresses, making it particularly suited for complex projects with evolving requirements and a focus on managing risks effectively.

Diagram of Spiral Model :


1.Define project goals and scope
2.Identify and mitigate project risks
3.Develop and code software incrementally
4.Assess progress and adjust plans

Why Spiral Model?

The Spiral Model is chosen for software development projects because of its inherent ability to manage risks effectively while accommodating changing requirements. Its iterative approach allows for the early detection and mitigation of potential issues, reducing the likelihood of costly project failures. This model is particularly well-suited for complex projects and industries like law enforcement where evolving needs and strict compliance requirements demand a flexible and adaptive development process, ensuring that the end product aligns closely with the organization's evolving objectives and challenges.

Applications of the Spiral Model :

The Spiral Model is widely used in the software industry as it is in sync with the natural development process of any product, i.e., learning with maturity which involves minimum risk for the customer as well as the development firms. A few pointers of the spiral model are as follows:

1. Ideal for managing evolving requirements in dynamic software projects.
2. Well-suited for complex systems development where risk mitigation is critical.
3. Effective in industries with strict compliance and regulatory requirements.
4. Enables iterative refinement of prototypes and proof-of-concept applications.
5. Valuable for research and development projects involving experimentation.
6. Facilitates client-centric projects that require active user involvement and feedback.

CHAPTER 3. SYSTEM ANALYSIS

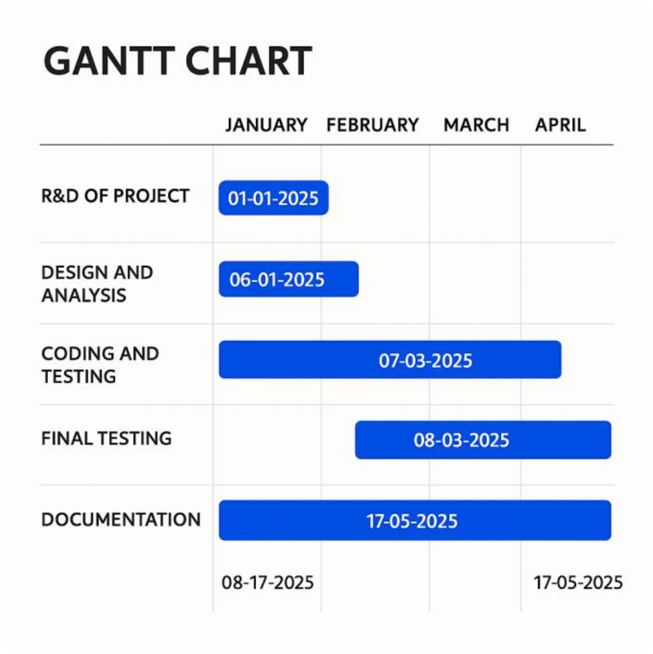
3.1 System Analysis [12]

System analysis is a structured process used to study, understand, and evaluate complex systems or processes. It involves breaking down a system into its components, examining how those components interact, and assessing their efficiency and effectiveness in achieving specific objectives.. Through techniques like data gathering, modeling, and evaluation, system analysis aims to identify problems, opportunities for improvement, and optimal solutions, ensuring that systems are designed, maintained, and upgraded to meet desired goals efficiently. The primary goal of system analysis is to bridge the gap between existing system deficiencies and desired outcomes by detecting proper words to get perfect accuracy .

3.2 Analysis of the Proposed System

- The system planning started to strategize for evolving the ideas and successfully getting a clear picture of how the new system will provide the functionalities and implement it.
- Firstly, we decided the tools and technologies on which the system will work. And then the functioning of the modules.
- Secondly, considering the limitations in the existing system, a proper GPU is required for better results .
- Allowing the user to upload and detect the image .
- Perform live detection or prediction using the webcam .

3.3 Gantt chart [2]



Tasks	Duration	Start Date	End Date
R&D of project	1 Week	01-01-2025	07-01-2025
Design and Analysis	5 Days	06-01-2025	11-01-2025
Coding and Testing	8 Weeks	12-01-2025	07-03-2025
Final Testing	1 Week	08-03-2025	14-03-2025
Documentation	10 Weeks	08-01-2025	17-03-2025

CHAPTER 4. SURVEY OF TECHNOLOGY

4.1 Python [7]

Python is a versatile and popular programming language created by Guido van Rossum and released in 1991. It is widely used in web development (server-side), software development, mathematics, and system scripting. One of Python's major strengths is its cross-platform compatibility—it runs on Windows, Mac, Linux, Raspberry Pi, and more. Python has a simple, English-like syntax that allows developers to write programs more efficiently, often with fewer lines of code compared to other languages. It operates on an interpreter system, enabling quick prototyping and immediate execution of code. Python supports multiple programming paradigms, including procedural, object-oriented, and functional programming.
In summary, PHP is a powerful and versatile scripting language for web development. It offers a broad range of applications, from creating simple, static web pages to complex web applications. Its open-source nature, extensive community support, and compatibility with various databases make it an enduring and valuable technology in the realm of web development.

4.2 TensorFlow [14]

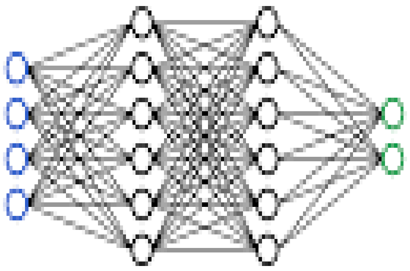
TensorFlow is a powerful platform that simplifies machine learning for both beginners and experts, offering tools to build models for desktop, mobile, web, and cloud. Its tutorials, available as Jupyter notebooks, run directly in Google Colab, requiring no setup—just click the "Run in Google Colab" button to get started. TensorFlow provides robust data tools to help you consolidate, clean, and preprocess data efficiently, with access to standard datasets for training and validation. Built on the Core framework, TensorFlow’s ecosystem supports distributed training, rapid model iteration, and debugging through Keras. You can track and analyze your model’s performance using tools like TensorBoard and Model Analysis. To accelerate development, TensorFlow Hub and the Model Garden offer collections of pre-trained models and implementations of cutting-edge research. The platform also includes preprocessing layers, scalable data pipelines, and responsible AI tools that help ensure fairness by identifying and addressing bias in your data.

4.3 Kereas [18]

Data parallelism involves replicating a single model across multiple devices or machines, with each replica processing different batches of data in parallel and later merging their results. There are various approaches to data parallelism, differing in how model replicas synchronize and combine results—some stay in sync after each batch (synchronous), while others are more loosely coupled (asynchronous). In contrast, model parallelism splits a single model across multiple devices, allowing different components of the model to process the same batch together. This method is most effective for models with naturally parallel architectures, such as those with multiple branches.

This guide specifically focuses on synchronous data parallelism, where all replicas remain synchronized after each batch, ensuring consistent model convergence similar to single-device training. It explains how to use PyTorch's `DistributedDataParallel` (DDP) module to train Keras models with minimal code modifications across multiple GPUs (typically 2 to 16) on a single machine—an ideal setup for researchers and small-scale industrial applications.

.



4.4 Opencv [12]

OpenCV (Open Source Computer Vision Library) was started at Intel in 1999 by Gary Bradsky, with its first release in 2000. Vadim Pisarevsky later joined to help lead the project. In 2005, OpenCV gained recognition when it was used in Stanley, the self-driving car that won the DARPA Grand Challenge. The development of OpenCV continued with support from Willow Garage, and today, it includes many algorithms for computer vision and machine learning. OpenCV supports multiple programming languages like C++, Python, and Java, and works across platforms such as Windows, Linux, macOS, Android, and iOS. It also includes support for GPU acceleration using CUDA and OpenCL.

OpenCV-Python is the Python version of OpenCV, combining the power of OpenCV's C++ functions with the simplicity of Python. Python, created by Guido van Rossum, is popular for its easy-to-read and concise code. Although Python is slower than C++, it can run fast by using C++ extensions in the background. OpenCV-Python uses NumPy, a library for efficient numerical operations, which makes it easy to handle image data and integrate with other scientific libraries like SciPy and Matplotlib.

OpenCV

4.5 Pandas [13]

Pandas is a powerful Python library created by Wes McKinney in 2008 for working with data sets. It helps in analyzing, cleaning, exploring, and manipulating data efficiently. The name "Pandas" comes from "Panel Data" and "Python Data Analysis". It is widely used in data science to handle large data sets, clean messy or irrelevant data, and make it meaningful. With Pandas, you can perform tasks like finding correlations between columns, calculating averages, maximum and minimum values, and removing empty or incorrect data to ensure accuracy in analysis.

Pandas makes data handling simple and intuitive by using DataFrames, which are like tables with rows and columns. It allows you to load data from various file formats such as CSV, Excel, or SQL databases, and then process it easily. With just a few lines of code, you can filter specific data, sort values, group information, and visualize trends—making it an essential tool for data analysis and decision-making.

CHAPTER 5. SYSTEM DESIGN

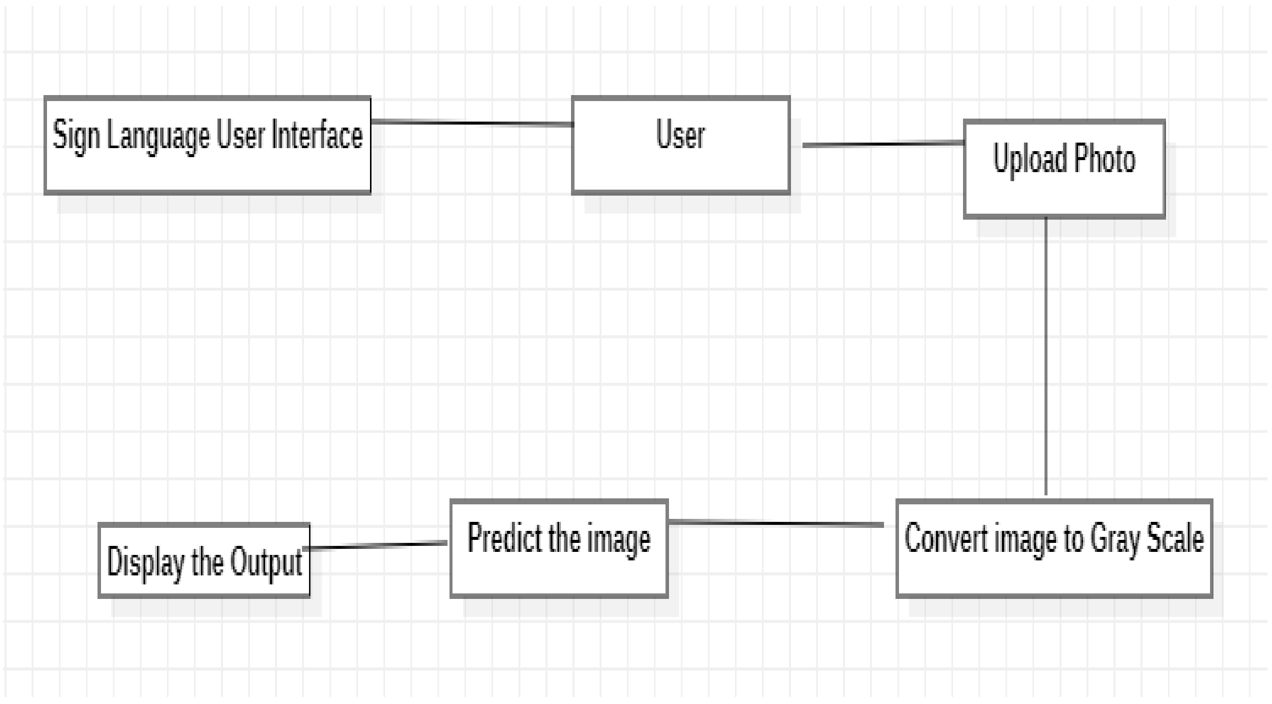
5.1 Introduction [9]

In the dynamic landscape of modern technology and assistive communication, the system design of our web app serves as a critical and strategic phase in our mission to transform the way sign language is interpreted and understood. The structured design of our Sign Language Interpreter Web App reflects not only the result of thoughtful planning but also acts as the bridge between our innovative vision and its real-world application.

As we explore the complexities of this system design, this documentation provides a detailed overview of the core components, infrastructure, data processing, and security measures that form the foundation of the Sign Language Interpreter Web App. It is designed to ensure accuracy, accessibility, and real-time interaction, making communication more inclusive for the Deaf and Hard of Hearing community.

5.2 System Architecture Diagram

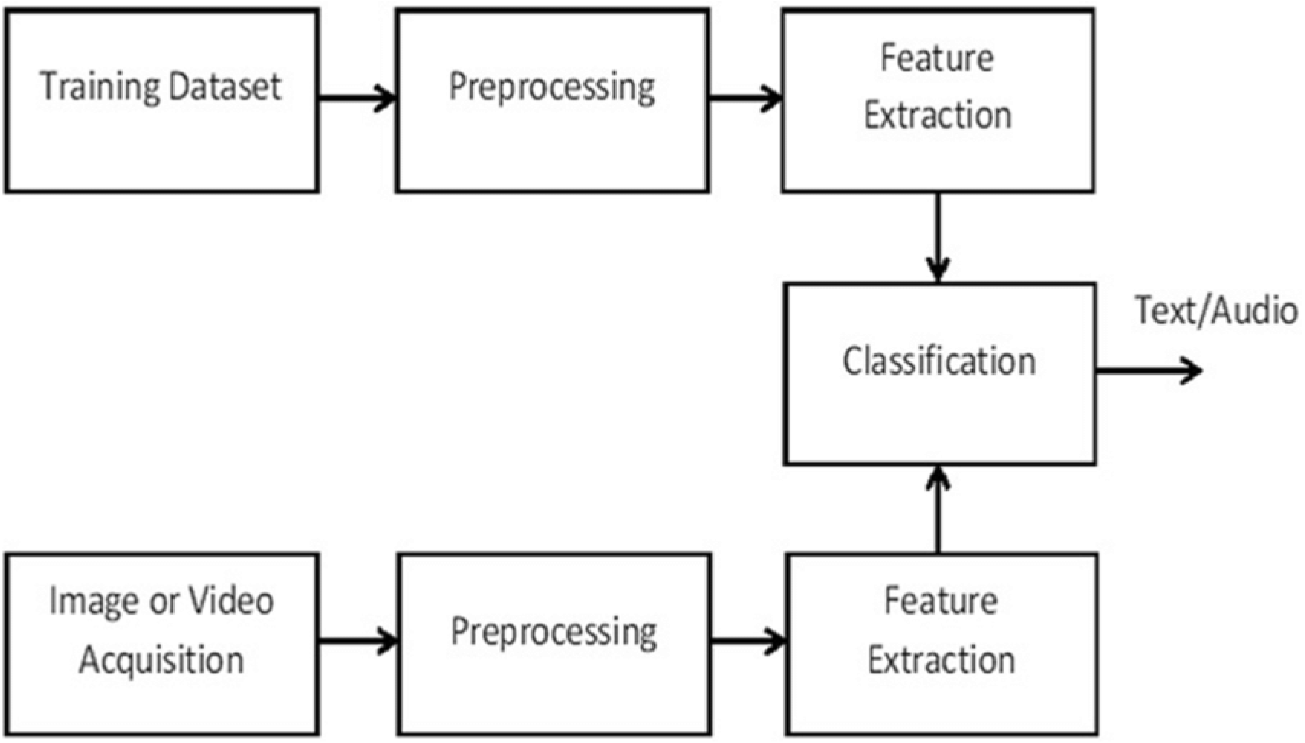
The system architecture diagram is a visual representation of the system architecture. It shows the connections between the various components of the system and indicates what functions each component performs. The general system representation shows the major functions of the system and the relationships between the various system components.



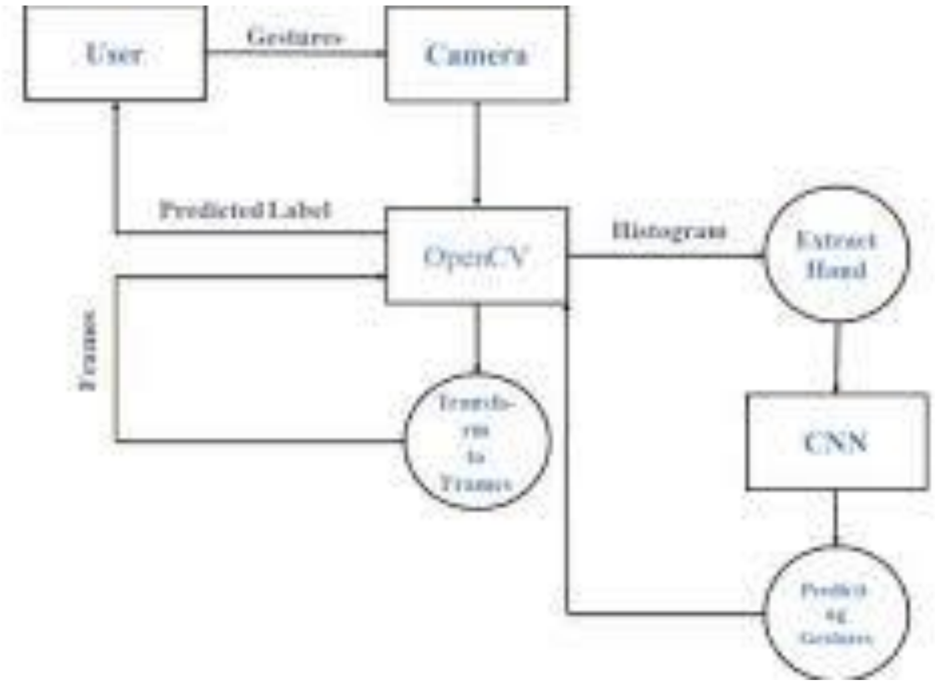
5.3 Data Flow Diagram

Data flow diagrams are used to graphically represent the flow of data in a business information system. DFD describes the processes that are involved in a system to transfer data from the input to the file storage and reports generation.

Data Flow Daigram Level – 0

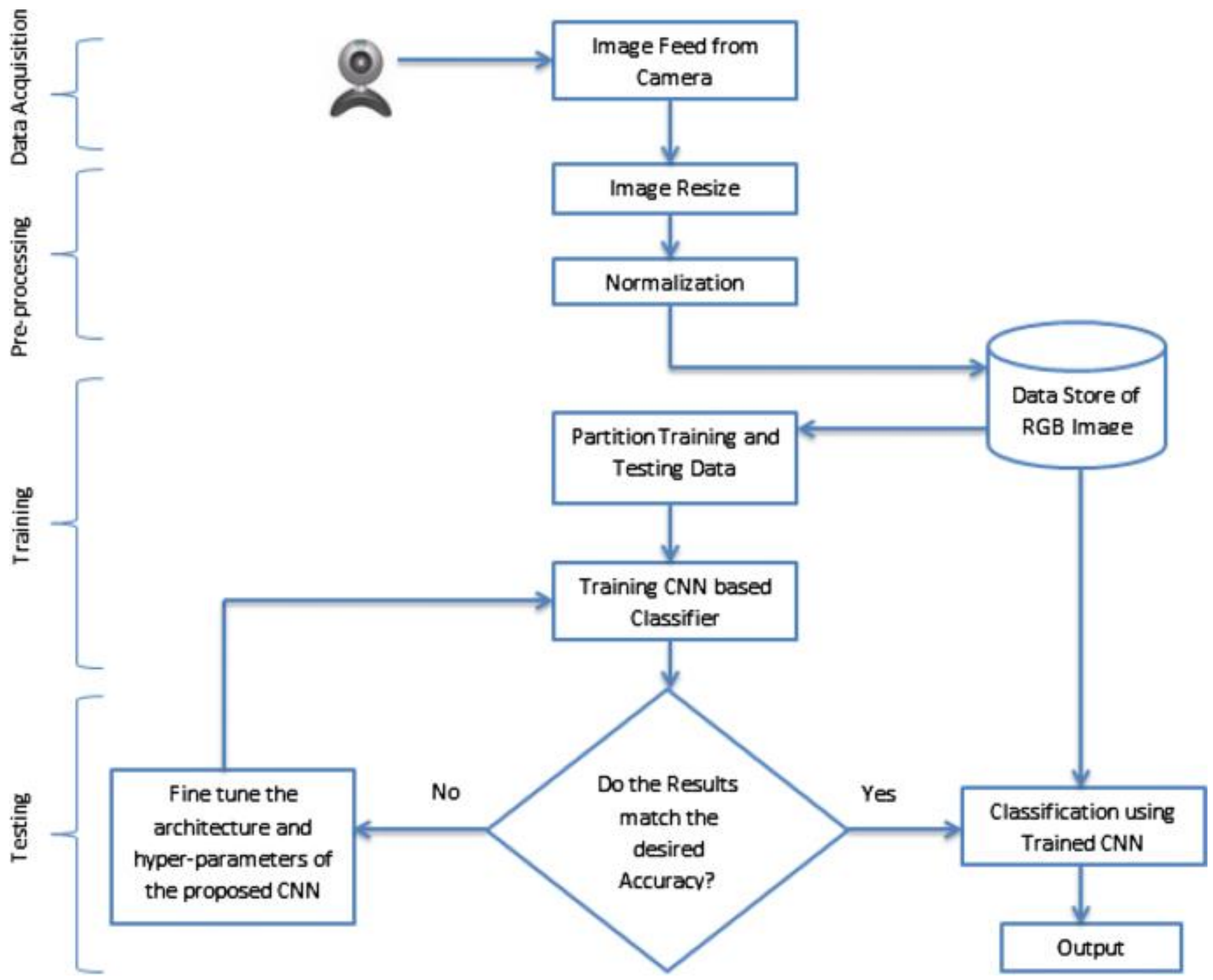


Data Flow Daigram Level



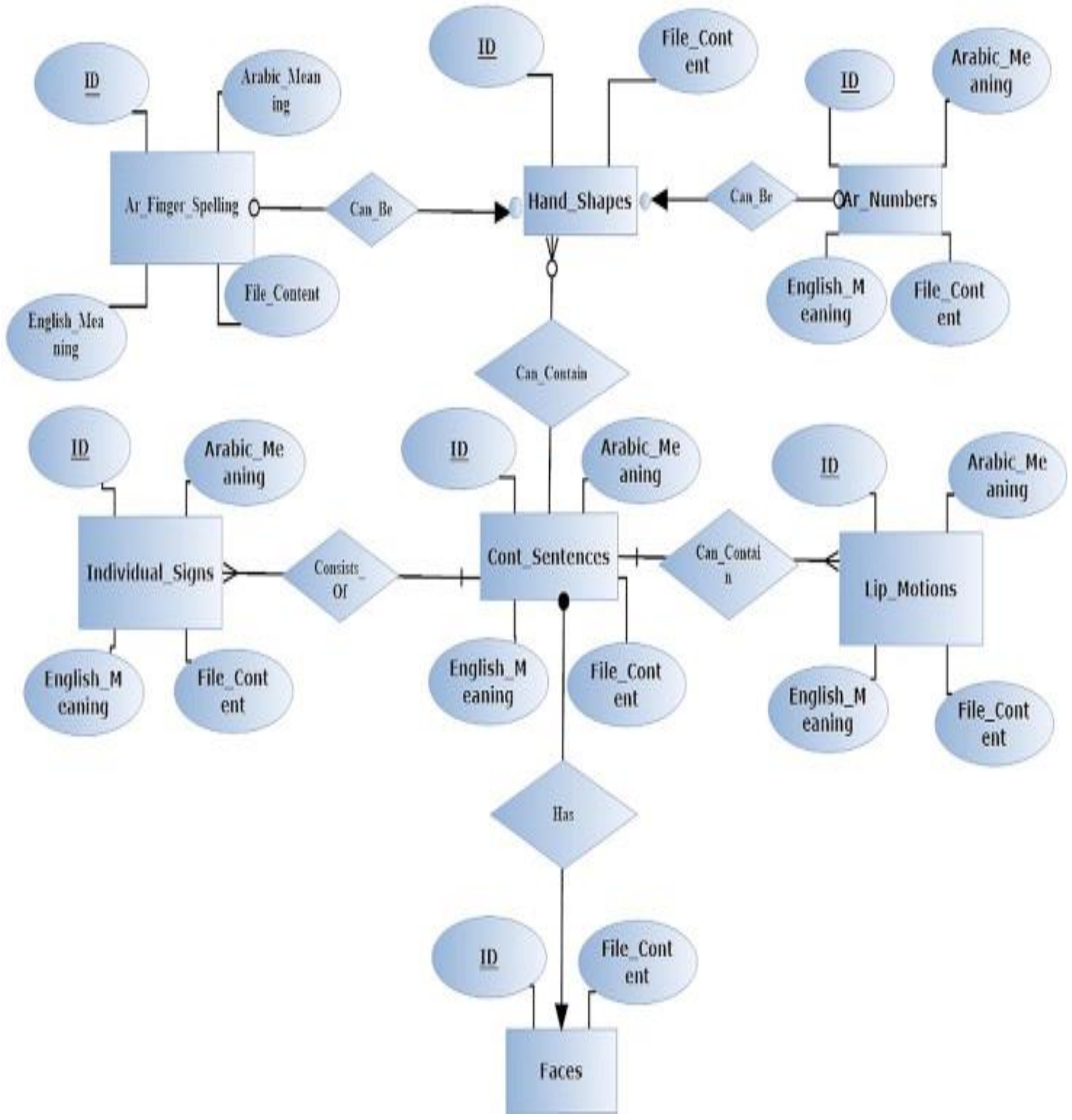
5.4 Activity Diagram [15]

An activity diagram is a type of UML (Unified Modeling Language) diagram used to visualize and model the flow of activities, actions, and decisions within a system, process, or workflow. It helps in depicting the sequential and parallel activities, along with decision points and control flows, making it a valuable tool for understanding, documenting, and improving processes or systems in various domains, such as software development, business processes, and project management.


Data Acquisition
Pre-processing
Training
Testing
Image Feed from Camera
Image Resize
Normalization
Data Store of RGB Image
Partition Training and Testing Data
Training CNN based Classifier
Do the Results match the desired Accuracy?
No
Yes
Fine tune the architecture and hyper-parameters of the proposed CNN
Classification using Trained CNN
Output

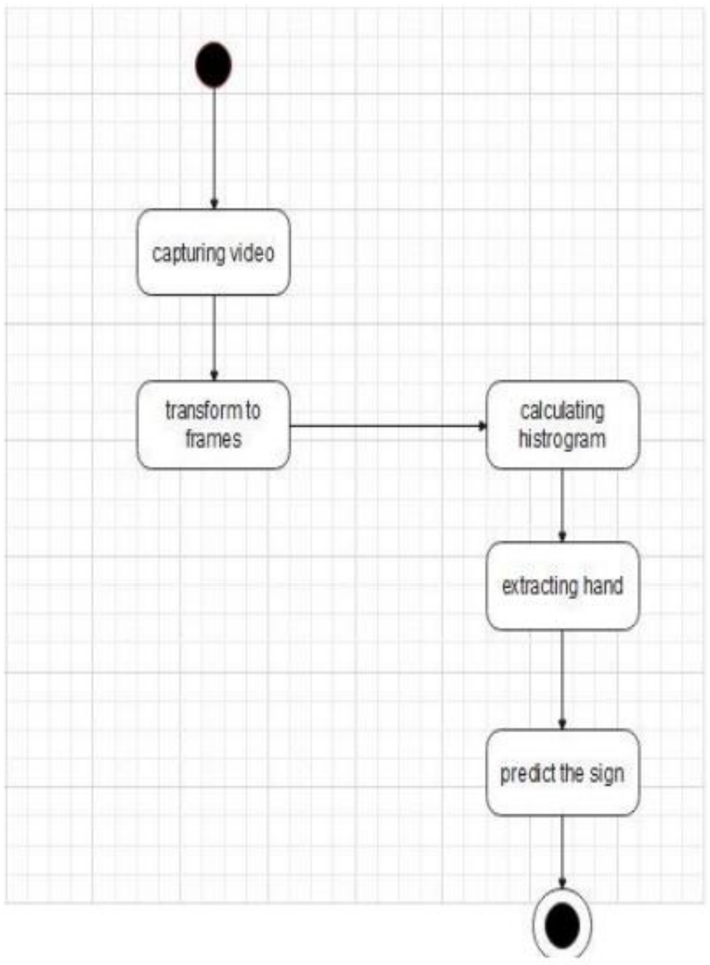
5.5 E-R Diagram [16]

An ER diagram shows the relationship among entity sets. An entity set is a group of similar entities and these entities can have attributes. In terms of DBMS, an entity is a table or attribute of a table in database, so by showing relationship among tables and their attributes, ER diagram shows the complete logical structure of a database. Entity Relational (ER) Model is a high-level conceptual data model diagram. ER modelling helps you to analyse data requirements systematically to produce a well-designed database. The Entity-Relation model represents real-world entities and the relationship between them.



5.6 State Chart Diagram [17]

A state chart diagram describes a state machine which shows the behavior of classes. It shows the actual changes in state not processes or commands that create those changes and is the dynamic behavior of objects over time by model1ing the life cycle of objects of each class. It describes how an object is changing from one state to another state. There are mainly two states in State Chart Diagram: 1. Initial State 2. Final-State. Some of the components of State Chart Diagram are: State: It is a condition or situation in life cycle of an object during which it's satisfies same condition or performs some activity or waits for some event. Transition: It is a relationship between two states indicating that object in first state performs some actions and enters into the next state or event. Event: An event is specification of significant occurrence that has a location in time and space

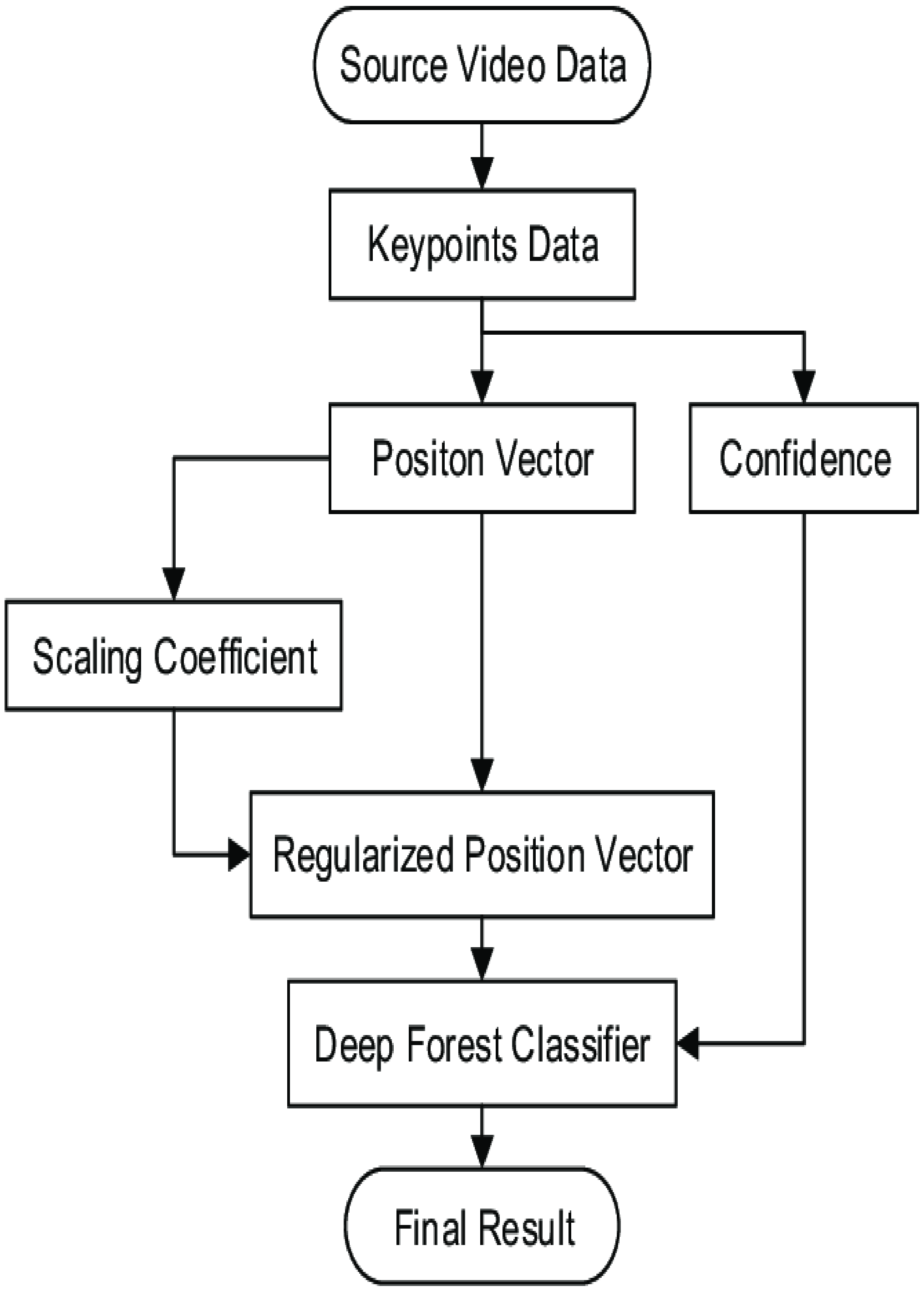


CHAPER 6. SYSTEM IMPLEMENTATION

6.1 Introduction [10]

Project implementation is the process of putting a project plan into action to produce the deliverables, otherwise known as the products or services, for clients or stakeholders. It takes place after the planning phase, during which a team determines the key objectives for the project, as well as the timeline and budget. Implementation involves coordinating resources and measuring performance to ensure the project remains within its expected scope and budget. It also involves handling any unforeseen issues in a way that keeps a project running smoothly.

6.2 Flowchart [22]



6.3 Coding

Training

```
import os
import cv2
import numpy as np
import tensorflow as tf
from tensorflow import keras
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Conv2D, MaxPooling2D, Flatten, Dense, Dropout
from tensorflow.keras.utils import to_categorical
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.callbacks import EarlyStopping
from sklearn.model_selection import train_test_split

# Set dataset directory
data_dir = "C:\\Users\\DELL\\Desktop\\Sign GPT\\ISL Dataset"  # Change this to your
dataset folder path
img_size = 64

def load_data():
    X, Y = [], []
    classes = sorted(os.listdir(data_dir))  # Sorted to ensure correct label order

    for label, sign in enumerate(classes):
        sign_path = os.path.join(data_dir, sign)
        for img_name in os.listdir(sign_path):
            img = cv2.imread(os.path.join(sign_path, img_name), cv2.IMREAD_GRAYSCALE)
            img = cv2.resize(img, (img_size, img_size))
            X.append(img)
            Y.append(label)

    X = np.array(X).reshape(-1, img_size, img_size, 1) / 255.0
    Y = np.array(Y)
    return X, Y, classes

# Load data
X, Y, class_names = load_data()
print(f"Loaded {len(X)} images across {len(class_names)} classes.")

# Save class labels for later use
with open("class_labels.txt", "w") as f:
    f.write("\n".join(class_names))

# Split dataset
```

```
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2, random_state=42)
Y_train = to_categorical(Y_train, num_classes=len(class_names))
Y_test = to_categorical(Y_test, num_classes=len(class_names))

# Data Augmentation
datagen = ImageDataGenerator(
    rotation_range=20,
    width_shift_range=0.2,
    height_shift_range=0.2,
    zoom_range=0.2,
    horizontal_flip=True
)

# Build CNN Model
model = Sequential([
    Conv2D(32, (3,3), activation='relu', input_shape=(img_size, img_size, 1)),
    MaxPooling2D(2,2),
    Conv2D(64, (3,3), activation='relu'),
    MaxPooling2D(2,2),
    Conv2D(128, (3,3), activation='relu'),
    MaxPooling2D(2,2),
    Flatten(),
    Dense(128, activation='relu'),
    Dropout(0.5),
    Dense(len(class_names), activation='softmax')
])

model.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])
model.summary()

# Early Stopping
early_stop = EarlyStopping(monitor='val_loss', patience=5, restore_best_weights=True)

# Train the Model
history = model.fit(
    datagen.flow(X_train, Y_train, batch_size=32),  # Data augmentation applied here
    validation_data=(X_test, Y_test),
    epochs=50,
    callbacks=[early_stop]
)

# Evaluate Model
loss, acc = model.evaluate(X_test, Y_test)
print(f"Test Accuracy: {acc * 100:.2f}%")

# Save Model
model.save("isl_model.h5")
```

```
# Load Model
model = keras.models.load_model("isl_model.h5")

# Load Class Labels
with open("class_labels.txt", "r") as f:
    class_names = f.read().splitlines()

def preprocess_frame(frame):
    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
    resized = cv2.resize(gray, (img_size, img_size))
    normalized = resized / 255.0
    reshaped = np.reshape(normalized, (1, img_size, img_size, 1))
    return reshaped

# Live Detection using Webcam
cap = cv2.VideoCapture(0)

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

    processed_frame = preprocess_frame(frame)
    prediction = model.predict(processed_frame)
    label = class_names[np.argmax(prediction)]

    cv2.putText(frame, label, (50, 50), cv2.FONT_HERSHEY_SIMPLEX, 1, (0, 255, 0), 2)
    cv2.imshow("Sign Detection", frame)

    if cv2.waitKey(1) & 0xFF == ord('q'):
        break

cap.release()
cv2.destroyAllWindows()


# Set dataset directory
data_dir = "C:\\Users\\DELL\\Desktop\\Sign GPT\\ISL Dataset"  # Change this to your
dataset folder path
img_size = 64

def load_data():
    X, Y = [], []
    classes = sorted(os.listdir(data_dir))  # Sorted to ensure correct label order

    for label, sign in enumerate(classes):
        sign_path = os.path.join(data_dir, sign)
        for img_name in os.listdir(sign_path):
            img = cv2.imread(os.path.join(sign_path, img_name), cv2.IMREAD_GRAYSCALE)
            img = cv2.resize(img, (img_size, img_size))
            X.append(img)
            Y.append(label)

    X = np.array(X).reshape(-1, img_size, img_size, 1) / 255.0
    Y = np.array(Y)
    return X, Y, classes

# Load data
X, Y, class_names = load_data()
print(f"Loaded {len(X)} images across {len(class_names)} classes.")

# Save class labels for later use
with open("class_labels.txt", "w") as f:
    f.write("\n".join(class_names))

# Split dataset
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2, random_state=42)
Y_train = to_categorical(Y_train, num_classes=len(class_names))
Y_test = to_categorical(Y_test, num_classes=len(class_names))

# Data Augmentation
datagen = ImageDataGenerator(
    rotation_range=20,
    width_shift_range=0.2,
    height_shift_range=0.2,
    zoom_range=0.2,
    horizontal_flip=True
)
# Build CNN Model
model = Sequential([
    Conv2D(32, (3,3), activation='relu', input_shape=(img_size, img_size, 1)),
    MaxPooling2D(2,2),
    Conv2D(64, (3,3), activation='relu'),
    MaxPooling2D(2,2),
    Conv2D(128, (3,3), activation='relu'),
    MaxPooling2D(2,2),
    Flatten(),
    Dense(128, activation='relu'),
    Dropout(0.5),
    Dense(len(class_names), activation='softmax')
])

model.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])
model.summary()
```

```
# Early Stopping
early_stop = EarlyStopping(monitor='val_loss', patience=5, restore_best_weights=True)

# Train the Model
history = model.fit(
    datagen.flow(X_train, Y_train, batch_size=32),  # Data augmentation applied here
    validation_data=(X_test, Y_test),
    epochs=50,
    callbacks=[early_stop]
)

# Evaluate Model
loss, acc = model.evaluate(X_test, Y_test)
print(f"Test Accuracy: {acc * 100:.2f}%")

# Save Model
model.save("isl_model.h5")

# Load Model
model = keras.models.load_model("isl_model.h5")

# Load Class Labels
with open("class_labels.txt", "r") as f:
    class_names = f.read().splitlines()

def preprocess_frame(frame):
    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
    resized = cv2.resize(gray, (img_size, img_size))
    normalized = resized / 255.0
    reshaped = np.reshape(normalized, (1, img_size, img_size, 1))
    return reshaped

# Live Detection using Webcam
cap = cv2.VideoCapture(0)

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

    processed_frame = preprocess_frame(frame)
    prediction = model.predict(processed_frame)
    label = class_names[np.argmax(prediction)]

    cv2.putText(frame, label, (50, 50), cv2.FONT_HERSHEY_SIMPLEX, 1, (0, 255, 0), 2)
    cv2.imshow("Sign Detection", frame)
    if cv2.waitKey(1) & 0xFF == ord('q'):
        break

cap.release()
cv2.destroyAllWindows()
```

App

```
import os
import cv2
import numpy as np
import tensorflow as tf
from tensorflow import keras
import streamlit as st

# Load trained model
model = keras.models.load_model("isl_model.h5")

# Load class labels
with open("class_labels.txt", "r") as f:
    class_names = f.read().splitlines()

img_size = 64

def preprocess_frame(frame):
    """Preprocesses image for model prediction."""
    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
    resized = cv2.resize(gray, (img_size, img_size))
    normalized = resized / 255.0
    reshaped = np.reshape(normalized, (1, img_size, img_size, 1))
    return reshaped

def predict_image(image):
    """Predicts the class of an uploaded image."""
    processed_img = preprocess_frame(image)
    prediction = model.predict(processed_img)
    label = class_names[np.argmax(prediction)]
    return label

# Streamlit UI
st.title("Indian Sign Language Detection")
st.write("Upload an image or use your webcam for real-time sign language detection.")

# File Upload
```

```
uploaded_file = st.file_uploader("Upload an image", type=["jpg", "png", "jpeg"])
if uploaded_file is not None:
    try:
        file_bytes = np.asarray(bytearray(uploaded_file.read()), dtype=np.uint8)
        image = cv2.imdecode(file_bytes, cv2.IMREAD_COLOR)
        if image is None:
            st.error("Error loading image. Please upload a valid file.")
        else:
            st.image(image, caption="Uploaded Image", use_column_width=True)
            label = predict_image(image)
            st.success(f"Prediction: {label}")
    except Exception as e:
        st.error(f"Error processing image: {str(e)}")

# Live Detection with Webcam
st.write("Press 'Start Webcam' to detect signs in real-time.")

if "webcam_running" not in st.session_state:
    st.session_state.webcam_running = False

if st.button("Start Webcam"):
    st.session_state.webcam_running = True

if st.button("Stop Webcam"):
    st.session_state.webcam_running = False

if st.session_state.webcam_running:
    cap = cv2.VideoCapture(0)
    stframe = st.empty()  # Create a placeholder for video frames

    while cap.isOpened() and st.session_state.webcam_running:
        ret, frame = cap.read()
        if not ret:
            st.error("Failed to capture image. Please check your webcam.")
            break

        processed_frame = preprocess_frame(frame)
        prediction = model.predict(processed_frame)
        label = class_names[np.argmax(prediction)]

        cv2.putText(frame, label, (50, 50), cv2.FONT_HERSHEY_SIMPLEX, 1, (0, 255, 0), 2)

        # Convert to RGB (Streamlit requires RGB images)
        frame = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)

        stframe.image(frame, channels="RGB")  # Display frame in Streamlit
    cap.release()


model = keras.models.load_model("isl_model.h5")

# Load class labels
with open("class_labels.txt", "r") as f:
    class_names = f.read().splitlines()

img_size = 64

def preprocess_frame(frame):
    """Preprocesses image for model prediction."""
    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
    resized = cv2.resize(gray, (img_size, img_size))
    normalized = resized / 255.0
    reshaped = np.reshape(normalized, (1, img_size, img_size, 1))
    return reshaped

def predict_image(image):
    """Predicts the class of an uploaded image."""
    processed_img = preprocess_frame(image)
    prediction = model.predict(processed_img)
    label = class_names[np.argmax(prediction)]
    return label

# Streamlit UI
st.title("Indian Sign Language Detection")
st.write("Upload an image or use your webcam for real-time sign language detection.")

# File Upload
uploaded_file = st.file_uploader("Upload an image", type=["jpg", "png", "jpeg"])
if uploaded_file is not None:
    try:
        file_bytes = np.asarray(bytearray(uploaded_file.read()), dtype=np.uint8)
        image = cv2.imdecode(file_bytes, cv2.IMREAD_COLOR)
        if image is None:
            st.error("Error loading image. Please upload a valid file.")
        else:
            st.image(image, caption="Uploaded Image", use_column_width=True)
            label = predict_image(image)
            st.success(f"Prediction: {label}")
    except Exception as e:
        st.error(f"Error processing image: {str(e)}")

# Live Detection with Webcam
st.write("Press 'Start Webcam' to detect signs in real-time.")
```

```
if "webcam_running" not in st.session_state:
    st.session_state.webcam_running = False

if st.button("Start Webcam"):
    st.session_state.webcam_running = True

if st.button("Stop Webcam"):
    st.session_state.webcam_running = False

if st.session_state.webcam_running:
    cap = cv2.VideoCapture(0)
    stframe = st.empty()  # Create a placeholder for video frames

    while cap.isOpened() and st.session_state.webcam_running:
        ret, frame = cap.read()
        if not ret:
            st.error("Failed to capture image. Please check your webcam.")
            break

        processed_frame = preprocess_frame(frame)
        prediction = model.predict(processed_frame)
        label = class_names[np.argmax(prediction)]

        cv2.putText(frame, label, (50, 50), cv2.FONT_HERSHEY_SIMPLEX, 1, (0, 255, 0), 2)

        # Convert to RGB (Streamlit requires RGB images)
        frame = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)

        stframe.image(frame, channels="RGB")  # Display frame in Streamlit

    cap.release()
```

6.4 Testing Approach [10]

To ensure the reliability, usability, and security of our Indian Sign Language Recognition system, we adopted a multi-pronged testing strategy. This included unit testing**,** usability testing**,** and security testing, each aimed at validating different aspects of the system from model accuracy to user interaction and application safety..

6.5 Unit Testing

Unit testing was applied to individual components of the system, such as image preprocessing functions, model prediction functions, and utility scripts responsible for data handling and normalization. These tests ensured that each function behaved as expected for example, that grayscale conversion produced consistent outputs, and that the trained model returned valid predictions for known inputs. Unit testing was also used to validate error handling in cases of unsupported or corrupted image files. This testing was crucial in maintaining code stability as we iteratively improved the model and interface.

6.6 Usability Testing

Usability testing focused on evaluating the interface from an end-user perspective. Participants were asked to perform tasks such as:

- Uploading gesture images for prediction
- Using the live webcam-based detection feature
- Interpreting the predicted output

We gathered feedback on clarity, responsiveness, and ease of navigation. Observations revealed improvements needed in labeling, loading times, and interface responsiveness. Based on this, the interface was refined to provide a smoother, more intuitive experience for users of varying technical backgrounds, including students, teachers, and differently-abled individuals.

.

6.7 Security Testing

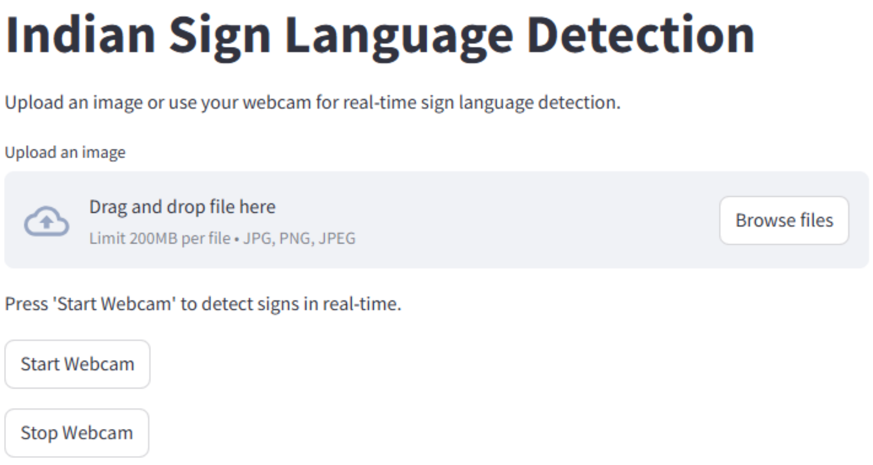
Security testing was conducted to ensure the safety of the system, especially given its web interface. Key considerations included:

- **File upload validation**: Ensuring that uploaded images are safe, within acceptable formats (e.g., .jpg, .png), and within size limits.
- **Input sanitization**: Preventing malicious inputs through form submissions.
- **Live webcam access**: Ensuring that webcam access is explicitly authorized and properly terminated.

6.8 Test Cases

Test-Case ID	Case	Action	Expected Results	Actual Results	Status
1	Upload Image – Detection	Upload a palm gesture image for recognition.	Correct Alphabet or Number Displayed	Correct Alphabet or Number Displayed	Pass
2	Live Detection via Webcam	Show a hand gesture in front of the camera.	Real-time Detection of Alphabet/Number	Real-time Detection of Alphabet/Number	Pass
3	Invalid Image Input	Upload a non-hand or blurred image	"Invalid Gesture" or Error Message Displayed	"Invalid Gesture" or Error Message Displayed	Pass
4	Reset/Refresh Detection	Try detecting a new gesture immediately after one is recognized.	Previous Detection Cleared and New Detected	Previous Detection Cleared and New Detected	Pass

CHAPTER 7. RESULTS

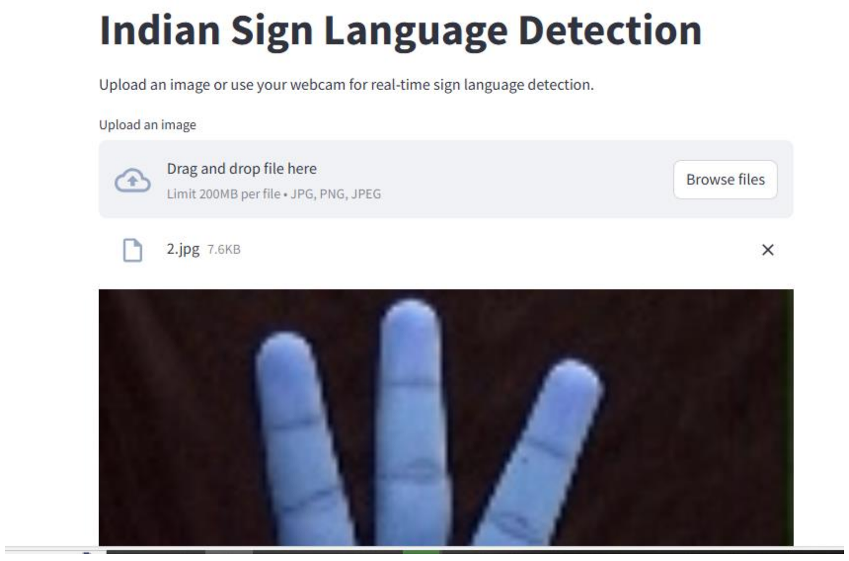


Indian Sign Language Detection

Upload an image or use your webcam for real-time sign language detection.

Upload an image

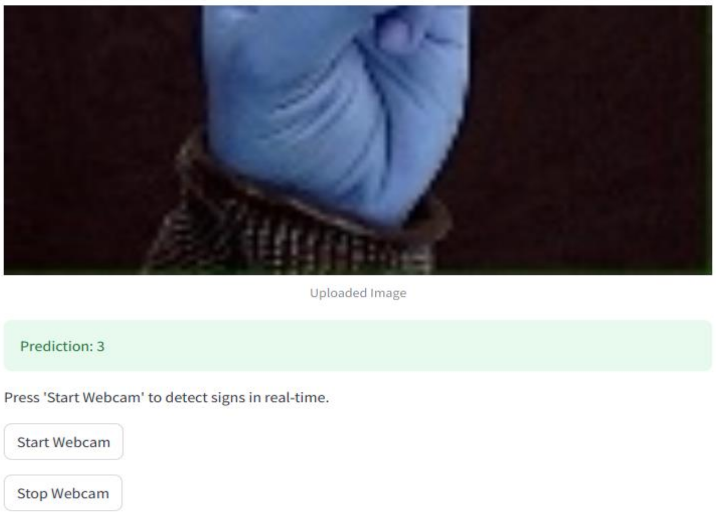
Press 'Start Webcam' to detect signs in real-time.



Indian Sign Language Detection

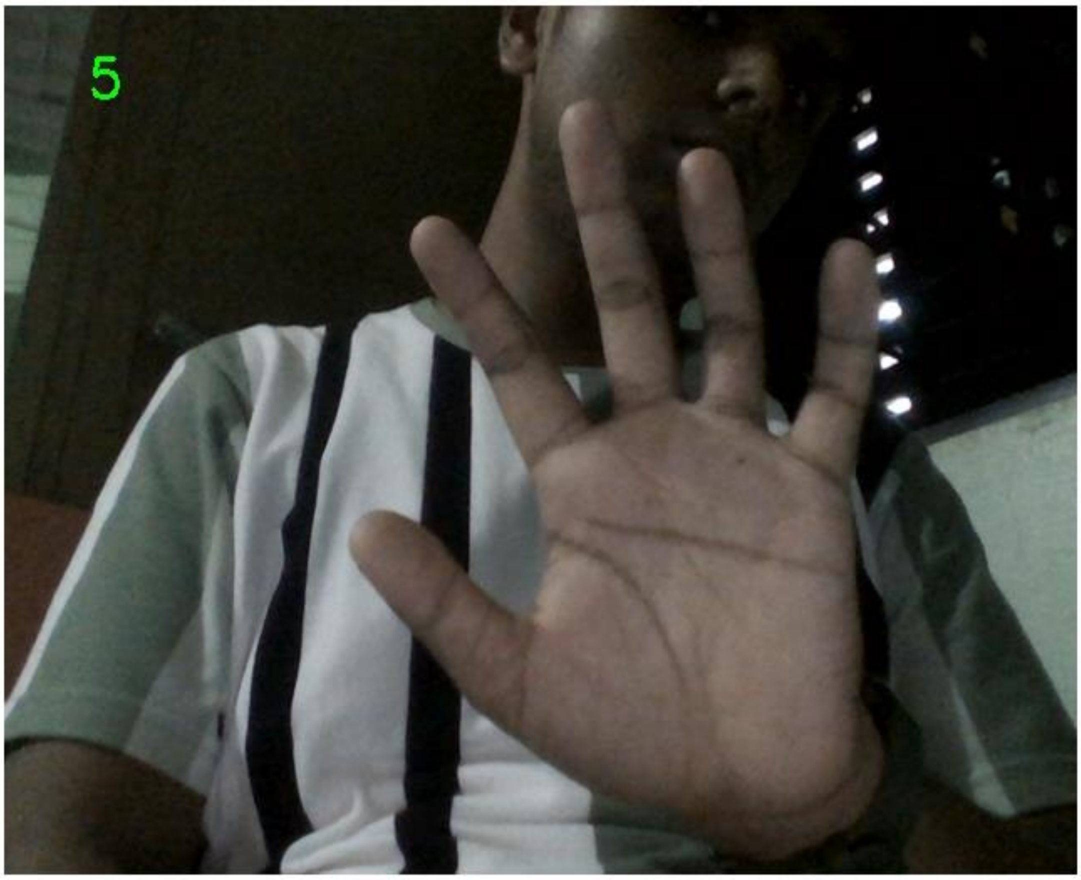
Upload an image or use your webcam for real-time sign language detection.

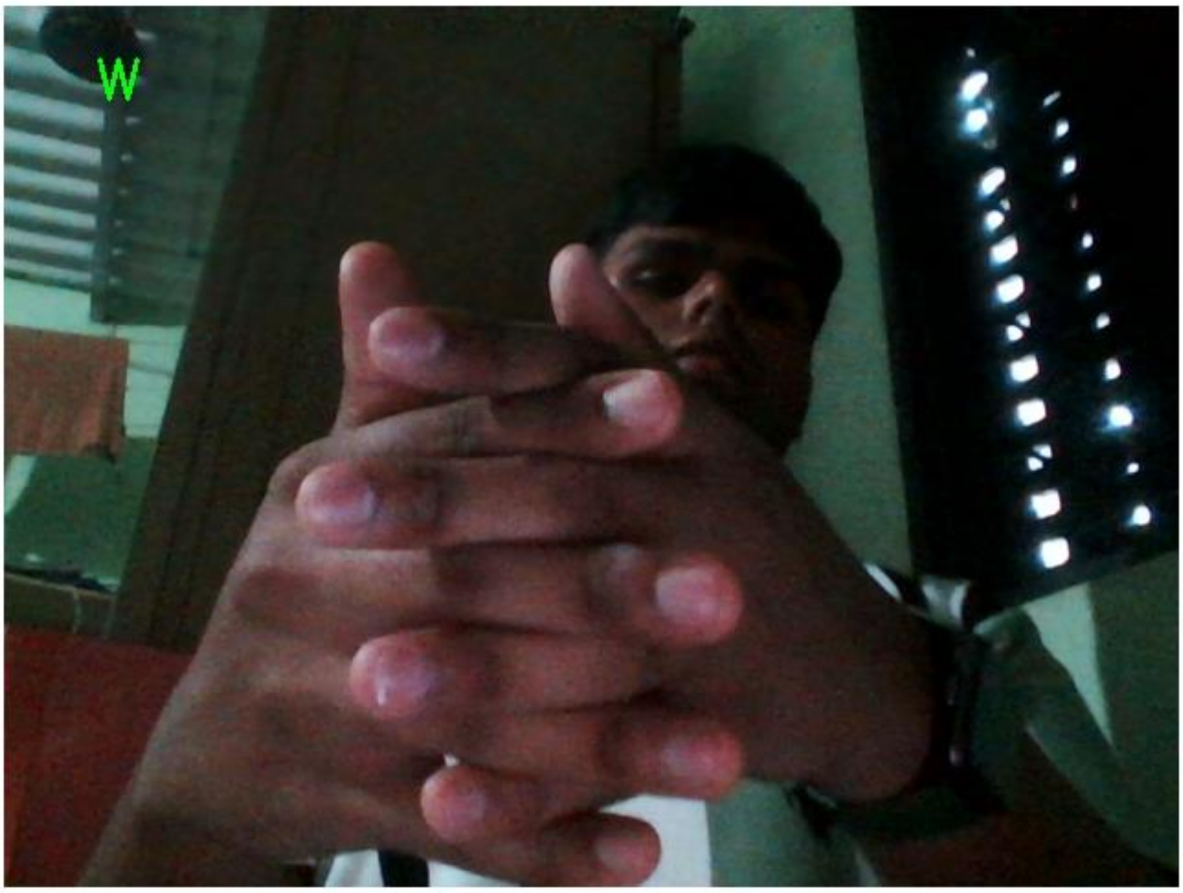
Upload an image


Uploaded Image

Prediction: 3

Press 'Start Webcam' to detect signs in real-time.




W

CHAPTER 8. CONCLUSION AND FUTURE SCOPE

8.1 Conclusion [18]

The Indian Sign Language Recognition System is a step forward in bridging communication gaps between the hearing and speech-impaired community and the rest of society. This project successfully demonstrates how computer vision and deep learning can be harnessed to recognize and interpret hand gestures representing alphabets and numbers in Indian Sign Language. By incorporating grayscale image preprocessing, a robust CNN model, and a user-friendly interface, the system allows users to either upload gesture images or use real-time webcam detection to identify signs with promising accuracy.

Our project contributes to the broader vision of inclusivity and accessibility in technology, especially in educational and assistive domains. While there is still room for improvement, the current implementation sets a strong foundation for future advancements in real-time sign language interpretation and broader language model support.

8.2 Future Scope and Enhancement [18]

As the project evolves, there are several opportunities to enhance the functionality, performance, and user experience of the Sign Language Recognition System:

1. **Real-time Sentence Recognition**
   Future versions can move beyond single-character detection to support continuous gesture tracking for forming complete words and sentences, allowing for more natural communication.
2. **Support for Dynamic Gestures**
   Many signs in Indian Sign Language involve motion. Enhancing the model to capture dynamic gestures using LSTM (Long Short-Term Memory) or other temporal models can expand the system's capabilities significantly.
3. **Multilingual Output**
   Adding the ability to translate recognized signs into different languages (e.g., Hindi, English, Tamil) can make the tool useful across diverse linguistic groups.
4. **Mobile App Integration**
   Developing a mobile version of the application will enable broader accessibility and portability, especially for daily communication needs in real-world scenarios.
5. **Voice Output and Text-to-Speech Integration**
   Adding a voice assistant that reads out the recognized sign will enhance communication for speech-impaired users by enabling them to "speak" using hand gestures.

Chapter 9. References

9.1 Project References

1. https://youtu.be/kIOTtRf9Drk?si=BHOmCA-ktHxiBZa4 – R&D.
2. https://youtu.be/zC22yPmc6Kw?si=or5Ibzw5H9_dBAbd – Ghantt Chart.
3. E-Commerce strategy methodology development and implementation | Systems Design Engineering | University of Waterloo (uwaterloo.ca) – Methodology
4. https://en.wikipedia.org/wiki/Sign_language – Introduction.
5. How to Write An effective Project Objective, With Examples [2024] • Asana Objectives.
6. https://www.quora.com/What-are-the-advantages-and-disadvantages-of-a-signed-language - Advantages & limitations.
7. https://www.w3schools.com/python/python_intro.asp – Python.
8. Bridging Tech Gaps with Technology Surveys (voxco.com) – Survey of Technology.
9. UML models and diagrams - IBM Documentation – System Design.
10. Software Testing - Implementation Testing (tutorialspoint.com) – Implementation & Testing.
11. System Analysis and Design - Overview (tutorialspoint.com) – System Analysis and Design.
12. https://docs.opencv.org/4.x/d0/de3/tutorial_py_intro.html - Opencv.
13. https://www.w3schools.com/python/pandas/pandas_intro.asp - Pandas
14. https://www.tensorflow.org/learn - Tensorflow.
15. https://creately.com/diagram/example/jul8esgx1/new-e-commerce-activity-diagram-classic - Activity Diagram.
16. Simple ER Diagram Template | Creately – ER Diagram.
17. https://ijrpr.com/uploads/V2ISSUE9/IJRPR1329.pdf - Stae Chart Diagram.
18. https://www.sciencedirect.com/topics/computer-science/sign-language-recognition – Future Scope.
19. https://keras.io/guides/distributed_training_with_torch/ – Keras.
20. What is Spiral Model and How is it Used? (techtarget.com) – Spiral Model.
21. What Does a Plagiarism Report Consist of? | Copyleaks – Plagiarism Report.
22. https://www.researchgate.net/figure/Flow-chart-diagram-Online-e-commerce-shopping-web-app_fig1_347169962 - Flow Chart.